



Release
18.01.10
Last update:
November
2019

ADVANCED DIGITAL DATA, INC.

Engineering department

ADD Energy SmartConnect

Programming Manual

Copyright © 2018 All rights Reserved Advanced Digital Data, Inc.

The contents of this manual and the software it describes are protected by copyright. Any reproduction in whole or in part is strictly prohibited. For additional copies of the manual, contact Advanced Digital Data, Inc.

Advanced Digital Data, Inc.
6 Laurel Drive
Flanders, New Jersey 07836
Tel: (973) 584-4026
Fax: (973) 584-3205

Web Site: www.addsys.com

Note

SmartConnect is constantly evolving software. This manual may not reflect with 100% accuracy with your installed version of SmartConnect. The best way is to query the wsdl of each class and get what it is expecting

Table of Contents

Note	3
Table of Contents	4
Revision History	14
<i>Introduction</i>	24
Overview	24
Licensing	25
Basic Concepts	27
Security	28
Main Classes	30
<i>CUserProfile Class</i>	31
Overview	31
cUserProfile.CreateUser()	32
cUserProfile.ModifyUserInfo()	34
cUserProfile.ChangeEmailAddress ()	36
cUserProfile.ChangePassword()	38
cUserProfile.DeleteUser()	40
cUserProfile.GetUserInfo()	42
cUserProfile.LinkAccount ()	45
cUserProfile.UnLinkAccount ()	47
cUserProfile.ListAccounts ()	49
cUserProfile.Login()	52
cUserProfile.GetPassword()	54
cUserProfile.ChangeDefaultAccount()	56
<i>cCustomerInfo Class</i>	58
Overview	58
Data Classes and Datasets	59
cCustomerInfo.ValidateAccount()	60
cCustomerInfo.ValidateAccountWithInfo()	62
cCustomerInfo.GetValidAccountList()	65
cCustomerInfo.GetCustomerGeneralInfo()	68
cCustomerInfo.GetCreditInfo()	72

cCustomerInfo.GetCreditInfoPPD()	75
cCustomerInfo.GetGeneralText()	77
cCustomerInfo.GetDeliveryLocations()	79
cCustomerInfo.GetServiceLocations()	81
cCustomerInfo.GetWDMSLocations()	83
cCustomerInfo.GetMeters()	85
cCustomerInfo.GetItsProducts ()	87
cCustomerInfo.GetAccountsByGroupCode()	89
<i>cDelivery Class</i>	92
Overview	92
Data Classes and Datasets	93
cDelivery.GetDeliveryGeneralInfo ()	102
cDelivery. GetProductPrice()	104
cDelivery.CalculateDeliveryUnits()	106
cDelivery.GetTicketInfo()	108
cDelivery.GetDeliveryHistory()/cDelivery.GetDeliveryHistoryWithCredits()	110
cDelivery.LogPhoneOrder()	112
cDelivery.LogPhoneOrderCC()	114
cDelivery.LogPhoneOrder_V2()	115
cDelivery.ReschedulePhoneOrder()	117
cDelivery.ReschedulePhoneOrder_V2()	119
cDelivery.CancelPhoneOrder()	120
cDelivery.CancelPrintedTicket()	121
cDelivery.GetDegreeDayInfo()	123
cDelivery. ITRForecasting ()	125
cDelivery.LoadScheduledProducts()	127
cDelivery.LogScheduledProductPhoneOrder()	129
cDelivery.ComputeTaxes()	132
cDelivery.GetDeliveryARAddons()	135
<i>cService Class</i>	137
Overview	137
cService.GetGeneralInfo()	138
cService.GetServiceCallInfo()	140

cService.GetServiceHistory()	142
cService.GetCallReason()	144
cService.GetDaySegment()	146
cService.GetAvailableSlot()	148
cService.LogWorkOrder()	150
cService.LogWorkOrderNTC()	153
cService.RescheduleWorkOrder()	157
cService.RescheduleWorkOrderWithNotes()	160
cService.RescheduleWorkOrderNTC()	163
cService.CancelWorkOrder()	165
cService.GetServiceWaitTime()	166
cService.SaveServiceWaitTime()	168
cService.DeleteServiceWaitTime()	170
cService.UpdateCallInformation()	171
cService.LoadTechnicians()	173
<i>CTransaction Class</i>	174
Overview	174
cTransaction.GetARTransactions() / GetARTransactionsWithNoCreditActions()	175
cTransaction.GetARTransactionsPPD()	178
cTransaction.AcceptPayment()	180
cTransaction.AcceptPaymentDiscount()	182
cTransaction.AcceptPaymentReference()	184
cTransaction.GetNewUploadTransactionDataSet ()	186
cTransaction.UpLoadTransactions()	187
cTransaction.LoadPreDefinedComments()	190
cTransaction.AcceptPayment_V2()	191
cTransaction.LoadFinancialInstitutions()	194
cTransaction.GetARTransactions_V2()	195
cTransaction.CreateAdjustmentTransferEntry()	206
cTransaction.GetAssetDeliveryDetails()	208
<i>cLookup Class</i>	211
Overview	211
cLookup.GetCompanyName()	212

cLookUp.GetDivisionName()	214
cLookup.GetLookupStoredValues()	216
cLookup.ValidateZipCode()	218
<i>cUtility Class</i>	220
Overview	220
cUtility.GetServerDateTime()	221
cUtility.GetErrorDescription()	222
cUtility.SetLogVerbosity()	223
cUtility.GetDatabaseVersion()	225
cUtility.GetEnergyAPIVersion()	226
cUtility.ClearCacheData()	227
cUtility.SetDataCacheRefreshTimeout()	228
cUtility.GetDataCacheRefreshTimeout()	229
<i>cCustomerLog Class</i>	230
Overview	230
cCustomerLog.CreateLogEntry()	231
cCustomerLog.CreateLogEntryWithAttachment()	236
cCustomerLog.LoadLogAttachmentBytes()	241
cCustomerLog.LoadLogAttachmentFile()	243
cCustomerLog.SaveLogAttachmentBytes()	245
cCustomerLog.SaveLogAttachmentFile()	247
cCustomerLog.RemoveLogAttachment()	249
cCustomerLog.CleanupLocalResource()	251
cCustomerLog.FindCustomerLogEntries()	252
cCustomerLog.LoadCustomerLogDetails()	255
cCustomerLog.UpdateCustomerLogEntry()	257
cCustomerLog.LoadCustomerLogTemplates()	262
cCustomerLog.LoadCustomerLogFollowupResources()	264
cCustomerLog.LoadCustomerLogDivisions()	266
cCustomerLog.LoadCustomerLogCollectors()	268
<i>CCustomerMaint class</i>	270
Overview	270
cCustomerMaint.GetCustTemplates()	272

cCustomerMaint.GetCustMasterFields()	275
cCustomerMaint.AddCustMaster()	280
cCustomerMaint.UpdateCustMaster()	282
cCustomerMaint.GetTankFields()	317
cCustomerMaint.AddTank()	321
cCustomerMaint.UpdateTank()	323
cCustomerMaint.GetMeterFields()	325
cCustomerMaint.AddMeter()	328
cCustomerMaint.UpdateMeter()	330
cCustomerMaint.GetServiceFields()	332
cCustomerMaint.AddService()	336
cCustomerMaint.UpdateService()	338
cCustomerMaint.GetWdmsFields()	340
cCustomerMaint.AddWdms()	343
cCustomerMaint.UpdateWdms()	345
cCustomerMaint.GetItsFields()	347
cCustomerMaint.AddItsProduct()	350
cCustomerMaint.UpdateItsProduct()	352
cCustomerMaint.GetContactTypes()	354
cCustomerMaint.LoadCustContactInfo()	356
cCustomerMaint.SaveCustContactInfo()	358
cCustomerMaint.DeleteCustContactInfo()	361
cCustomerMaint.LoadDeliveryStopCodes()	363
cCustomerMaint.LoadDeliveryStopInfo()	364
cCustomerMaint.SetDeliveryStop()	366
cCustomerMaint.ClearDeliveryStop()	368
cCustomerMaint.GetTemplates()	370
cCustomerMaint.AddOrUpdateCustomerRecord()	373
<i>CPlanPrice class</i>	390
Overview	390
cPlanPrice.LoadStatus()	391
cPlanPrice.LoadPlans()	392
cPlanPrice.LoadTanks()	395

cPlanPrice.LoadProducts()	396
cPlanPrice. LoadAlternateEligibleProducts()	397
cPlanPrice. LoadAlternateEligiblePricing()	398
cPlanPrice. LoadPlanCodeValidationInfo()	399
cPlanPrice. LoadPlanCodesWithValues()	402
cPlanPrice. LoadPlanTypes()	405
cPlanPrice. LoadSalesTypes()	406
cPlanPrice. LoadCostTypes()	407
cPlanPrice. LoadDepartmentCodes()	408
cPlanPrice. GetProjectedUsage()	409
cPlanPrice. Save()	411
SLPPData	416
<i>cCustBatch class</i>	418
Overview	418
CCustBatch.CustBatchUpdate()	419
CCustBatch.CustBatchAdd()	421
CCustBatch.GetLogHeaderDS()	424
CCustBatch.GetLogDetailDS()	425
CCustBatch.CustBatchLogEntry()	426
CCustBatch.BatchUpdateMaster()	428
CCustBatch.BatchUpdateTank()	431
CCustBatch.BatchUpdateService()	434
CCustBatch.BatchUpdateWDMS()	437
<i>cCreditCard class</i>	440
Overview	440
Vault	441
Data Classes and Datasets	442
cCreditCard.LoadCreditCardTypes()	449
cCreditCard.LoadSourcesOfCharge()	451
cCreditCard.VerifyCardNumber()	452
cCreditCard.ValidateCreditCard()	454
cCreditCard.LoadCCCCustomerInfo()	457
cCreditCard.LoadCCCCustomerSourcesOfCharge()	459

cCreditCard.SaveCCCustomerInfo()	461
cCreditCard.GetCCTransactionEligibility	467
cCreditCard.GetCCMerchantIDForCC()	471
cCreditCard.PreAuthorizeCreditCard()	473
cCreditCard.ChargeCreditCard()	476
cCreditCard.VoidCreditCard()	479
cCreditCard.EncryptCardNumber()	482
cCreditCard. GetCCMerchantIdAndVaultModeForCustomer()	484
cCreditCard. GetCCVaultModeForCCMerchant()	486
General steps to accepting a payment:	488
Was the transaction successful?	489
General steps to logging a phone order with CC:	490
General Credit Card Method Return Codes	492
<i>cEFT class</i>	493
Overview	493
cEFT.LoadAccountTypes()	494
cEFT.LoadWeekendDrafting()	496
cEFT.LoadCreditLineDrafting()	497
cEFT.LoadPostingGroup()	498
cEFT.GetEFTCustomerDetails()	499
cEFT.GetEFTCustomerLocations()	501
cEFT.GetEFTCustomerNotifications()	502
cEFT.GetEFTCustomerPostingGroups()	503
cEFT.SaveEFTCustomerDetails()	504
cEFT.TeleDraft_AccountQuery()	507
<i>cCustomerDocument class</i>	508
Overview	508
cCustomerDocument.LoadDocumentTypeDesc ()	509
cCustomerDocument. LoadCustomerDocuments ()	510
cCustomerDocument.GetDocumentImage ()	512
cCustomerDocument.AddOnDemandDocument()	514
Overview	516
cContactInfo.LoadContactTypeIDs()	519

cContactInfo.LoadContactTypes() _____	521
cContactInfo.LoadContactInfo () _____	523
cContactInfo.DeleteContactInfo () _____	531
cContactInfo.LoadContactSummary() _____	533
cContactInfo.LoadDocuments() _____	535
cContactInfo.LoadDocumentOutputFormats() _____	537
cContactInfo.SaveContactInfoWithDocuments () _____	539
<i>cOrder class</i> _____	543
Overview _____	543
cOrder.SaveOrders() _____	545
<i>cBestBuy class</i> _____	554
Overview _____	554
cBestBuy.GetBestBuy() _____	555
<i>PriceQuote class</i> _____	580
Overview _____	580
PriceQuote. GetPriceQuotes() _____	580
REQUEST PARAMETERS _____	581
RESPONSE CLASSES: _____	582
.XML Documents: _____	582
<i>CreditAction Class</i> _____	585
Overview _____	585
CreditAction.LoadAcquisitionReasonCodes() _____	585
CreditAction.LoadCreditActionCodes() _____	587
CreditAction.LoadTerminationCodes() _____	589
CreditAction.CreateCreditAction() _____	591
CreditAction.TerminateAccount() _____	595
CreditAction.ReInstateAccount() _____	599
CreditAction:Error Codes _____	605
<i>LocationStatus class</i> _____	606
Overview _____	606
Data Classes and Datasets _____	607
LocationStatus.LoadCanceledReasons() _____	612
LocationStatus.LoadReinstateReasons() _____	613

LocationStatus.GetDeliveryLocationStatus()	614
LocationStatus.CancelDeliveryLocation()	615
LocationStatus.ReinstateDeliveryLocation()	617
LocationStatus.GetServiceLocationStatus()	619
LocationStatus.CancelServiceLocation()	620
LocationStatus.ReinstateServiceLocation()	622
LocationStatus.GetWdmsLocationStatus()	624
LocationStatus.CancelWdmsLocation()	625
LocationStatus.ReinstateWdmsLocation()	627
<i>Contract Class</i>	629
Overview	629
Contract.LoadCostTypes()	629
Contract.LoadDepartmentCodes()	630
Contract.LoadMFGCodes()	631
Contract.LoadGroupCodes()	632
Contract.LoadStockIDS()	633
Contract.LoadPostingCodes()	634
Contract.LoadSalesPersons()	635
Contract.LoadSalesTypes()	636
Contract.LoadSuppliers()	637
Contract.LoadSupplyLocations()	638
Contract.LoadWdmsLocations()	639
Contract.LoadContracts()	640
Contract.LoadContract()	641
Contract.LoadSlaveContracts()	642
Contract.SaveContract()	644
<i>Appendix A</i>	654
Enterprise v. Web Service Overview	654
<i>Appendix B</i>	658
SmartConnect Templates	658
Master (template –900)	659
Delivery (template –901)	661
Service (template –902)	663

WDMS (template –903) _____	665
Transaction Upload Data (DSARTransactions) _____	666
<i>Appendix C</i> _____	673
All Return Codes & Descriptions _____	673
<i>Appendix D</i> _____	688
Getting Started _____	688
Code Snippet Examples _____	688
Using Visual Studio 2005 _____	689
Using Visual Studio 2008 _____	690
Using Visual Web Developer 2010 Express _____	691
How to test your request _____	692
<i>Appendix E</i> _____	692
cOrder schema _____	692
<i>Appendix F</i> _____	699
EnergyAPI.CustomerLogSelectionCriteria _____	699
Customerlog.CustomerLogHeader _____	701
DCEnergy.Customerlog.CustomerLogDetail _____	704

Revision History

Revision	Date	Author	Description
1.0	06/10/2002	MB	Preliminary specification describing the Energy API interface, to be submitted to Chrysalis, DMS and Petro engineers for discussion purposes.
2.0	06/27/2002	MB	1) Modified cCustomerInfo.GetCreditInfo() method to return both the budget payment amount and budget payment date 2) Added cUtility.GetErrorDescription():this should be called to get a specific error message pertaining to an error code
3.0	07/04/2002	MB	Added 2 new methods: cUtility.SetLogVerbosity() and cUtility.GetEnergyAPIVersion()
3.1	07/08/2002	MB	Added "Confirmation number" when logging/rescheduling a phone order
3.2	02/11/2003	MB	The "ValidateAccountWithInfo()" method was modified to return the "customer type" field
3.2a	02/14/2003	MB	Added the undocumented Customer log class (cCustLogEntry) to the API document
4.0	July 2003	MB/DC /KM	Added the Account maintenance API calls + Minor other fixes
4.1	Aug 2003	MB	1) Modified the Cdelivery.GetGeneralInfo() method to add the following tank information: will call flag, Period tank flag, day of week scheduling, Delivery interval, Usage rate per day, Estimated inventory, days in reserve 2) Modified the CcustomerInfo.GetCreditInfo() method to return the date & time of the very last AR transaction posted to the account.
4.2	Sept. 2003	MB	ValidateAccountWithInfo() method was modified to return error 103 if No account is found; it will return error # 144 if more than 1 account is found
4.3	Feb 2004	DC	Introduced new methods for task 37521 Service W/O API. New methods in CService are: GetCallReason(), GetDaySegment(),GetAvailableSlot(),LogWorkOrder(), ReScheduleWorkOrder, CancelWorkOrder. GetGeneralInfo, GetServiceCallInfo were modified.
4.3a	Mar 2004	DC	All methods having p_iDaysofWeek were showing examples using "Y" and "N" when it should represent a bitmask Changed all occurrences of p_lErrorCode as Long to p_lErrorCode as Variant RescheduleWorkOrder method had parameter list corrected.
4.4	May 2004	DC	Introduction of methods in cService: GetServiceWaitTime, SaveServiceWaitTime and DeleteServiceWaitTime. Created for task 37670. Pages 82-86.

Revision	Date	Author	Description
			Correction to example for cService.CancelWorkOrder
4.5	July 2004	KM	1) cCustomerInfo.GetDeliveryLocations → modified to return tank_status 2) cCustomerInfo.GetServiceLocations → modified to return service_status 3) cCustomerInfo.GetWdmsLocations → modified to return location_status
4.6	08/23/2004	MB	Modified the cCustomerInfo.GetGeneralInfo() method to include the “Account Priority” flag Y/N in the list of returned values
4.7	08/25/2004	DC	- Added “Information” sidebars to parameters/return values that were not clear. Pages modified 70,72,74,77,79 - Added ‘Appendix’ section that introduces the field names for the Customer Maintenance templates.
4.8	06/07/2005	MB	Proj. 37725 Call #65240 – Modified the cService.GetGeneralInfo() method to add field # 14 ("number_peg_cleaning_calls") - # of Uncompleted Pegasus cleaning calls
4.9	06/29/2006	JF	Updated the documentation to reflect Visual Studio software upgrade changes. Mostly changed ADODB.Recordsets to System.Data.Datasets with some other minor variable changes.
5.0	08/25/2006	DC	Proj: 37731 (88262ADD) Appendix section modified. Added newly added fields for Customer Maintenance (master, tank,service) to list.
5.1	08/31/2006	KM	Proj 37731 #88262ADD Added new methods (Customer Contact Information) to the cCustomerMaint class. GetContactTypes, LoadCustContactInfo, SaveCustContactInfo, DeleteCustContactInfo
5.2	11/29/2006	CG	Added component overview diagram from installation instruction (Security section)
5.3	11/15/2007	DC	153144ADD. Modified code/documentation with regards to “special consideration” fields delivery_address (pg 126) and service_address (pg 141).
5.4	1/23/2008	DC	140054ADD. Added 2 new fields to the Master CM template; balance and data_conversion. Refer to pg 185/186 for details
5.5	2/01/2008	CB	140054ADD. Added Transaction Data Class and Transaction Upload Method.
5.6	05/20/2008	CEB	Added extra fields to cCustomerInfo.GetCreditInfo resultset
7.1.3	09/29/2008	KRM	Updated documentation for CcustomerLog.CreateLogEntry

Revision	Date	Author	Description
7.1.3	12/04/2008	DC	Introduced the cCustBatch branch.
7.1.3	01/06/2009	DC	Introduced the Enterprise v. Web Service section
9.1.2	05/15/2009	DC	Introduced the cCreditCard branch
9.1.2	07/16/2009	DC	Corrected release version from above revision entry to 9.1.2. Documented additional return fields for methods "GetGeneralInfo","GetProductPrice","GetCustTemplates", "GetCustMasterFields","GetTankFields","GetServiceFields", "GetMeterFields","GetWDMSFields" and "GetItsFields".
9.1.3	07/21/2009	CF	Added 2 new fields "Delivery GetGeneralInfo" – Added 'Tank Status (199490) "GetProductPrice" – Added 'Tax amount' (198185)
9.1.3	07/21/2009	KM	199591ADD – added void method to CC interface
9.1.3	7/24/2009	DC	Added cEFT
9.1.3	08/4/2009	CF	198187ADD – added CalculateDeliveryUnits to Delivery
9.1.4	08/12/2009	MB	199521ADD – Added cDelivery.GetDegreeDayInfo
9.1.4	08/26/2009	KM	No Call primarily for internal use – added ClearCacheData method to ADDDataCache.dll & a New Method In EnergyAPI.cUtility
9.1.4	11/06/2009	CF	203993ADD – Added Inventory Reading Function to Delivery
9.1.9	03/05/2010	CF	210709ADD – Changed get delivery history to return newest to oldest.
9.1.9	03/08/2010	RAC	Updated license descriptions
10.1.0	03/29/2010	CF	Added 'Has Documents' and 'trx unique key' to get AR Transactions.
10.1.0	04/01/2010	CF	Added Return Codes.

Revision	Date	Author	Description
10.1.0	04/27/2010	CEB	Added cCustomerDocument class.
10.1.0	04/29/2010	CEB	Added cContactInfo class. <u>This class must be used with 10.01 and higher level E3 databases and replaces the cCustomerMaint Load and Save contact info API calls as of release 10.01.</u>
10.1.0	04/30/2010	MB	Updated the Overview and Basic Concepts sections to remove the “COM+” references and add support for Web Services specification.
10.1.0	05/06/2010	CF	Added note to Customer Maintenance about DIN.
10.1.1	09/17/2010	CF	Added minimum size for passwords and email address in User Profile.
10.1.3	10/06/2010	CF	Added note and correction to cDelivery.GetProductPrice.
11.01.00	01/13/2011	FB	Added extra output to cDelivery.GetGeneralInfo Tank size, Reserve, K-factor
-----	3/24/2011	CG	Updated Licensing Information
	3/28/11	CG	Adjusted format to be a consistent look
11.1.0	04/01/2011	CF &DC	Added Getting started.
11.1.1	05/30/2011	FB	Added cUserProfile.ChangeEmailAddress () method.
11.01.01	06/08/2011	MB	Updated license descriptions
11.01.01	06/17/2011	MB	Added cTransaction.AcceptPaymentDiscount() WebMethod to accept payments and allow for an additional amount to post to the account for prompt pay discount.
12.01.00	01/17/2012	MB	Added cTransaction.AcceptPaymentReference WebMethod to accept payments and allow for a payment reference number between 0 and 8388350.
12.01.00	01/18/2012	CEB	Added the ability to link documents for transmission to Customer Contact records – New methods were added to the cContactInfo class for this.

Revision	Date	Author	Description
12.01.00	01/19/2012	MB	Added extra output to the cDelivery.GetDeliveryHistory() method: partial fill flag and in tank units
12.01.00	01/24/2012	CF	Fixed cCustomerInfo.GetCustomerGeneralInfo() description.
12.01.01	05/01/2012	KRM	218858ADD – Updated information related to new “CC Vault” processing
12.01.01	06/29/2012	DC	266336ADD – Added WDMS customer maintenance template definition in Appendix.
12.01.01	07/27/2012	CF	Updated cContactInfo.
12.01.02	09/21/2012	KRM	270516ADD – return code if unable to unlink account 253821ADD – Added ability to use Pre-defined & user comments when accepting payments, also added method to load pre-defined payments
12.01.02	09/27/2012	CF	Updated cTransaction with xml examples.
12.01.03	2012-06-07	DC	GetCreditInfo() – added 2 additional return values (prepayment_amount and installment_balance)
12.01.03	2012-10-01	JF/KM	271858ADD – added new column “vault_change_made” to DSCCCustomerHeader dataset.
12.01.03	2012-10-04	CF	Updated cCustomerMaint Notes.
12.01.03	2012-11-12	DC	Introduced cCustBatch.BatchUpdateMaster() and cCustBatch.BatchUpdateTank()..Future releases will have BatchUpdateService(),BatchUpdateWDMS()
12.01.03	2012-11-20	CF	Updated cDelivery.GetDeliveryGeneralInfo() Notes.
12.01.03	2013-03-05	CF	Updated cPlanPrice.Save() Notes.
12.01.03	2013-03-12	CF	Updated cCreditCard.
12.01.04	2013-03-27	KM	Call #283380ADD Added “GetCCMerchantIdAndVaultModeForCustomer” & “GetCCVaultModeForCCMerchant” to cCreditCard

Revision	Date	Author	Description
12.01.04	2013-04-18	CF	Updated cCreditCard.
13.01.01	2013-07-15	MB	Call# 255871 Added partial fill reason, runout flag, runout reason fields to delivery history record
13.01.01	2013-07-16	MB	Call# 269374 Added GetValidAccountList() WebMethod to return the list of customer numbers that have identical telephone numbers.
13.01.01	2013-07-17	KM	Call #279261 Added LoadScheduledproducts() & LogScheduledProductPhoneOrder() methods to cDelivery module
13.01.01	2013-07-24	KM	Call #283982ADD Added methods to Set/Get data cache refresh timeout
13.01.01	2013-07-31	MB/JF	Call# 267174 Added cCustomerDocuments.GetDocumentImage() method to pull a customer document image from the E3 system
13.01.01	2013-08-05	DC	Call# 269931. Added PlanCommittedBy, PlanCommittedDT (introduced by call 209483 but never added to programmer documentation), PreBuyPlan to the SLPPDATA object.
13.01.01	2013-08-05	MB	Call# 292795 Add daily_usage_rate and delivery_interval to the Smartconnect TANKS/DELIVERY template.
13.01.01	2013-08-13	CCF	Update cUtility
13.01.01	2013-10-07	CCF	Clarified Credit Card Vault ID.
13.01.02	2013-10-29	DC	Call#296173 added field “invoice_special_handling” to the WDMS (-903) template
13.01.02	2013-11-14	CCF	Added Vault overview to cCreditCard.
13.01.02	2014-02-19	CCF	Clarification and revisions to cCreditCard..
xx.xx.xx	2014-03-25	CG	Corrected licence requirements for various ‘save’ contact information methods
14.01.01	2014-11-13	KM	Call #311035ADD Added Event & Account dates to the “DCEnergyAPI.cTransaction.AcceptmentData” structure used by cTransaction.AcceptPayment_V2. E3/DB version 12.01+
14.01.01	2014-12-01	MB	Call# 303667ADD Enhance transaction upload methods to better enforce database rules.

Revision	Date	Author	Description
			Call# 310425ADD Trim off time passed in for transaction dates-event, accounting, posting.
14.01.04	2015-03-30	KM	Call #330807ADD Enhance delivery log phone order to accept charge CC on posting flag. NOTE: *_V2 versions of the log & reschedule phone order methods have been created that accept a data structure instead of individual parameters
15.01.00	2015-03-25	DC	Changed data type for “card_veh_num” in the TRANS_MAIN definition from int to varchar
15.01.00	2015-07-03	MB	Call# 335110ADD Modified cPlanPrice.Save() method to update the log header comments.
15.01.00	2015-07-15	MB	Call# 327060ADD Smartconnect - Enhanced GetTicketInfo() API to execute it for all tanks and include multiple tickets per tank (day of week scheduling and/or route scheduling)
15.01.01	2015-09-24	KM	Call #346510ADD SmartConnect – Added new method cCustomerInfo.GetAccountsByGroupCode
15.01.02	2015-10-21	DC	Call #343810ADD – Modified GetDeliveryHistory : added “trx_unique_key” to list of returned values
15.01.03	2015-10-27	DC	Call#346507 – Added “po_number” to GetARTransactions()
15.01.03	2015-10-27	DC	Call#346508 – Added “its_qoh”, “date_last_inventoried” and “its_product” to TANKS template
15.01.03	2015-10-27	DC	Call#346509 – Added “open_amt” to GetARTransactions()
15.01.03	2015-10-27	DC	Call#346505 – Added “tank_category_2” to TANKS template
15.01.03	2015-10-27	DC	Call#346502 – Added “po_required”, “po_number” and “po_expiration” to TANKS template
15.01.03	2015-10-27	DC	Call#344254 – Added “in_tank_units” and “percent_fill” to GetDeliveryHistory() and GetARTransactions()
15.01.03	2015-10-28	DC	Call#343525 – Added “part_base_price_code”, “department_code” and “acquired” to SERVICE template
15.01.03	2015-11-02	DC	Call#346506 – Added “contract_tax_5”, “contract_tax_4”, “contract_tax_3”, “contract_tax_2”, “last_contract_bill_date”, “initial_contract_date”, “contract_tax_amount”*, “contract_price”*

Revision	Date	Author	Description
			and “contract_total_price”* to SERVICE template <i>*fields are calculated fields and not from the base table</i>
15.01.03	2015-11-09	DC	Call#347998ADD – Added “keyoff_flag” to GetARTransactions() [open_amt was added by call#346509ADD]
15.01.03	2015-11-17	KM	Call #346504ADD – Delivery Stop Code APIs
15.01.03	2015-11-17	KM	Call #346500ADD – Read/Write PO Number for Deliverys
15.01.03	2015-11-24	JF	Call #348521ADD – GetDocumentImage() note for Pegasus signature.
15.01.05	2016-02-02	MB	Call# 357189ADD Added GetARTransactionsWithNoCreditActions() method to exclude the credit actions
15.01.05	2016-02-17	KM	Call #342417ADD, added new methods to allow saving & load of attachments to the E3 Customer log.
15.01.05	2016-02-25	DC	349499ADD+349500ADD: added new methods to allow for loading and/or updating of existing logs (non system)
15.01.05	2016-03-02	CF	354656ADD: Added dont_price_meter, elec_in_tank_units, elec_pending_delivery, elec_percent_reading, elec_tank_read_date, exempt_from_mar, key_code, key_required, misc_charge_remote_invoice, sales_type, snow_drop, special_ticket_handling to TANKS template
16.01.00	2016-04-22	MB	Call# 354598ADD Add "Invoice Number" to 'GetARTransactions()'
16.01.00	2016-04-22	MB	Call# 354592ADD Added cDelivery.GetDeliveryHistoryWithCredits() to include zero/negative units
16.01.00	2016-09-07	DFC	Call# 377197ADD: Updated the Return Code+Description listing (Appendix C). Added to the list: 504,788,831,980,981,1011,1483-1499
16.01.03	2016-12-13	AJG	376388ADD: Documentation for cBestBuy.GetBestBuy web method
16.01.04	2016-12-21	MB	Call# 385975ADD Enhancemnet to provide a special price when logging a phone order; Added the ability to delete a printed ticket: cDelivery.CancelPrintedTicket()

Revision	Date	Author	Description
16.01.04	2016-12-31	MB	Doc: Changed the TANKS..last_delivery_units data type to Long Integer with no decimals
16.01.04	2017-01-04	DFC	Call #385177ADD GetAvailableSlots(), LogWorkOrder() and RescheduleWorkOrder(): The documentation was not accurate for when scheduling system is NT Calendar. Added some 'notes' and 'attentions' along with corrections.
16.01.05	2017-02-08	DFC	Call#360437ADD New method Delivery.ComputeTaxes()
16.01.05	2017-02-10	KM	Call #387881ADD New method cTransactions.GetARTransactions_V2
16.01.06	2017-03-10		Call# 392713ADD Add Budget fields to SmartConnect cCustomerInfo.GetCreditInfo() API
16.01.07	2017-03-23		Call# 392713ADD Added more Budget fields to SmartConnect cCustomerInfo.GetCreditInfo() API
17.01.00	2017-03-28	AJG	Call# 384854ADD: New PriceQuote.GetPriceCodes() API
17.01.00	2017-05-12	DC	Call#392780ADD Added the CreditAction branch to the API
16.01.09	2017-05-24	KM	Call #392797ADD – LocationStatus APIs
17.01.01	2017-07-20	DC	Call#402294ADD - Added RescheduleWorkOrderWithNotes() to cService class
17.01.01	2017-08-10	KM	Call #402294ADD - Added p_sErrMsg to cServiceRescheduleWorkOrderWithNotes() method
17.01.02	2017-10-27	MB	Call# 405453ADD Enhancements to SmartConnect related to prompt pay discounts; added 2 new web methods: 1) cTransaction.GetARTransactionsPPD() 2) cCustomerInfo.GetCreditInfoPPD()
17.01.02	2017-11-10	DFC	Call#s 410206ADD + 411977ADD Service Branch: New methods -> LogWorkOrderNTC(), RescheduleWorkOrderNTC(), LoadTechnicians() & UpdateCallInformation()
17.01.04	2017-11-17	DFC	Call#403356ADD. Enhanced cPlanPrice.Save to allow for updating an existing record. This required a modification to LoadPlans() as well which now returns more data. Additional fields TransactionKey & LastMaintenanceDate were added to the SLPPData structure.

Revision	Date	Author	Description
17.01.04	2017-11-20	DFC	Call#410206ADD: UpdateCallInformation() will now allow for updating of PreferredTechnician & will create log entry (to be in line with LogWorkOrderNTC+RescheduleWorkOrderNTC).
17.01.06	2018-03-12	DFC	Call#421375ADD: CreateAdjustmentTransferEntry(): To be used to create data entries in the E3 system that will be processed via various E3 functions/interfaces. currently only supports E3 General Interface File Creation: "Refunds - Energy to AP".
18.01.00	2018-05-30	DFC	Call#398678ADD - Enhance SmartConnect Customer Maintenance to be able to use Customer defined templates. New apis: GetTemplates () + AddOrUpdateCustomerRecord()
18.01.00	2018-05-31	AJG	Call#429679ADD: new method GetAssetDeliveryDetails()
18.01.00	2018-06-29	DFC	Call#423204ADD: Added new Contract branch to SmartConnect
18.01.01	2018-08-13	MB	Call# 430099ADD: Enhance SmartConnect Transaction load to include EFT fields (term ID, eft transaction status, eft due date, ude by date.
18.01.01	2018-08-28	DFC	Call#434756ADD: added field "total_price" to dataset returned by GetProductPrice(). This field represents the average_price+deviation regardless if price plan or not.
18.01.02	2018-11-19	MB	Call# 358785ADD Enhance E3/SmartConnect AcceptPayment to allow more than one posting code for each payment type (e.g., Check, Credit Card).
18.01.02	2018-11-27	KM	Call #407351ADD Enhance the API to allow storage of non-E3 documents in the Customer Document Viewer
18.01.05	2019-03-06	DFC	Call442471ADD: Customer Maintenance branch - Delivery/Service Customer Defined templates: make available the address parsed fields (appendix B: although dedicated to the smartconnect standard templates, highlights in yellow the parsed fields) *IMPORTANT* - when updating an existing record,if <u>any</u> parsed field is updated, <u>ALL</u> parsed fields, whether updated or not <u>must be included</u> in the request/dataset to prevent loss of data. - when using the address parsed fields, "delivery_address" or "service_address", although available should not be modified. The system will rebuild the value based on the parsed data.
18.01.09	2019-11-21	KM	Call #422772ADD: Delivery Add-ons Allow loading of Delivery AR Add-ons Allow associating Delivery Ar Add-ons with logging & rescheluding phone order. Requires E3 v18+

Introduction

Overview

The Energy SmartConnect API provides a framework for accessing and updating Energy data from other applications. It enables the user to access the following modules:

- Web user profile
- Energy account validation
- Customer information lookup
- Delivery information lookup including history
- Service information lookup including history
- Transaction history
- Payment entry
- Delivery request
- Service request
- Customer maintenance (ADD/MODIFY)
- Plan Price SLPP
- EFT maintenance
- Credit Card processing

Licensing

SmartConnect is a licensed product marketed with various features. Each feature requires a 'license'. Each license provides the ability to use one or more of the APIs (Application Programmer Interface) contained in the SmartConnect suite.

The information below describes each of the licenses available in SmartConnect and a short description of what they provide. Where appropriate, the ADD Energy E3 Client equivalent (or similar function) is listed. Please pay special attention to the pre-requisite licenses within ADD Energy E3 noted in the descriptions below.

Individual components of SmartConnect may require different licenses.

To date the following licenses apply;

- SmartConnect API Customer Look-up Business Edition*
This is the 'base' API and is required for all SmartConnect sales. This API contains the core security/login functions and the ability to retrieve customer data from the E3 database.
- SmartConnect API Delivery Ticket Business Edition
This API allows for the ability to create a Delivery Order (aka 'Phone order') for an existing E3 Customer/Delivery Location.
- SmartConnect API Payment & Credit Card Entry Business Edition
This license bundles together two separate and distinct, but related, functions:
First, this API allows for the ability to create a 'payment'/'credit' on an E3 Customer. The API creates an entry in a 'Payment Entry' batch, which is manually updated by an E3 user. The 'Payment Entry' batch is noted as a 'SmartConnect' batch.
This API can be used to create an A/R credit of any kind on the E3 Customer.

Second, it also allows for the ability to use the Credit Card Information stored in the E3 database for an E3 Customer to generate an A/R credit on the E3 Customer by charging their Credit Card. It also allows for the ability to save a new Credit Card for the Customer. This API interacts with and requires the Credit Card Gateway used by the Credit Card Processing component of E3. **The E3 Credit Card Processing license is a pre-requisite to using the Credit Card portion of this API.**

Note that this license does not interact with the E3 EFT Processing Software. See the EFT API license description that follows.

- SmartConnect API EFT Maintenance Business Edition
This API allows for the ability to use the EFT Information stored in the E3 database for an E3 customer to generate an A/R credit on the E3 customer. **The E3 EFT license is a pre-requisite to using this API.**

- SmartConnect API Service Order Business Edition
This API allows for the ability to create a Service Order for an existing E3 Customer/Service location. The Service order that is created is typically for a 'future'/'scheduled' service visit.
- SmartConnect API Customer Add & Modify Bus Edition
This API allows for the ability to create new Customers in E3 and modify existing Customers in E3. It provides similar functionality to the ADD Energy E3 Client function called: Customer Maintenance.
- SmartConnect API Phone Pops Business Edition
This API allows for the ability to integrate the Phone Switch to the E3 Customer Display and 'pop' the Customer on the screen based on their phone number using 'Caller ID'. This API requires additional 3rd party software and a compatible Phone Switch/System.
- SmartConnect API Plan Price SLPP Business Edition
This API allows for the ability to create a Plan Price record on an existing E3 Customer.
- SmartConnect API Create Log Entries Business Edition
This API allows for the ability to create a Customer Log entry on an existing E3 Customer.
- SmartConnect API Delivery Maintenance/Tank Monitor Bus. Ed.
This API allows for the ability to use the 'Tank Percentage Reading' feature in the E3 Delivery Maintenance component. The API can be used to update the Delivery forecast by using a Tank Percentage Reading coming from a Remote Tank Monitor device.
- SmartConnect API WDMS Order Creation/Maintenance

*All of the classes listed below require the API Customer Look-up Business Edition license as a minimum. Some classes may require additional license(s) as noted within the class.

The application will return "**The selected feature requires a valid license**" error message if an unlicensed component method is invoked.

Important Note: When new smart connect licenses are applied, they will be active after the data cache has been refreshed. The data cache stores the status of Smart Connect licenses and is refreshed by default every 12 hours or the value set by `cUtility.SetDataCacheRefreshTimeout()`. If you wish to have the license(s) activated sooner, you can call `cUtility.ClearCacheData()` to initialize the data cache right away. If you need further assistance, please contact our support department.

Basic Concepts

SmartConnect is a collection of web services that allow external applications to programmatically exchange data with the ADD Energy/E3 system. It uses XML documents to move data into and out of the E3 database(s), this allows SmartConnect to provide back office integration to any application capable of creating or consuming these XML documents.

The SmartConnect product offers support for web service communication based on the WS-I basic profile 1.1 (WS-BP 1.1) specification, this includes standards such SOAP 1.1, WSDL 1.1, and message security 1.0. It uses HTTP for the transport and offers interoperability across heterogeneous systems.

The SmartConnect web services provide the ability to search and locate energy accounts across multiple Energy databases. It uses an advanced data cache mechanism to minimize round-trips to database servers.

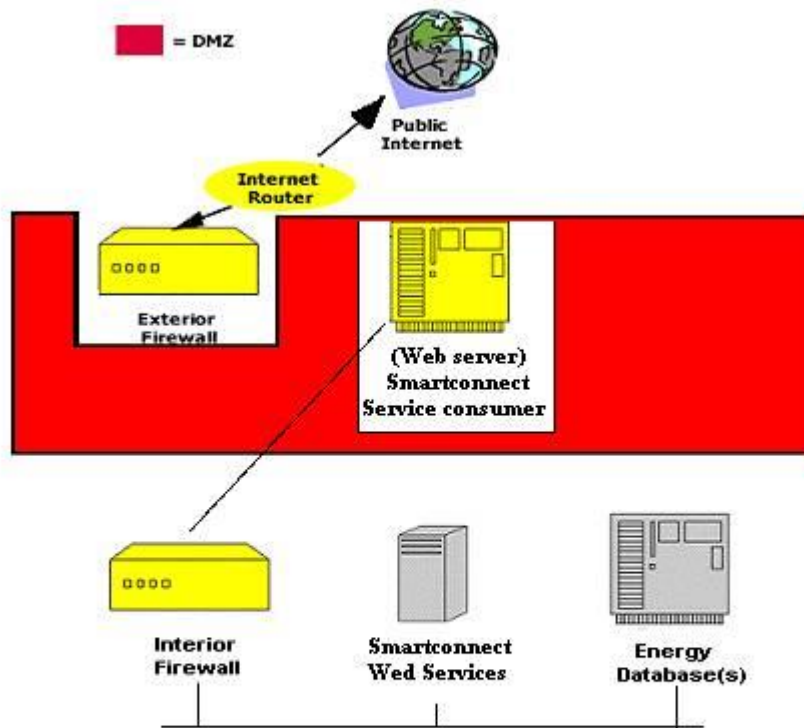
Once the Energy account is located using a 10-digit telephone, and optionally an account number, each subsequent attempt to connect to an Energy database requires a company number as well as a valid account number.

Security

Several layers of security are currently implemented before an end-user can establish a connection to Energy databases:

- **Connect Authentication:** Energy SmartConnect uses the authentication services provided by Microsoft Windows security to enable the identity verification and make sure the client identity we're checking is authentic.
- **Energy security layer:** an ID will be created in Energy security for each calling application. Any client application that doesn't have a valid caller ID will not have access to the Energy SmartConnect
- **Security by Company:** Energy accounts will be accessed based on a list of companies to which each calling application has access
- **Security by user profile:** the status for a user profile must be "Active" in order to log on. The user profile password will be encrypted within the database

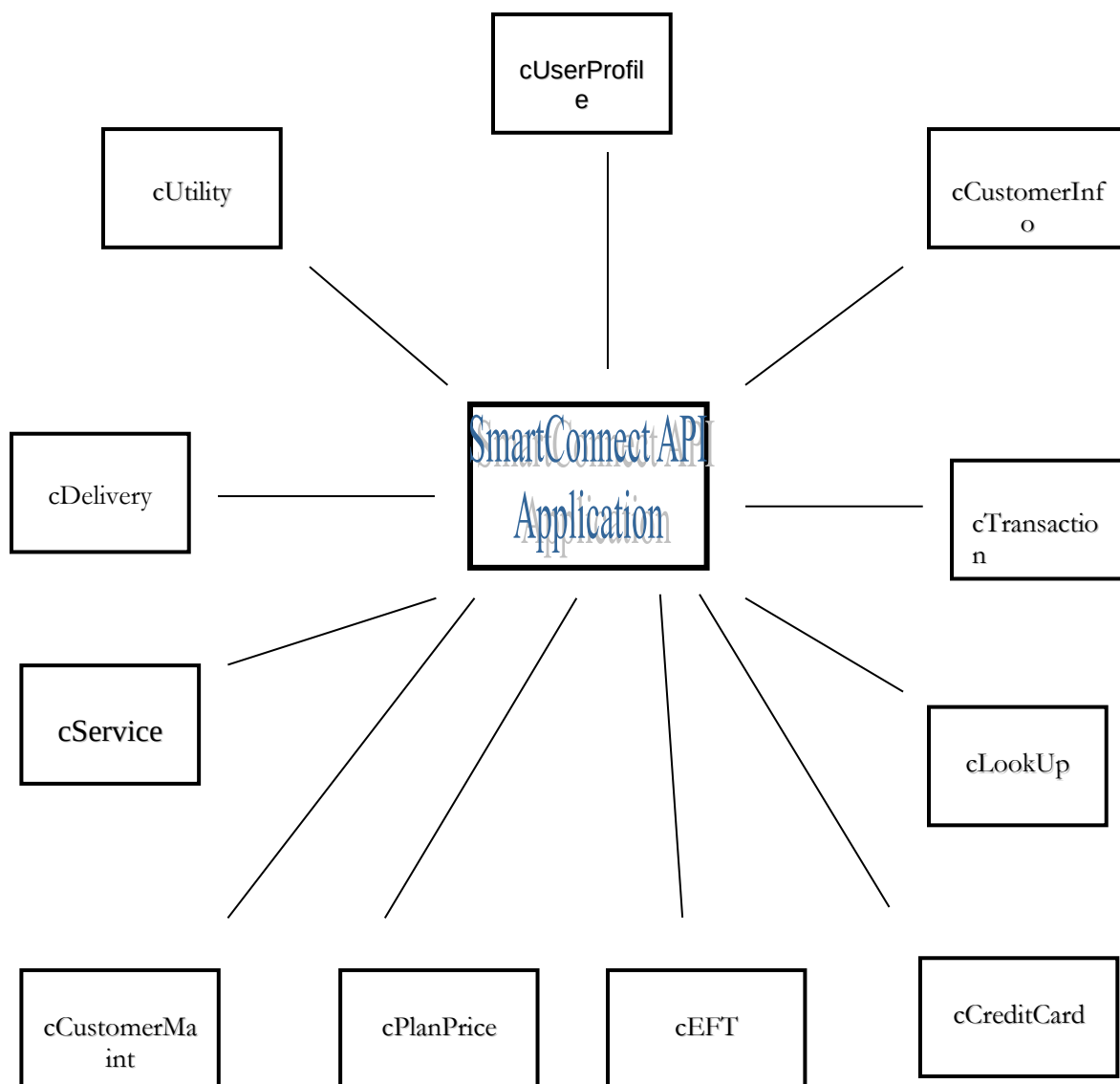
The following diagram outlines one method as to how to configure a DMZ to support a secured website:



ENERGY SmartConnect Main Classes

The current Energy SmartConnect API is based on the main classes displayed below. A detailed interface description for each class is provided within this document

Main Classes



CUserProfile Class

Overview

This section describes the methods that enable the creation and maintenance of user profiles. The following user profile methods are covered in this section: **(E-Mail address is case sensitive)**

- CreateUser(): creates a new user profile
- ModifyUserInfo(): allows the user to change the user profile (status, nickname, password hint, hint answer)
- ChangePassword(): allows the user to change the password
- DeleteUser(): deletes user profile
- GetUserInfo(): retrieves user profile information
- LinkAccount(): links additional accounts
- UnlinkAccounts(): unlinks a specific Energy account
- ListAccounts(): retrieves a list of linked accounts
- Login(): allows the user to login to the system using an email address and password
- GetPassword(): retrieves the password entered for a user profile using the password hint and hint answer
- ChangeDefaultAccount(): this method allows the web user to select the default account linked to his/her profile

Special Characters

As of SmartConnect 14.1.5 all characters are accepted in the password field. However, it should be noted that when working in XML there are characters that have special meaning which can cause the xml parser to misunderstand the resulting data and can therefore cause issues. This needs to be dealt with programmatically.

eg: Password = password!@#%^*()<

The "<" character in this password will be a problem and will cause parsing errors

```
<add:p_sPassword> password!@#%^*()<</add:p_sPassword>
```

Solution: Wrap the text within **<![CDATA[** and **]]>**

```
<add:p_sPassword><![CDATA[password!@#%^*()<]]></add:p_sPassword>
```

cUserProfile.CreateUser()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Creates a new user profile.

```
Public Function CreateUser(ByVal p_sEmailAddress As String, _
    ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sPassword As String, _
    ByVal p_sName As String, _
    ByVal p_sPasswordHint As String, _
    ByVal p_sHintAnswer As String, _
    ByVal p_lPreferences As Integer, _
    ByVal p_sSysUser As String) As Integer
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'    p_iCompanyID -- Energy Company ID -- Short integer (1-32767)
'    p_lAccountNum -- Energy account number without the check digit -- Long Integer
'    p_sPassword -- User password -- Char(6-20)
'    p_sName -- Web user nickname -- Char(50)
'    p_sPasswordHint - Password hint - Char(100)
'    p_HintAnswer - Hint answer - Char(100)
'    p_lPreferences -- a bitwise combination of preferences:
'        Masking value = 0x0001 -- Send information about my account to my e-mail address
'        Masking value = 0x0002 -- Send promotional information that I may find useful to my e-mail address
'    p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: Return Type Integer -- 0 If Successful: <> 0 Error Number
'
```

Return code	Description
96	Unknown Error occurred.
109	A User Profile already exists for this email address.

Comments

The account number must be validated before calling this method, `cCustomerInfo.ValidateAccountWithInfo()` method should be used to validate the account and retrieve the company id and account number used as input parameters to this method.

XML Example

The following example shows how to create a new user profile once the energy account has been validated:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CreateUser>
      <add:p_sEmailAddress>tests1800@addsys.com</add:p_sEmailAddress>
      <add:p_iCompanyID>1800</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_sPassword>adcadc</add:p_sPassword>
      <add:p_sName>Tests 1800</add:p_sName>
      <add:p_sPasswordHint>adc_Hint</add:p_sPasswordHint>
      <add:p_sHintAnswer>adc_ans</add:p_sHintAnswer>
      <add:p_lPreferences>3</add:p_lPreferences>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:CreateUser>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <CreateUserResponse xmlns="http://www.addsys.com/">
      <CreateUserResult>0</CreateUserResult>
    </CreateUserResponse>
  </soap:Body>
</soap:Envelope>
```

cUserProfile.ModifyUserInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Use this method to modify user profile information

```
Public Function ModifyUserInfo(ByVal p_sEmailAddress As String, _
    ByVal p_sName As String, _
    ByVal p_sPasswordHint As String, _
    ByVal p_sHintAnswer As String, _
    ByVal p_lPreferences As Integer, _
    ByVal p_sStatus As String, _
    ByVal p_sSysUser As String) As Integer
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'         p_sName -- Web user nickname -- Char(50)
'         p_sPasswordHint - Password hint - Char(100)
'         p_sHintAnswer - Hint answer - Char(100)
'         p_lPreferences -- a bitwise combination of preferences:
'             Masking value = 0x0001 -- Send information about my account to my e-mail address
'             Masking value = 0x0002 -- Send promotional information that I may find useful to my e-mail address
'         p_sStatus - user profile status -- Char(1) - "A"=Active, "I"=Inactive
'         p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

'Returns: Return Type Integer -- 0 If Successful : <> 0 Error Number

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to modify an existing user profile information:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ModifyUserInfo>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sName>Test 1800</add:p_sName>
      <add:p_sPasswordHint>addadd</add:p_sPasswordHint>
      <add:p_sHintAnswer>adc_hint</add:p_sHintAnswer>
      <add:p_lPreferences>0</add:p_lPreferences>
      <add:p_sStatus>A</add:p_sStatus>
```

```
<add:p_sSysUser>WebUser</add:p_sSysUser>
</add:ModifyUserInfo>
</soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ModifyUserInfoResponse xmlns="http://www.addsys.com/">
      <ModifyUserInfoResult>0</ModifyUserInfoResult>
    </ModifyUserInfoResponse>
  </soap:Body>
</soap:Envelope>
```

cUserProfile.ChangeEmailAddress ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Changes the Email Address for an existing Web user

Public Function **ChangeEmailAddress** (ByVal p_sEmailAddress As String, _
ByVal p_sNewEmailAddress As String, _
ByVal p_sSysUser As String) As Integer

Input Parameters

'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
' p_sNewEmailAddress - new Email Address -- Char(1-50)
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: Return Type Integer -- 0 If Successful: <> 0 Error Number
,

Return code	Description
96	Unknown Error occurred.
109	A User Profile already exists for this email address.
1055	No changes.

XML Example

The following example shows how to modify an existing user profile Email Address:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ChangeEmailAddress>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sNewEmailAddress>tester1800@addsys.com</add:p_sNewEmailAddress>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:ChangeEmailAddress>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ChangeEmailAddressResponse xmlns="http://www.addsys.com/">
      <ChangeEmailAddressResult>0</ChangeEmailAddressResult>
    </ChangeEmailAddressResponse>
  </soap:Body>
</soap:Envelope>
```

cUserProfile.ChangePassword()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Changes the password for an existing Web user

Public Function **ChangePassword**(ByVal p_sEmailAddress As String, _
ByVal p_sNewPassword As String, _
ByVal p_sSysUser As String) As Integer

Input Parameters

'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
' p_sNewPassword - new password - Char(6-20)
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: Return Type Integer -- 0 If Successful: <> 0 Error Number
,

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to call this method in order to change the password:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ChangePassword>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sNewPassword>a~!@#z</add:p_sNewPassword>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:ChangePassword>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ChangePasswordResponse xmlns="http://www.addsys.com/">
```

```
    <ChangePasswordResult>0</ChangePasswordResult>
  </ChangePasswordResponse>
</soap:Body>
</soap:Envelope>
```

cUserProfile.DeleteUser()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Deletes an existing user profile from the system.

Public Function **DeleteUser** (ByVal p_sEmailAddress As String, _
ByVal p_sSysUser As String) As Integer

Input Parameters

'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: Return Type Integer -- 0 If Successful: <> 0 Error Number

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to delete a web user profile:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add>DeleteUser>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add>DeleteUser>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <DeleteUserResponse xmlns="http://www.addsys.com/">
      <DeleteUserResult>0</DeleteUserResult>
    </DeleteUserResponse>
  </soap:Body>
</soap:Envelope>
```



```
</DeleteUserResponse>  
</soap:Body>  
</soap:Envelope>
```

cUserProfile.GetUserInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns user profile information

Public Function **GetUserInfo**(ByVal p_sEmailAddress As String, _
ByVal p_sSysUser As String, _
ByRef p_lErrorCode As Integer) As System.Data.DataSet

Input Parameters

'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

NOTE: If the eMail address is not found an empty data set is returned with an error code of zero.

Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number

' Return Type DataSet.Tables(0).Rows (0)Web nickname - ("name") - Char(50)
' (1)Password hint - ("password_hint") - Char(100)
' (2)Hint Answer - ("hint_answer") - Char(100)
' (3)Bitwise combination of preferences ("preferences") -- Long integer
' (4)Default Company ID ("default_company_id") -- Short integer
' (5)Default Account number ("default_account_num") -- Long integer
' (6)Date created ("created_dt") -- Date && Time
' (7)Last access Date & time ("last_access_dt") -- Date && Time
'

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to retrieve web user profile information:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetUserInfo>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetUserInfo>
  </soap:Body>
</soap:Envelope>
```

```

    </add:GetUserInfo>
  </soap:Body>
</soap:Envelope>

```

Response:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetUserInfoResponse xmlns="http://www.addsys.com/">
      <GetUserInfoResult>
        <xs:schema id="DSUserInfo"
targetNamespace="http://www.addsys.com/EnergyAPI/UserProfile/DSUserInfo.xsd"
attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/UserProfile/DSUserInfo.xsd"
xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSUserInfo.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-
com:xml-msdata">
          <xs:element name="DSUserInfo" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="UserInfo">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="password_hint" type="xs:string" minOccurs="0"/>
                      <xs:element name="hint_answer" type="xs:string" minOccurs="0"/>
                      <xs:element name="preferences" type="xs:int" minOccurs="0"/>
                      <xs:element name="default_company_id" type="xs:short"
minOccurs="0"/>
                      <xs:element name="default_account_num" type="xs:int" minOccurs="0"/>
                      <xs:element name="created_dt" type="xs:string" minOccurs="0"/>
                      <xs:element name="last_access_dt" type="xs:dateTime" minOccurs="0"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:schema>
        <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
          <DSUserInfo
xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSUserInfo.xsd">
            <UserInfo diffgr:id="UserInfo1" msdata:rowOrder="0">
              <name>Test 1800</name>

```

```
<password_hint>addadd</password_hint>
<hint_answer>adc_hint</hint_answer>
<preferences>0</preferences>
<default_company_id>0</default_company_id>
<default_account_num>0</default_account_num>
<created_dt>6/4/2018 11:08:09 AM</created_dt>
<last_access_dt>2019-04-23T14:35:48.863-04:00</last_access_dt>
</UserInfo>
</DSUserInfo>
</diffgr:diffgram>
</GetUserInfoResult>
<p_lErrorCode>0</p_lErrorCode>
</GetUserInfoResponse>
</soap:Body>
</soap:Envelope>
```

cUserProfile.LinkAccount ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Use this method to link additional accounts to an existing Web user profile

```
Public Function LinkAccount(ByVal p_sEmailAddress As String, _
    ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String) As Integer
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'          p_iCompanyID -- Energy Company ID -- Short integer (1-32767)
'          p_lAccountNum -- Energy account number without the check digit -- Long Integer
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

'Returns: Return Type Integer -- 0 If Successful : <> 0 Error Number

Return code	Description
96	Unknown Error occurred.
110	A link already exists to this company/account number.

XML Example

The following example shows how to link an additional energy account

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LinkAccount>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_iCompanyID>1800</add:p_iCompanyID>
      <add:p_lAccountNum>1003</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:LinkAccount>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <LinkAccountResponse xmlns="http://www.addsys.com/">
      <LinkAccountResult>0</LinkAccountResult>
    </LinkAccountResponse>
  </soap:Body>
</soap:Envelope>
```

cUserProfile.UnLinkAccount ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Use this method to unlink an Energy account from an existing Web user profile

```
Public Function UnlinkAccount(ByVal p_sEmailAddress As String, _  
    ByVal p_iCompanyID As Short, _  
    ByVal p_lAccountNum As Integer, _  
    ByVal p_sSysUser As String) As Integer
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)  
'    p_iCompanyID -- Energy Company ID -- Short integer (1-32767)  
'    p_lAccountNum -- Energy account number without the check digit -- Long Integer  
'    p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

'Returns: Return Type Integer -- 0 If Successful : <> 0 Error Number

Return code	Description
96	Unknown Error occurred.
229	Unable to unlink this company/account number combination.

XML Example

The following example shows how to unlink an account

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:UnLinkAccount>  
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>  
      <add:p_iCompanyID>1800</add:p_iCompanyID>  
      <add:p_lAccountNum>1003</add:p_lAccountNum>  
      <add:p_sSysUser>WebUser</add:p_sSysUser>  
    </add:UnLinkAccount>  
  </soap:Body>  
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <UnLinkAccountResponse xmlns="http://www.addsys.com/">
      <UnLinkAccountResult>0</UnLinkAccountResult>
    </UnLinkAccountResponse>
  </soap:Body>
</soap:Envelope>
```


cUserProfile.ListAccounts ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of accounts that were previously linked to a web user profile

```
Public Function ListAccounts(ByVal p_sEmailAddress As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Company ID ("company_id") -- Short integer
'          (1)Account number without the check digit ("account_num") -- Long integer
'          (2)Account number with the check digit ("full_account") -- Char(10)
'          (3)Account name ("name") -- Char(40)
'          (4)Tank count ("tank_count") -- Short integer
'          (5)Service Location count ("service_count") -- Short integer
'          (6)Wdms location count ("location_count") -- Short integer
```

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to retrieve a collection of linked accounts

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add>ListAccounts>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add>ListAccounts>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ListAccountsResponse xmlns="http://www.addsys.com/">
      <ListAccountsResult>
        <xs:schema id="DSListAccounts"
targetNamespace="http://www.addsys.com/EnergyAPI/UserProfile/DSListAccounts.xsd"
attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/UserProfile/DSListAccounts.xsd"
xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSListAccounts.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
          <xs:element name="DSListAccounts" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="ListAccounts">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="company_id" type="xs:short" minOccurs="0"/>
                      <xs:element name="account_num" type="xs:int" minOccurs="0"/>
                      <xs:element name="full_account" type="xs:string" minOccurs="0"/>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="tank_count" type="xs:short" minOccurs="0"/>
                      <xs:element name="service_count" type="xs:short" minOccurs="0"/>
                      <xs:element name="location_count" type="xs:short" minOccurs="0"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:complexType>
          </xs:element>
          <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
            <DSListAccounts
xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSListAccounts.xsd">
              <ListAccounts diffgr:id="ListAccounts1" msdata:rowOrder="0">
                <company_id>1800</company_id>
                <account_num>1003</account_num>
                <full_account>10033</full_account>
                <name>GORDON SMITH</name>
                <tank_count>1</tank_count>
                <service_count>1</service_count>
                <location_count>0</location_count>
              </ListAccounts>
```

```
        </DSListAccounts>
      </diffgr:diffgram>
    </ListAccountsResult>
    <p_lErrorCode>0</p_lErrorCode>
  </ListAccountsResponse>
</soap:Body>
</soap:Envelope>
```

cUserProfile.Login()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method allows the user to login to the system using an email address and password

```
Public Function Login(ByVal p_sEmailAddress As String, _
    ByVal p_sPassword As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'          p_sPassword -- Web user password -- Char(6-20)
'          p_sSysUser - Calling program (IVRUser/WebUser)
'
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Web user nickname ("name") -- Char(50)
'          (1)Last access date && time ("last_access_dt") -- Date && Time
'
```

Return code	Description
96	Unknown Error occurred.
112	Login failed. Invalid login credentials supplied.

XML Example

The following example shows how to login to the system using an email address and password

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:Login>
      <add:p_sEmailAddress>tests1800@addsys.com</add:p_sEmailAddress>
      <add:p_sPassword>adcadc</add:p_sPassword>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:Login>
  </soap:Body>
</soap:Envelope>
```

Response:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <LoginResponse xmlns="http://www.addsys.com/">
      <LoginResult>
        <xs:schema id="DSLogin"
targetNamespace="http://www.addsys.com/EnergyAPI/UserProfile/DSLogin.xsd"
attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/UserProfile/DSLogin.xsd"
xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSLogin.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
          <xs:element name="DSLogin" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="Login">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="last_access_dt" type="xs:dateTime" minOccurs="0"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:schema>
        <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
          <DSLogin xmlns="http://www.addsys.com/EnergyAPI/UserProfile/DSLogin.xsd">
            <Login diffgr:id="Login1" msdata:rowOrder="0">
              <name>Tests 1800</name>
              <last_access_dt>1960-12-31T00:00:00-05:00</last_access_dt>
            </Login>
          </DSLogin>
        </diffgr:diffgram>
      </LoginResult>
      <p_lErrorCode>0</p_lErrorCode>
    </LoginResponse>
  </soap:Body>
</soap:Envelope>

```

cUserProfile.GetPassword()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a password entered for a user profile

```
Public Function GetPassword(ByVal p_sEmailAddress As String, _
    ByVal p_sPasswordHint As String, _
    ByVal p_sHintAnswer As String, _
    ByVal p_sSysUser As String, _
    ByVal p_lErrorCode As Integer) As String
```

Input Parameters

```
'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
'          p_sPasswordHint -- Password hint entered when creating a user profile -- Char(100)
'          p_sHintAnswer -- Hint answer -- Char(100)
'          p_sSysUser - Calling program (IVRUser/WebUser)
'
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type String -- Password -- Char(20) if found -- Null String if not found
'
```

Return code	Description
96	Unknown Error occurred.
111	The answer to the Password Hint did not match. Unable to retrieve the password.

XML Example

The following example shows how to retrieve a web user password

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetPassword>
      <add:p_sEmailAddress>tests1800@addsys.com</add:p_sEmailAddress>
      <add:p_sPasswordHint>adc_Hint</add:p_sPasswordHint>
      <add:p_sHintAnswer>adc_ans</add:p_sHintAnswer>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:GetPassword>
  </soap:Body>
</soap:Envelope>
```

```
<add:p_lErrorCode>0</add:p_lErrorCode>
</add:GetPassword>
</soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetPasswordResponse xmlns="http://www.addsys.com/">
      <GetPasswordResult>adcadc</GetPasswordResult>
      <p_lErrorCode>0</p_lErrorCode>
    </GetPasswordResponse>
  </soap:Body>
</soap:Envelope>
```

cUserProfile.ChangeDefaultAccount()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Use this method to Change the default energy account by selecting the default Company ID/Account number.

Public Function **ChangeDefaultAccount**(ByVal p_sEmailAddress As String, _
ByVal p_iDefaultCompanyID As Short, _
ByVal p_lDefaultAccountNum As Integer, _
ByVal p_sSysUser As String) As Integer

Input Parameters

'Accepts: p_sEmailAddress -- Email Address -- Char(1-50)
' p_iDefaultCompanyID -- Default Company ID -- Short integer (1-32767)
' p_lDefaultAccountNum -- Default account number without the check digit -- Long Integer
' p_sSysUser - Calling program (IVRUser/WebUser)
,

Return Values

'Returns: Return Type Integer -- 0 If Successful : <> 0 Error Number.

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to change the default account number

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ChangeDefaultAccount>
      <add:p_sEmailAddress>test1800@addsys.com</add:p_sEmailAddress>
      <add:p_iDefaultCompanyID>1800</add:p_iDefaultCompanyID>
      <add:p_lDefaultAccountNum>1</add:p_lDefaultAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:ChangeDefaultAccount>
  </soap:Body>
</soap:Envelope>
```

Response:


```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ChangeDefaultAccountResponse xmlns="http://www.addsys.com/">
      <ChangeDefaultAccountResult>0</ChangeDefaultAccountResult>
    </ChangeDefaultAccountResponse>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo Class

Overview

This section describes the methods that enable the programmer to validate accounts referenced by the web/phone users. Other methods are provided to retrieve the customer information. The following methods are covered in this section:

- **ValidateAccount():** Validates an energy account
- **ValidateAccountWithInfo():** Validates an energy account and returns account related information
- **GetCustomerGeneralInfo():** retrieves account general information
- **GetCreditInfo():** retrieves credit information for a given account number
- **GetGeneralText():** retrieves account general text
- **GetDeliveryLocations():** retrieves list of delivery locations
- **GetServiceLocations():** retrieves list of service locations
- **GetWdmsLocations():** retrieves list of WDMS locations
- **GetMeters():** retrieves list of meter records
- **GetItsProducts():** retrieves list of ITS Products for a given account number and WDMS location number

Data Classes and Datasets

Class AccountsByGroupCode Definition:

Datatype/column name	Description
int AccountNum	The E3 account number (without check digit)
string FullAccount	E3 account number (with check digit)
string CustomerName	Name on the customer's account
int GroupCode1	Value for Group code 1
int GroupCode2	Valud for Group code 2

cCustomerInfo.ValidateAccount()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method is used to validate an account by entering a 10-digit telephone number. It returns the Energy account number and company ID if a unique match is found.

```
Public Function ValidateAccount(ByVal p_sAccountIn As String, _  
    ByVal p_sSysUser As String, _  
    ByRef p_sFullAccount As String, _  
    ByRef p_lAccountNum As Integer, _  
    ByRef p_iCompanyID As Short) As Integer
```

Input Parameters

'Accepts: p_sAccountIn : 10-digit telephone number
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: p_sFullAccount (ByRef) -- The Full Account Number Including Check Digit (For Display Purposes)
' p_lAccountNum (ByRef) -- The Account Number Without Check Digit (To Be Used For Querying Account Info,...)
' p_iCompanyID (ByRef) -- The Energy Company ID For Use With Multiple Energy Companies
'
' Return Type Integer -- 0 If Successful : 103 if NO account is found, 144 if more than 1 account if found
'

Return code	Description
102	Invalid database connection requested.
103	Unable to locate a unique account number. No account was found.
144	Unable to locate a unique account number. More than 1 account was found.

See Also

ValidateAccountWithInfo() method

XML Example

The following example shows to validate an account using a 10-digit telephone number:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>
```

```
<soap:Body>
  <add:ValidateAccount>
    <!--Optional:-->
    <add:p_sAccountIn>9166228768</add:p_sAccountIn>
    <!--Optional:-->
    <add:p_sSysUser>WebUser</add:p_sSysUser>
    <!--Optional:-->
    <add:p_sFullAccount></add:p_sFullAccount>
    <add:p_lAccountNum>0</add:p_lAccountNum>
    <add:p_iCompanyID>0</add:p_iCompanyID>
  </add:ValidateAccount>
</soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ValidateAccountResponse xmlns="http://www.addsys.com/">
      <ValidateAccountResult>0</ValidateAccountResult>
      <p_sFullAccount>3690</p_sFullAccount>
      <p_lAccountNum>369</p_lAccountNum>
      <p_iCompanyID>1800</p_iCompanyID>
    </ValidateAccountResponse>
  </soap:Body>
</soap:Envelope>
```

Comment

This method can only be used for unique phone numbers. It will search for the phone number across all the databases it has access to. It will return an error if it does not find the phone number or if it finds more than one account with the same phone number.

cCustomerInfo.ValidateAccountWithInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method is used to validate an account by entering a 10-digit telephone number, and optionally, an account number with the check digit. If a unique match is found, the method will return some customer basic information such as the name, delivery hold flag, terminated flag, number of delivery locations, etc.

```
Public Function ValidateAccountWithInfo(ByVal p_sTelephoneNo As String, _
    ByVal p_sFullAccount As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_sTelephoneNo-> 10-digit telephone number
'          p_sFullAccount-> Account number with the check digit
'          p_sSysUser - Calling program (IVRUser/WebUser)
'
```

Return Values

```
'Returns: p_lErrorCode -- 0 If Successful : 103 if no account is found, 144 if more than 1 account is found
'
'          Return Type DataSet.Tables(0).Rows (0)Company ID ("company_id") -- Short integer
'          (1)Account # without the check digit ("account_num") -- Long integer
'          (2)Full account # with the check digit ("full_account") -- char(10)
'          (3)Customer name ("name") -- Char(40)
'          (4)Total number of tanks ("tank_count") -- Short integer
'          (5)Total number of service locations ("service_count") -- Short integer
'          (6)Total number of WDMS locations ("location_count") -- Short integer
'          (7)Delivery hold ("delivery_hold") -- Char(1) -- Y/N
'          (8)Service hold ("service_hold") -- Char(1) -- Y/N
'          (9)Terminated account ("terminated") -- Char(1) -- Y/N
'          (10)Association hold ("association_hold") -- Char(1) -- Y/N
'          (11)Customer type ("type") -- Short integer
'
```

Return code	Description
102	Invalid database connection requested.
103	Unable to locate a unique account number. No account was found.
144	Unable to locate a unique account number. More than 1 account was found.

XML Example

The following example shows to validate an account using a 10-digit telephone number:

Request:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ValidateAccountWithInfo>
      <add:p_sTelephoneNo>9166228768</add:p_sTelephoneNo>
      <add:p_sFullAccount>3690</add:p_sFullAccount>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:ValidateAccountWithInfo>
  </soap:Body>
</soap:Envelope>

```

Response:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <ValidateAccountWithInfoResponse xmlns="http://www.addsys.com/">
      <ValidateAccountWithInfoResult>
        <xs:schema id="DSValidateAccountWithInfo"
targetNamespace="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithI
nfo.xsd" attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.x
sd"
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-
com:xml-msdata">
          <xs:element name="DSValidateAccountWithInfo" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="ValidateAccountWithInfo">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="company_id" type="xs:int" minOccurs="0"/>
                      <xs:element name="account_num" type="xs:int" minOccurs="0"/>
                      <xs:element name="full_account" type="xs:string" minOccurs="0"/>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="tank_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="service_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="location_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="delivery_hold" type="xs:string" minOccurs="0"/>
                      <xs:element name="service_hold" type="xs:string" minOccurs="0"/>
                      <xs:element name="terminated" type="xs:string" minOccurs="0"/>

```

```

        <xs:element name="association_hold" type="xs:string" minOccurs="0"/>
        <xs:element name="type" type="xs:int" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
</xs:schema>
<diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
    <DSValidateAccountWithInfo
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.xsd">
        <ValidateAccountWithInfo diffgr:id="ValidateAccountWithInfo1"
msdata:rowOrder="0">
            <company_id>1800</company_id>
            <account_num>369</account_num>
            <full_account>3690</full_account>
            <name>ROBERT GRIND</name>
            <tank_count>2</tank_count>
            <service_count>1</service_count>
            <location_count>0</location_count>
            <delivery_hold>Y</delivery_hold>
            <service_hold>N</service_hold>
            <terminated>N</terminated>
            <association_hold>N</association_hold>
            <type>1</type>
        </ValidateAccountWithInfo>
    </DSValidateAccountWithInfo>
</diffgr:diffgram>
</ValidateAccountWithInfoResult>
<p_lErrorCode>0</p_lErrorCode>
</ValidateAccountWithInfoResponse>
</soap:Body>
</soap:Envelope>

```

Comment

ValidateAccountWithInfo() method should be used to validate an account and retrieve account basic information to be displayed e.g. customer name and/or make some decisions based on the information returned by this method e.g. delivery hold, terminated, delivery location count, etc.

This method can only be used for unique phone numbers. It will search for the phone number across all the databases it has access to. It will return an error if it does not find the phone number or if it finds more than one account with the same phone number.

cCustomerInfo.GetValidAccountList()

- License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method is used to get the list of customer numbers that have identical telephone numbers.

```
Public Function GetValidAccountList(ByVal p_sTelephoneNo As String, _
    ByVal p_sFullAccount As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_sTelephoneNo-> 10-digit telephone number
'          p_sFullAccount-> Account number with the check digit
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode -- 0 If Successful : 103 if no account is found.
'
'          Return Type DataSet.Tables(0).Rows (0)Company ID ("company_id") -- Short integer
'          (1)Account # without the check digit ("account_num") -- Long integer
'          (2)Full account # with the check digit ("full_account") -- char(10)
'          (3)Customer name ("name") -- Char(40)
'          (4)Total number of tanks ("tank_count") -- Short integer
'          (5)Total number of service locations ("service_count") -- Short integer
'          (6)Total number of WDMS locations ("location_count") -- Short integer
'          (7)Delivery hold ("delivery_hold") -- Char(1) -- Y/N
'          (8)Service hold ("service_hold") -- Char(1) -- Y/N
'          (9)Terminated account ("terminated") -- Char(1) -- Y/N
'          (10)Association hold ("association_hold") -- Char(1) -- Y/N
'          (11)Customer type ("type") -- Short integer
```

Return code	Description
102	Invalid database connection requested.
103	Unable to locate a unique account number. No account was found.

XML Example

The following example shows to validate an account using a 10-digit telephone number:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
```

```

<soap:Body>
  <add:GetValidAccountList>
    <add:p_sTelephoneNo>5141234567</add:p_sTelephoneNo>
    <add:p_sFullAccount></add:p_sFullAccount>
    <add:p_sSysUser>WebUser</add:p_sSysUser>
    <add:p_lErrorCode>0</add:p_lErrorCode>
  </add:GetValidAccountList>
</soap:Body>
</soap:Envelope>

```

Response:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetValidAccountListResponse xmlns="http://www.addsys.com/">
      <GetValidAccountListResult>
        <xs:schema id="DSValidateAccountWithInfo"
targetNamespace="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithI
nfo.xsd" attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.x
sd"
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-
com:xml-msdata">
          <xs:element name="DSValidateAccountWithInfo" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="ValidateAccountWithInfo">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="company_id" type="xs:int" minOccurs="0"/>
                      <xs:element name="account_num" type="xs:int" minOccurs="0"/>
                      <xs:element name="full_account" type="xs:string" minOccurs="0"/>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="tank_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="service_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="location_count" type="xs:int" minOccurs="0"/>
                      <xs:element name="delivery_hold" type="xs:string" minOccurs="0"/>
                      <xs:element name="service_hold" type="xs:string" minOccurs="0"/>
                      <xs:element name="terminated" type="xs:string" minOccurs="0"/>
                      <xs:element name="association_hold" type="xs:string" minOccurs="0"/>
                      <xs:element name="type" type="xs:int" minOccurs="0"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:element>
          </xs:schema>
        </GetValidAccountListResult>
      </GetValidAccountListResponse>
    </soap:Body>
  </soap:Envelope>

```

```

        </xs:choice>
    </xs:complexType>
</xs:element>
</xs:schema>
<diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
    <DSValidateAccountWithInfo
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSValidateAccountWithInfo.xsd">
        <ValidateAccountWithInfo diffgr:id="ValidateAccountWithInfo1"
msdata:rowOrder="0">
            <company_id>1800</company_id>
            <account_num>369</account_num>
            <full_account>3690</full_account>
            <name>ROBERT GRIND</name>
            <tank_count>2</tank_count>
            <service_count>1</service_count>
            <location_count>0</location_count>
            <delivery_hold>Y</delivery_hold>
            <service_hold>N</service_hold>
            <terminated>N</terminated>
            <association_hold>N</association_hold>
            <type>1</type>
        </ValidateAccountWithInfo>
    </DSValidateAccountWithInfo>
</diffgr:diffgram>
</GetValidAccountListResult>
<p_lErrorCode>0</p_lErrorCode>
</GetValidAccountListResponse>
</soap:Body>
</soap:Envelope>

```

Comment

GetValidAccountList() method should be used to retrieve the full list of accounts that match on a phone number. The following customer fields are returned to the calling program: company_id, account_num, full_account, name, tank_count, service_count, location_count, delivery_hold, service_hold, terminated, association_hold, type.

This method can return one or more accounts. It will return all the accounts it can find with that phone number. For example, if you have two accounts with the same phone number. Both accounts will be returned by this method.

cCustomerInfo.GetCustomerGeneralInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves general information for a given energy account number/company ID

```
Public Function GetCustomerGeneralInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
```

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
```

```
'          Return Type DataSet.Tables(0).Rows (0)Account number with the check digit ("full_account") - Char(10)
'          (1)customer name ("name") - Char(40)
'          (2)Title ("title") - Char(4)
'          (3)First name ("first_name") - Char(40)
'          (4)Middle initial ("middle_initial") - Char(1)
'          (5)Last name ("last_name") - Char(40)
'          (6)Name suffix ("name_suffix") - Char(3)
'          (7)Street 1 ("street1") - Char(40)
'          (8)Street 2 ("street2") - Char(40)
'          (9)Town ("town") - Char(40)
'          (10)State ("state") - Char(2)
'          (11)City ("city") - Char(38)
'          (12)Zip/Postal code ("postal_code") - Char(9)
'          (13)Country ("country") - Short integer
'          (14)division ("division") - Short integer
'          (15)Tank count ("tank_count") - Short integer
'          (16)Service count ("service_count") - Short integer
'          (17)Wdms Location count ("location_count") - Short integer
'          (18)telephone number ("telephone") - Char(10)
'          (19)Delivery hold ("delivery_hold") -- Char(1) -- Y/N
'          (20)Service hold ("service_hold") -- Char(1) -- Y/N
'          (21)Terminated account ("terminated") -- Char(1) -- Y/N
'          (22)Association hold ("association_hold") -- Char(1) -- Y/N
'          (23)Priority Account flag ("priority_account") -- Char(1) -- Y/N
'          (24)Must charge credit card ("must_charge_cc") -- Char(1) -- Y/N
```

Return code	Description
96	Unknown Error Occurred.

XML Example

The following example shows how to retrieve the account general information:

Request:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCustomerGeneralInfo>
      <add:p_iCompanyID>1800</add:p_iCompanyID>
      <add:p_lAccountNum>3690</add:p_lAccountNum>
      <add:p_sSysUser>webuseR</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetCustomerGeneralInfo>
  </soap:Body>
</soap:Envelope>
```

Response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetCustomerGeneralInfoResponse xmlns="http://www.addsys.com/">
      <GetCustomerGeneralInfoResult>
        <xs:schema id="DSGetGeneralInfo"
targetNamespace="http://www.addsys.com/EnergyAPI/CustomerInfo/DSGetGeneralInfo.xsd"
attributeFormDefault="qualified" elementFormDefault="qualified"
xmlns:mstns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSGetGeneralInfo.xsd"
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSGetGeneralInfo.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
          <xs:element name="DSGetGeneralInfo" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="GetGeneralInfo">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="full_account" type="xs:string" minOccurs="0"/>
                      <xs:element name="name" type="xs:string" minOccurs="0"/>
                      <xs:element name="title" type="xs:string" minOccurs="0"/>
                      <xs:element name="first_name" type="xs:string" minOccurs="0"/>
                      <xs:element name="middle_initial" type="xs:string" minOccurs="0"/>
                      <xs:element name="last_name" type="xs:string" minOccurs="0"/>
                      <xs:element name="name_suffix" type="xs:string" minOccurs="0"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:schema>
      </GetCustomerGeneralInfoResult>
    </GetCustomerGeneralInfoResponse>
  </soap:Body>
</soap:Envelope>
```

```

        <xs:element name="street1" type="xs:string" minOccurs="0"/>
        <xs:element name="street2" type="xs:string" minOccurs="0"/>
        <xs:element name="town" type="xs:string" minOccurs="0"/>
        <xs:element name="state" type="xs:string" minOccurs="0"/>
        <xs:element name="city" type="xs:string" minOccurs="0"/>
        <xs:element name="postal_code" type="xs:string" minOccurs="0"/>
        <xs:element name="country" type="xs:unsignedByte" minOccurs="0"/>
        <xs:element name="division" type="xs:short" minOccurs="0"/>
        <xs:element name="tank_count" type="xs:short" minOccurs="0"/>
        <xs:element name="service_count" type="xs:short" minOccurs="0"/>
        <xs:element name="location_count" type="xs:short" minOccurs="0"/>
        <xs:element name="telephone" type="xs:string" minOccurs="0"/>
        <xs:element name="delivery_hold" type="xs:string" minOccurs="0"/>
        <xs:element name="service_hold" type="xs:string" minOccurs="0"/>
        <xs:element name="terminated" type="xs:string" minOccurs="0"/>
        <xs:element name="association_hold" type="xs:string" minOccurs="0"/>
        <xs:element name="priority_account" type="xs:string" minOccurs="0"/>
        <xs:element name="must_charge_cc" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
</xs:schema>
<diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
    <DSGetGeneralInfo
xmlns="http://www.addsys.com/EnergyAPI/CustomerInfo/DSGetGeneralInfo.xsd">
        <GetGeneralInfo diffgr:id="GetGeneralInfo1" msdata:rowOrder="0">
            <full_account>36906</full_account>
            <name>DALE A. NAGY</name>
            <title xml:space="preserve"></title>
            <first_name xml:space="preserve"></first_name>
            <middle_initial xml:space="preserve"></middle_initial>
            <last_name>DALE A. NAGY</last_name>
            <name_suffix xml:space="preserve"></name_suffix>
            <street1>3217 VELD WAY</street1>
            <street2 xml:space="preserve"></street2>
            <town>CAMERON PARK CA</town>
            <state xml:space="preserve"></state>
            <city>CAMERON PARK CA</city>
            <postal_code>956820000</postal_code>
            <country>0</country>
            <division>4</division>
            <tank_count>1</tank_count>
            <service_count>1</service_count>
            <location_count>0</location_count>

```

```
<telephone>5306765790</telephone>
<delivery_hold>Y</delivery_hold>
<service_hold>N</service_hold>
<terminated>N</terminated>
<association_hold>N</association_hold>
<priority_account>N</priority_account>
<must_charge_cc>N</must_charge_cc>
</GetGeneralInfo>
</DSGetGeneralInfo>
</diffgr:diffgram>
</GetCustomerGeneralInfoResult>
<p_lErrorCode>0</p_lErrorCode>
</GetCustomerGeneralInfoResponse>
</soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetCreditInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves the credit information for a given energy account number/company ID

```
Public Function GetCreditInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Budget start month ("budget_start") - Short value -- <> 0 -->
Budget account
'          (1)AR Balance ("balance") - currency
'          (2)Budget balance ("budget_balance") - currency
'          (3)Past due balance ("past_due_balance") - Short - <> 0 MIN code
'              MIN code values in one's digit:
'              0 = No credit problem
'              1 = On delivery hold but no dunning letter
'              3 = Budget late
'              4 = Balance + delivery > Credit limit/Credit line
'              5 = Balance age + degree day next > credit time limit
'              6 = Balance age > credit time limit
'              7 = Balance > Credit limit/Credit line
'              If the MIN has a ten's digit, they correspond to the dunning letter that should be sent –
'              e.g., 36 means send letter 3, customer's balance age > credit time limit.
'          (3)Last payment date ("last_paid_date") - Date
'          (4)Last payment amount ("last_payment_amount") – Currency
'          (5)Budget payment amount ("budget_payment_amount") - Currency
'          (6)Budget payment date ("budget_payment_date") – Date
'          (7)Last transaction date ("last_transaction_date") - Date && Time
'          (8)credit_line - integer
'          (9)credit_limit_code - integer
'          (10)credit_limit_value_dollars - currency
'          (11)credit_time_code - integer
'          (12)credit_time_value_days - integer
'          (13)mini_code – integer
'          (14)prepayment_amount – Currency
'          (15)installment_balance - Currency
'          (16) Full Account Number (full_account) -- String
'          (17) Budget paid (budget_paid) – Currency
'          (18) Budget adjustment dollars (budget_adj_dollars) -- Currency
'          (19) Budget dollars billed (bgt_dol_billed) -- Currency
'          (20) Current non-budget billed (cur_nbgt_dollars) -- Currency
```



```

'          (21) Prior non-budget billed (prior_nbgt_dollars) -- Currency
'          (22) Settlement period (settlement_period) -- String (Y/N)
'          (23) Total number of Budget payments (num_budget_payments) -- Integer
'          (24) Number of budget payments due (num_of_payments_due) -- Integer
'          (25) Budget start date (bgt_start_date) -- Date
'          (26) Budget end date (bgt_end_date) -- date
'          (27) Budget beginning balance -- BTO (bgt_begin_bal)
'          (28) Settlement start date (settlement_start_date) -- Date
'          (29) Settlement end date (settlement_end_date) -- Date
'          (30) Budget coverage period (bgt_coverage_period) -- Integer
'          -- Account Ageing information
'          (31) Current (bal_curr) -- Currency
'          (32) Over bucket 1 (bal_over1) -- Currency
'          (33) Over bucket 2 (bal_over2) -- Currency
'          (34) Over bucket 3 (bal_over3) -- Currency
'          (35) Over bucket 4 (bal_over4) -- Currency
'          (36) Over bucket 5 (bal_over5) -- Currency
'          (37) Over bucket 6 (bal_over6) -- Currency
'          (38) Over bucket 7 (bal_over7) -- Currency
'          (39) Over bucket 8 (bal_over8) -- Currency
'          (40) Over bucket 9 (bal_over9) -- Currency
'          (41) Over bucket 10 (bal_over10) -- Currency

```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve the account credit information:

```

Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objCreditInfo As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objCreditInfo = objCustomerInfo.GetCreditInfo(iCompanyID,
lAccountNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    If Not objCreditInfo Is Nothing Then
        cARBalance = objCreditInfo.Tables(0).Rows(0).Item(0) ' Account balance
        sBudgetBalance = objCreditInfo.Tables(0).Rows(0).Item(1) ' Budget amount due
        Etc.
    Else ' Should never happen
        ' Unable to locate the account
    End If
End If

```

XML Example

The following example shows how to retrieve the account credit information:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCreditInfo>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetCreditInfo>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetCreditInfoPPD()**License Required: SmartConnect API Prompt Pay Discount information****Description**

Retrieves the credit information, including Discounted amount due and Discount date, for a given energy account number/company ID

Public Function **GetCreditInfoPPD**(ByVal p_iCompanyID As Short, _
 ByVal p_lAccountNum As Integer, _
 ByVal p_sSysUser As String, _
 ByRef p_lErrorCode As Integer) As System.Data.DataSet

Input Parameters

'Accepts: p_lAccountNum -- The Account Number Without Check Digit
 ' p_iCompanyID -- The Energy Company ID
 ' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
 ' Return Type DataSet.Tables(0).Rows (0) Budget start month ("budget_start") - Short value -- <> 0 -->
 Budget account
 ' (1)AR Balance ("balance") - currency
 ' (2)Budget balance ("budget_balance") - currency
 ' (3)Past due balance ("past_due_balance") - Short - <> 0 MIN code
 MIN code values in one's digit:
 0 = No credit problem
 1 = On delivery hold but no dunning letter
 3 = Budget late
 4 = Balance + delivery > Credit limit/Credit line
 5 = Balance age + degree day next > credit time limit
 6 = Balance age > credit time limit
 7 = Balance > Credit limit/Credit line
 If the MIN has a ten's digit, they correspond to the dunning letter that should be sent –
 e.g., 36 means send letter 3, customer's balance age > credit time limit.
 ' (3)Last payment date ("last_paid_date") - Date
 ' (4)Last payment amount ("last_payment_amount") – Currency
 ' (5)Budget payment amount ("budget_payment_amount") - Currency
 ' (6)Budget payment date ("budget_payment_date") – Date
 ' (7)Last transaction date ("last_transaction_date") - Date && Time
 ' (8)credit_line - integer
 ' (9)credit_limit_code - integer
 ' (10)credit_limit_value_dollars - currency
 ' (11)credit_time_code - integer
 ' (12)credit_time_value_days - integer
 ' (13)mini_code – integer
 ' (14)prepayment_amount – Currency
 ' (15)installment_balance - Currency
 ' (16) Full Account Number (full_account) -- String
 ' (17) Budget paid (budget_paid) – Currency
 ' (18) Budget adjustment dollars (budget_adj_dollars) -- Currency
 ' (19) Budget dollars billed (bgt_dol_billed) -- Currency
 ' (20) Current non-budget billed (cur_nbgt_dollars) -- Currency

```

'      (21) Prior non-budget billed (prior_nbgt_dollars) -- Currency
'      (22) Settlement period (settlement_period) -- String (Y/N)
'      (23) Total number of Budget payments (num_budget_payments) -- Integer
'      (24) Number of budget payments due (num_of_payments_due) -- Integer
'      (25) Budget start date (bgt_start_date) -- Date
'      (26) Budget end date (bgt_end_date) -- date
'      (27) Budget beginning balance -- BTO (bgt_begin_bal)
'      (28) Settlement start date (settlement_start_date) -- Date
'      (29) Settlement end date (settlement_end_date) -- Date
'      (30) Budget coverage period (bgt_coverage_period) -- Integer
'      -- Account Ageing information
'      (31) Current (bal_curr) -- Currency
'      (32) Over bucket 1 (bal_over1) -- Currency
'      (33) Over bucket 2 (bal_over2) -- Currency
'      (34) Over bucket 3 (bal_over3) -- Currency
'      (35) Over bucket 4 (bal_over4) -- Currency
'      (36) Over bucket 5 (bal_over5) -- Currency
'      (37) Over bucket 6 (bal_over6) -- Currency
'      (38) Over bucket 7 (bal_over7) -- Currency
'      (39) Over bucket 8 (bal_over8) -- Currency
'      (40) Over bucket 9 (bal_over9) -- Currency
'      (41) Over bucket 10 (bal_over10) -- Currency
'      (42) Discounted amount due -- Currency
'      (43) Earliest prompt pay discount date -- Date

```

Return code	Description
96	Unknown Error Occurred.

XML Example

The following example shows how to retrieve the account credit information:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCreditInfoPPD>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>100</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetCreditInfoPPD>
  </soap:Body>
</soap:Envelope>

```

cCustomerInfo.GetGeneralText()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves general text for a given energy account number/company ID

```
Public Function GetGeneralText(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)General text ("general_text") - string value
```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve the account general text:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objGenText As System.Data.DataSet '
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objGenText = objCustomerInfo.GetGeneralText(iCompanyID,
lAccountNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    If Not objGenText Is Nothing Then
        sGeneralText = objGenText.Tables(0).Rows(0).Item(0)
    Else ' Should never happen
        ' Unable to locate the account
    End If
End If
```

XML Example

The following example shows how to retrieve the account general text:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetGeneralText>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetGeneralText>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetDeliveryLocations()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of delivery locations for a given energy account number/company ID

```
Public Function GetDeliveryLocations(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Tank number ("tank_number") - Short integer
'          (1)Delivery address ("delivery_address") - Char(250)
'          (2)Product code ("product") -- Short integer
'          (3)Product description ("product_desc") -- Char(30)
'          (4)Tank status ("tank_status") -- Char(1) --> "I"=Inactive/Canceled; "T"=Terminated;
'          "A"(All Other Values)=Active
```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve a list of delivery locations:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objTanks As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objTanks = objCustomerInfo.GetDeliveryLocations(iCompanyID,
lAccountNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objTanks.Tables(0).Rows
```

```
        Debug.Print "Tank number=" & Row (0)      & " Address=" & _  
        Row (1) & " Product Code=" & Row (2) & _  
        " Product Description=" & Row (3) & _  
        " Tank status=" & Row (4)  
    Next  
End If
```

XML Example

The following example shows how to retrieve a list of delivery locations:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:GetDeliveryLocations>  
      <add:p_iCompanyID>2</add:p_iCompanyID>  
      <add:p_lAccountNum>151</add:p_lAccountNum>  
      <add:p_sSysUser>WebUser</add:p_sSysUser>  
      <add:p_lErrorCode>0</add:p_lErrorCode>  
    </add:GetDeliveryLocations>  
  </soap:Body>  
</soap:Envelope>
```


cCustomerInfo.GetServiceLocations()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of service locations for a given energy account number/company ID

```
Public Function GetServiceLocations(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)service number ("service_number") - Short integer
'          (1)service address ("service_address") - Char(250)
'          (2)Service contract ("service_contract") -- Char(3)
'          (3)Contract renewal date ("contract_renewal_date") -- Date
'          (4)Service status ("service_status") -- Char(1) --> "I"=Inactive/Canceled; "T"=Terminated;
'          "A"(All Other Values)=Active
```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve a list of service locations:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objService As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objService = objCustomerInfo.GetServiceLocations(iCompanyID,
lAccountNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
```

```
For Each row In objService.Tables(0).Rows
    Debug.Print "Service number=" & Row (0) & " Address=" & _
    Row (1) & " Contract code=" & Row (2) & _
    " Renewal date=" & Row (3) & _
    "Service status=" & Row (4)
Next
End If
```

XML Example

The following example shows how to retrieve a list of service locations:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetServiceLocations>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>101</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetServiceLocations>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetWDMSLocations()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns a list of wdms locations created for a given energy account number/company ID

```
Public Function GetWdmsLocations(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
Return Values			
	p_lErrorCode	Short	0 - If Successful. <> - 0 Error Number

Return Type : System.Data.DataSet

(0) Location number ("location_num") - Short integer
 (1) location name ("name") - Char(40)
 (2) Street1 ("street1") - Char(40)
 (3) Street2 ("street2") - Char(40)
 (4) Town ("town") - Char(40)
 (5) State ("state") - Char(2)
 (6) Zip code ("zip") - Char(10)
 (7) Status ("location_status") - Char(1) --> "I"=Inactive/Canceled;
 "T"=Terminated; "A"(All Other Values)=Active

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve a list of Wdms locations:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objWdms As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objWdms = objCustomerInfo.GetWDMSLocations(iCompanyID,
lAccountNum,cSysUser,lRetCode)
'
```

```
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objWdms.Tables(0).Rows
        Debug.Print "Location number=" & Row (0) & " Name=" & _
        Row (1) & " Street1=" & Row (2) & _
        " Street2=" & Row (3) & _
        " Town=" & Row (4) & _
        " State=" & Row (5) & _
        " Zip=" & Row (6) & _
        " WDMS status=" & Row (7)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of Wdms locations:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetWDMSLocations>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetWDMSLocations>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetMeters()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns a list of meter records created for a given energy account number/company ID

```
Public Function GetMeters(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_lTankNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Integer	Account number without the checkdigit
	p_lTankNum	Integer	0 – All Tanks <> 0 specific tank number
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode	Integer	0 - If Successful. <> - 0 Error Number

Return Type : System.Data.DataSet

- (0) Tank number ("tank_num") - Short integer
- (1) Meter type ("meter_type") - Char(30)
- (2) Meter Serial number ("meter_serial_num ") – Char(15)
- (3) Meter Reading("meter_reading ") – numeric 2 decimals
- (4) Meter reading date ("meter_reading_date ") – Date

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve a list of meters:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objMeters As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objMeters = objCustomerInfo.GetMeters(iCompanyID,
lAccountNum,iTankNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
```

```
        ' Failure - Display error message
Else
    ' Success
    For Each row In objMeters.Tables(0).Rows
        Debug.Print "Tank number=" & Row (0) & " Type=" & _
        Row (1) & " Serial number=" & Row (2)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of meters:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetMeters>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_lTankNum>1</add:p_lTankNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetMeters>
  </soap:Body>
</soap:Envelope>
```

cCustomerInfo.GetItsProducts ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns a list of ITS Products created for a given energy account number/company ID

```
Public Function GetItsProducts(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_lLocationNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Integer	Account number without the checkdigit
	P_lLocationNum	Integer	0 – all locations <> 0 – Specific Location
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode	Integer	0 - If Successful. <> - 0 Error Number

Return Type : System.Data.DataSet

- (0) Location number ("location_num") - Short integer
- (1) Its Product number ("its_product_num") – Short integer
- (2) ITS Product description ("its_description") – Char(30)

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve a list of ITS products:

```
Dim objCustomerInfo as Object
Dim lRetCode as Integer
Dim objITS As System.Data.DataSet
'
Set objCustomerInfo = CreateObject("EnergyAPI.cCustomerInfo")
'
objITS = objCustomerInfo.GetMeters(iCompanyID,
lAccountNum,iLocationNum,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
```

```
        ' Success
    For Each row In objITS.Tables(0).Rows
        Debug.Print "Location number=" & Row (0) & " Number=" & _
            Row (1) & " Dsecription=" & Row (2)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of ITS products:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetItsProducts>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_lLocationNum>1</add:p_lLocationNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetItsProducts>
  </soap:Body>
</soap:Envelope>
```


cCustomerInfo.GetAccountsByGroupCode()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will be called to retrieve a list of accounts that match the group code criteria

Method Signature:

Public Function GetAccountsByGroupCode(ByVal pCompanyID As Short, ByVal pSysUser As String, _
ByVal pAccountNum As Integer, ByVal pGroupCode1 As Integer, ByVal pGroupCode2 As Integer, _
ByVal pIsGroupCode0Valid As Boolean, ByRef pErrCode As Integer) _
As DCEnergyAPI.CustomerInfo.AccountsByGroupCode()

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	int (32 bit integer)	Source/Master account number from which the group code values will be retrieved for querying
	pGroupCode1	Int (32 bit integer)	32bit integer value that will be use to match group code 1
	pGroupCode2	Int (32 bit integer)	32bit integer value that will be use to match group code 2
	pIsGroupCode0Valid	Boolean	If False the query will not filter on 0 group code, If true a 0 value is considered valid and will only retrieve values that are 0 in the table
	pErrCode	Int (32 bit integer)	An error code 103 = Non-zero account number is invalid, 831 = Terminated Account, 504 = no records for matching group code(s)
Return Values			
		DCEnergyAPI.CustomerInfo.AccountsByGroupCode	Data Structure containing account record information

Return code	Description
103	Unable to locate a unique account number. No account was found.
504	No data was found. Change the selection criteria and try again.
831	Customer is terminated.

XML Document**Request:**

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

```

<soap:Body>
  <GetAccountsByGroupCode xmlns="http://www.addsys.com/">
    <pCompanyID>short</pCompanyID>
    <pSysUser>string</pSysUser>
    <pAccountNum>int</pAccountNum>
    <pGroupCode1>int</pGroupCode1>
    <pGroupCode2>int</pGroupCode2>
    <pIsGroupCode0Valid>boolean</pIsGroupCode0Valid>
    <pErrCode>int</pErrCode>
  </GetAccountsByGroupCode>
</soap:Body>
</soap:Envelope>

```

Response:

```

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetAccountsByGroupCodeResponse xmlns="http://www.addsys.com/">
      <GetAccountsByGroupCodeResult>
        <AccountsByGroupCode>
          <AccountNum>int</AccountNum>
          <FullAccount>string</FullAccount>
          <CustomerName>string</CustomerName>
          <GroupCode1>int</GroupCode1>
          <GroupCode2>int</GroupCode2>
        </AccountsByGroupCode>
        <AccountsByGroupCode>
          <AccountNum>int</AccountNum>
          <FullAccount>string</FullAccount>
          <CustomerName>string</CustomerName>
          <GroupCode1>int</GroupCode1>
          <GroupCode2>int</GroupCode2>
        </AccountsByGroupCode>
      </GetAccountsByGroupCodeResult>
      <pErrCode>int</pErrCode>
    </GetAccountsByGroupCodeResponse>
  </soap12:Body>
</soap12:Envelope>

```

.NET Example

```

EnergyAPIWebService.cCustomerInfo custInfo = new EnergyAPIWebService.cCustomerInfo();
custInfo.Url="http://some address" + "cCustomerInfo.asmx";
//
Short companyID = 2;
String sysUser = "WebUser";
int accountNum = 25;
int groupCode1 = 1;
int groupCode2 = 0;
bool isGroupCode0Valid = false;

```

```
int errCode = 0
DCEnergyAPI.CustomerInfo.AccountsByGroupCode retData = null;

retData = custInfo.GetAccountsByGroupCode(companyID, sysUser, accountNum, groupCode1,
groupCode2, isGroupCode0Valid, ref, errCode);

If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cDelivery Class

Overview

This section describes the methods that enable the programmer to retrieve delivery location detailed information as well as delivery history for a given account number and company ID. This class also provides the methods needed to log a phone order, reschedule a phone order and cancel a previously logged phone order. The following methods are covered in this section:

- GetGeneralInfo(): Retrieves delivery location(s) general information
- GetProductPrice(): Returns the product price for a given delivery location
- CalculateDeliveryUnits(): Returns the number of units and tax per unit for a given dollar amount.
- GetTicketInfo(): returns delivery ticket information
- GetDeliveryHistory(): retrieves delivery history for a given account number
- LogPhoneOrder(): Logs a phone order for a given account number/Delivery location.
While still supported in the current version please use LogPhoneOrder_V2 instead
- LogPhoneOrder_V2(): Logs a phone order for a given account number/Delivery location, this version uses a data structure to accept input arguments.
- CancelPhoneOrder(): Cancels a previously logged phone order
- ReschedulePhoneOrder(): Allows the user to reschedule a previously logged phone order.
While still supported in the current version please use ReschedulePhoneOrder_V2 instead
- ReschedulePhoneOrder_V2(): Allows the user to reschedule a previously logged phone order, this version uses a data structure to accept input arguments.
- LoadScheduledProducts(): Returns general information about Scheduled Products
- LogScheduleProductPhoneOrder(): Allows the user to log a Scheduled Product order/ticket

Data Classes and Datasets

Class DeliveryOrder Definition:

Column Name	Data Type	Description
AccountNum	Int	Customer account number without check digit
TankNum	Short	Delivery location tank number
DeliverBy	Short	Must Do Delivery flag, 0=Deliver By; 1=Deliver On
DeliveryDate	Date	Delivery date
DeliveryTime	Short	Time in HHmm format
Units	Int	Units that are to be delivered. NOTE: Ignored when rescheduling a phone order
Instructions	String(60)	Delivery instructions. NOTE: Ignored when rescheduling a phone order
SpecialPrice	Decimal	4-decimal value to indicate the order Special price in cents
PONumber	String(25)	Purchase Order number. NOTE: Ignored when rescheduling a phone order
CCTrxHdrId	Int	As returned by cCreditCard.PreAuthorizeCreditCard(), requires "CCCustomerHdrId", cannot be used in conjunction with "ChargeOnPosting". NOTE: Ignored when rescheduling a phone order
AuthorizationAmt	Decimal	As returned by cCreditCard.GetTransactionChargeAmount(), requires "CCCustomerHdrId", cannot be used in conjunction with "ChargeOnPosting". NOTE: Ignored when rescheduling a phone order
CCCustomerHdrId	Int	As returned by cDelivery.GetDeliveryGeneralInfo(), must be used in conjunction with "CCTrxHdrId" & "AuthorizationAmt" or "ChargeOnPosting". NOTE: Ignored when rescheduling a phone order
ChargeOnPosting	boolean	Set to true if CC is not pre-authorized when logging phone order but the delivery needs to be charged to a CC when delivery is posted, cannot be used in conjunction with "AuthorizationAmt" or "CCTrxHdrID", requires "CCCustomerHdrId". NOTE: Ignored when rescheduling a phone order
CancelReason		Cancel reason to delete a printed ticket (maximum 50 characters)
DeleteSLPPRecord		True/False; When set to true, the system will remove the special price record when deleting a printed ticket.
DeliveryArAddons	DeliveryAddons	An array containing DeliveryAddon data structure. Use the "GetDeliveryArAddons" method to retrieve addons. Modify the "Selected" property to associate/disassociate for addons as needed.

DataSet DCCustomers.Display.DSWdmsScheduledProducts Definition:

Column Name	Data Type	Description
sp_tank_num	short	Scheduled product tank number assigned by the system.
mfg_code	string	Product mfg code
group_code	string	Product group code
stock_id	string	Product stock ID
will_call	string	Will call (Y/N)
order_out	string	Is there an outstanding order (Y/N) (1 char)
size	int	Tank size in gallons
order_num	int	Current order number (int)
deliver_qty	int	Amount of product to be delivered
return_qty	int	Amount of product to return upon delivery (ITS products only)
delivery_interval	short	Number of days between delivery
next_delivery_date	dateTime	Next delivery date
name	string	Product group name
description	string	Product group description
description1	string	Product description
ticket_required	string	Is a ticket required (Y/N)
print_order	string	Should SPDELI print order (Y/N)
bad_weather	string	Is special handling required in bad weather (Y/N)
tank_num	short	Customer tank (delivery location) number. Available only when the tank's taxes, size, etc. are used. Optional field that links scheduled products to tank record
unit_name	string	Unit name for unconverted product
usable	int	Usable gallons in tank
reserve	short	Reserve gallons in tank at time of delivery
price_type	byte	Price type value set in SPADD/SPMODI, i.e., 1 = fixed price; 2 = non-fixed price
base_price_code	short	Base price code
price_sub_level	short	Base price sub level code
tax_jurisdiction	short	Tax jurisdiction
default_supplier	string	Default supplier
tax_1	string	Tax flags
tax_2	string	Tax flags
tax_3	string	Tax flags
tax_4	string	Tax flags
tax_5	string	Tax flags
tax_7	string	Tax flags

tax_8	string	Tax flags
tax_9	string	Tax flags, if using expanded taxes
tax_10	string	Tax flags, if using expanded taxes
tax_11	string	Tax flags, if using expanded taxes
tax_12	string	Tax flags, if using expanded taxes
tax_13	string	Tax flags, if using expanded taxes
tax_14	string	Tax flags, if using expanded taxes
tax_15	string	Tax flags, if using expanded taxes
tax_16	string	Tax flags, if using expanded taxes
tax_17	string	Tax flags, if using expanded taxes
tax_18	string	Tax flags, if using expanded taxes
tax_19	string	Tax flags, if using expanded taxes
tax_20	string	Tax flags, if using expanded taxes
tax_21	string	Tax flags, if using expanded taxes
tax_22	string	Tax flags, if using expanded taxes
tax_23	string	Tax flags, if using expanded taxes
tax_24	string	Tax flags, if using expanded taxes
tax_25	string	Tax flags, if using expanded taxes
tax_26	string	Tax flags, if using expanded taxes
tax_27	string	Tax flags, if using expanded taxes
tax_28	string	Tax flags, if using expanded taxes
tax_29	string	Tax flags, if using expanded taxes
tax_30	string	Tax flags, if using expanded taxes
tax_31	string	Tax flags, if using expanded taxes
tax_32	string	Tax flags, if using expanded taxes
tax_33	string	Tax flags, if using expanded taxes
tax_34	string	Tax flags, if using expanded taxes
tax_35	string	Tax flags, if using expanded taxes
tax_36	string	Tax flags, if using expanded taxes
tax_37	string	Tax flags, if using expanded taxes
tax_38	string	Tax flags, if using expanded taxes
tax_39	string	Tax flags, if using expanded taxes
tax_40	string	Tax flags, if using expanded taxes
commercial	string	Commercial location (Y/N)
exempt_state_diesel	string	Exempt from state diesel tax (Y/N)
exempt_state_gas	string	Exempt from state gas tax (Y/N)
exempt_fed_diesel	string	Exempt from federal diesel tax (Y/N)
exempt_fed_gas	string	Exempt from federal gas tax (Y/N)
phone_order	string	Has a phone order been logged (Y/N)
phone_order_units	int	Unconverted units ordered over the phone
next_delivery_date1	dateTime	Next delivery date
deliver_qty1	int	Amount of product to be delivered
delivery_interval1	short	Number of days between delivery
suspend_delivery	dateTime	Starting date when deliveries will be suspended

reinstate_delivery	dateTime	Starting date when suspended deliveries will be reinstated
instructions1	string	Instructions line 1
instructions2	string	Instructions line 2
instructions3	string	This field provides a third line of instructions for the scheduled product record
instructions4	string	This field provides a fourth line of instructions for the scheduled product record
status	string	Record Status (internal use)
last_maintenance_dt	string	Last modification date of the record (internal use)
last_maintenance_userid	string	User ID for the user that last modified the record (internal use)
conversion_factor_1	double	Conversion factor for the product
tax_jur_desc	string	Tax jurisdiction description
location_num	short	Location number to which product delivered

XML document:

The following is the xml document representation of type DSWdmsScheduledProducts. The example below shows all (2) Scheduled Product records belonging to account 4515 (without the checkdigit) and Location Number 1.

```
<DSWdmsScheduledProducts xmlns="http://www.addsys.com/DSWdmsScheduledProducts.xsd">
  <WdmsScheduledProducts diffgr:id="WdmsScheduledProducts1" msdata:rowOrder="0">
    <sp_tank_num>2</sp_tank_num>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>2</stock_id>
    <will_call>N</will_call>
    <order_out>Y</order_out>
    <size>550</size>
    <order_num>17384</order_num>
    <deliver_qty>300</deliver_qty>
    <return_qty>0</return_qty>
    <delivery_interval>7</delivery_interval>
    <next_delivery_date>2009-10-06T00:00:00-04:00</next_delivery_date>
    <name>LIQUID PRODUCTS</name>
    <description>DISTILLATES</description>
    <description1>HEATING OIL</description1>
    <ticket_required>N</ticket_required>
    <print_order>Y</print_order>
    <bad_weather>N</bad_weather>
    <tank_num>0</tank_num>
    <unit_name>GAL</unit_name>
    <usable>500</usable>
    <reserve>450</reserve>
    <price_type>0</price_type>
    <base_price_code>0</base_price_code>
  </WdmsScheduledProducts>
</DSWdmsScheduledProducts>
```



```
<price_sub_level>32</price_sub_level>
<tax_jurisdiction>0</tax_jurisdiction>
<default_supplier xml:space="preserve"></default_supplier>
<tax_1>N</tax_1>
<tax_2>N</tax_2>
<tax_3>N</tax_3>
<tax_4>N</tax_4>
<tax_5>N</tax_5>
<tax_7>N</tax_7>
<tax_8>N</tax_8>
<tax_9>N</tax_9>
<tax_10>N</tax_10>
<tax_11>N</tax_11>
<tax_12>N</tax_12>
<tax_13>N</tax_13>
<tax_14>N</tax_14>
<tax_15>N</tax_15>
<tax_16>N</tax_16>
<tax_17>N</tax_17>
<tax_18>N</tax_18>
<tax_19>N</tax_19>
<tax_20>N</tax_20>
<tax_21>N</tax_21>
<tax_22>N</tax_22>
<tax_23>N</tax_23>
<tax_24>N</tax_24>
<tax_25>N</tax_25>
<tax_26>N</tax_26>
<tax_27>N</tax_27>
<tax_28>N</tax_28>
<tax_29>N</tax_29>
<tax_30>N</tax_30>
<tax_31>N</tax_31>
<tax_32>N</tax_32>
<tax_33>N</tax_33>
<tax_34>N</tax_34>
<tax_35>N</tax_35>
<tax_36>N</tax_36>
<tax_37>N</tax_37>
<tax_38>N</tax_38>
<tax_39>N</tax_39>
<tax_40>N</tax_40>
<commercial>N</commercial>
<exempt_state_diesel>N</exempt_state_diesel>
<exempt_state_gas>N</exempt_state_gas>
<exempt_fed_diesel>N</exempt_fed_diesel>
<exempt_fed_gas>N</exempt_fed_gas>
<phone_order>N</phone_order>
<phone_order_units>0</phone_order_units>
```

```
<next_delivery_date1>2009-10-06T00:00:00-04:00</next_delivery_date1>
<deliver_qty1>300</deliver_qty1>
<delivery_interval1>7</delivery_interval1>
<suspend_delivery>1960-12-31T00:00:00-05:00</suspend_delivery>
<reinstate_delivery>1960-12-31T00:00:00-05:00</reinstate_delivery>
<instructions1 xml:space="preserve"></instructions1>
<instructions2 xml:space="preserve"></instructions2>
<instructions3 xml:space="preserve"></instructions3>
<instructions4 xml:space="preserve"></instructions4>
<status>A</status>
<last_maintenance_dt>Jan 4 2011 9:32:18:816AM</last_maintenance_dt>
<last_maintenance_userid>TA_call_164881</last_maintenance_userid>
<conversion_factor_1>1</conversion_factor_1>
<tax_jur_desc>0 -</tax_jur_desc>
<location_num>1</location_num>
</WdmsScheduledProducts>
<WdmsScheduledProducts diffgr:id="WdmsScheduledProducts2" msdata:rowOrder="1">
  <sp_tank_num>1</sp_tank_num>
  <mfg_code>LIQ</mfg_code>
  <group_code>DISTIL</group_code>
  <stock_id>1</stock_id>
  <will_call>N</will_call>
  <order_out>Y</order_out>
  <size>275</size>
  <order_num>14161</order_num>
  <deliver_qty>200</deliver_qty>
  <return_qty>0</return_qty>
  <delivery_interval>1</delivery_interval>
  <next_delivery_date>2007-10-15T00:00:00-04:00</next_delivery_date>
  <name>LIQUID PRODUCTS</name>
  <description>DISTILLATES</description>
  <description1>KEROSENE - LIGHT OIL # 1</description1>
  <ticket_required>Y</ticket_required>
  <print_order>Y</print_order>
  <bad_weather>N</bad_weather>
  <tank_num>1</tank_num>
  <unit_name>GALS</unit_name>
  <usable>256</usable>
  <reserve>66</reserve>
  <price_type>0</price_type>
  <base_price_code>1</base_price_code>
  <price_sub_level>0</price_sub_level>
  <tax_jurisdiction>1</tax_jurisdiction>
  <default_supplier xml:space="preserve"></default_supplier>
  <tax_1>Y</tax_1>
  <tax_2>N</tax_2>
  <tax_3>N</tax_3>
  <tax_4>N</tax_4>
  <tax_5>N</tax_5>
```

<tax_7>N</tax_7>
<tax_8>N</tax_8>
<tax_9>N</tax_9>
<tax_10>N</tax_10>
<tax_11>N</tax_11>
<tax_12>N</tax_12>
<tax_13>N</tax_13>
<tax_14>N</tax_14>
<tax_15>N</tax_15>
<tax_16>N</tax_16>
<tax_17>N</tax_17>
<tax_18>N</tax_18>
<tax_19>N</tax_19>
<tax_20>N</tax_20>
<tax_21>N</tax_21>
<tax_22>N</tax_22>
<tax_23>N</tax_23>
<tax_24>N</tax_24>
<tax_25>N</tax_25>
<tax_26>N</tax_26>
<tax_27>N</tax_27>
<tax_28>N</tax_28>
<tax_29>N</tax_29>
<tax_30>N</tax_30>
<tax_31>N</tax_31>
<tax_32>N</tax_32>
<tax_33>N</tax_33>
<tax_34>N</tax_34>
<tax_35>N</tax_35>
<tax_36>N</tax_36>
<tax_37>N</tax_37>
<tax_38>N</tax_38>
<tax_39>N</tax_39>
<tax_40>N</tax_40>
<commercial>N</commercial>
<exempt_state_diesel>N</exempt_state_diesel>
<exempt_state_gas>N</exempt_state_gas>
<exempt_fed_diesel>N</exempt_fed_diesel>
<exempt_fed_gas>N</exempt_fed_gas>
<phone_order>N</phone_order>
<phone_order_units>0</phone_order_units>
<next_delivery_date1>2007-10-15T00:00:00-04:00</next_delivery_date1>
<deliver_qty1>200</deliver_qty1>
<delivery_interval1>1</delivery_interval1>
<suspend_delivery>2007-09-01T00:00:00-04:00</suspend_delivery>
<reinstate_delivery>2007-09-15T00:00:00-04:00</reinstate_delivery>
<instructions1>Gas heat</instructions1>
<instructions2>Electric water</instructions2>
<instructions3 xml:space="preserve"></instructions3>

```
<instructions4 xml:space="preserve"></instructions4>
<status>A</status>
<last_maintenance_dt>Jan 4 2011 9:32:18:816AM</last_maintenance_dt>
<last_maintenance_userid>TA_call_164881</last_maintenance_userid>
<conversion_factor_1>1</conversion_factor_1>
<tax_jur_desc>1 - PENNSYLVANIA</tax_jur_desc>
<location_num>1</location_num>
</WdmsScheduledProducts>
</DSWdmsScheduledProducts>
```

Class DcEnergyAPI.Delivery.DeliveryAddons Definition:

Property Name	Data Type	Description
Selected	Byte	0 = not selected; 1 = selected. All other fields are for information & are not to be modified.
AddonTransId	Short	System defined number (read-only)
AddonHeaderId	Int	System defined number (read-only)
AddonDetailId	Int	System defined number (read-only)
Description	String	Addon description (read-only)
RateType	Byte	Addon rate type (read-only)
RateTypeDesc	String	Description of the rate type (read-only)
RateAmount	Decimal	AR Addon amount (6 decimals) (read-only)
TaxFlagEntries	Byte	Used by tax programs to store taxable amount for posting (read-only)
RatePerUnit	Decimal	The per unit rate (read-only)
Units	Decimal	Addon units (read-only)
ProductAmount	Decimal	Product amount (read-only)
AutomaticAddon	Byte	1 = Automatic addon, will automatically be associated with the delivery (read-only)
AlreadyInPrice	Byte	Indicates if the addon is already included in the price (read-only)
TaxAmount	Decimal	Tax amount (read-only)
AddonPostingCode	Short	Posting code (read-only)
TaxJurisdiction	Short	Tax jurisdiction (read-only)
AddonPostingCodeDesc	String	Posting code description (read-only)

Class DcEnergyAPI.Delivery.DeliveryArAddonRequest Definition:

Property Name	Data Type	Description
AccountNum	Int	E3 delivery account number, without check digit
TankNum	Short	E3 delivery tank number
AsOfDate	DateTime	Delivery as of date

Class DcEnergyAPI.Delivery.DeliveryArAddonsResponse Definition:

Property Name	Data Type	Description
Errors	String[]	An array of errors if any
Addons	DeliveryAddons []	Array of addons

cDelivery.GetDeliveryGeneralInfo ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Gets detailed tank information for one or all delivery locations.

```
Public Function GetDeliveryGeneralInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iTankNum --- Tank number -- 0=All tanks, <> 0 specific tank (delivery location)
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Tank number ("tank_num") - Short Integer
'          (1)Delivery address ("delivery_address") - Char(250)
'          (2)Product code ("product") - Short integer
'          (3)Product description - ("product_desc") - Char(30)
'          (4)Tank size (Usable) - Long integer
'          (5)Ticket status flag ("ticket_status") - "N"one, "A"ssigned, "U"nassigned, "P"rinted
'          (6)Phone order flag ("phone_order") - Char(1) - Y/N
'          (7)Last delivery date - ("last_delivery_date") - Date
'          (8)Last delivery units - ("last_delivery_units") - Long integer (no decimals)
'          (9)Delivery Stop indicator - ("delivery_stop") - Short integer
'          (10)Delivery Reinstate date - ("delivery_reinstate_date") - Date
'          (11)Will call flag ("will_call_tank") - Char(1) - Y/N
'          (12)Period tank ("period_tank") - Char(1) - Y/N
'          (13)Day of week scheduling ("day_of_week_scheduling") - Char(1) - Y/N
'          (14)Delivery Interval ("delivery_interval") - Short integer
'          (15)Usage rate per day ("daily_usage_rate") - Decimal
'          (16)Estimated Inventory ("estimated_inventory") - Long integer
'          (17)Days in Reserve ("days_in_reserve") - Long Integer
'
'          (18) Next delivery date ("due_date") - Date
'          (19) Runout date ("runout_date") - Date
'          (20) Degree day due ("degree_day_due") - Long integer
'          (21) Degree day runout ("degree_day_runout") - Long integer
'          (22) Must charge Credit card flag ("must_charge_cc") -- char(1) Y/N
'          (23) Credit Card customer header id ("cc_customer_hdr_id") -- Long integer
'          (24) Tank status ("tank_status") -- char(1) -- A,I,T
'          (25) Tank size ("size") -- Long integer
'          (26) Reserve ("reserve") -- Long integer
'          (27) K-factor ("kfactor") - Decimal
'          (28) Purchase Order Number ("po_number") - String
```

```
'          (29) PO Expiration ("po_expiration") -- Date [Internal Use]
'          (30) Current Date ("curr_dt") -- Date [Internal Use]
```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to return delivery location (Tank) information:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objTanks As System.Data.DataSet
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
objTanks =
objDelivery.GetDeliveryGeneralInfo(iCompanyID,lAccountNum,iTankNum,cSysUser,lRetCode
)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objTanks.Tables(0).Rows
        Debug.Print "Tank #:" & Row (0) & "Address:" & Row (1), etc.
    Next
End If
```

XML Example

The following example shows how to return delivery location (Tank) information:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:GetDeliveryGeneralInfo>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iTankNum>1</add:p_iTankNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetDeliveryGeneralInfo>
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery. GetProductPrice()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns the product price for a given Company ID/Account number/Delivery location.

```
Public Function GetProductPrice(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iProductCode As Short, _
    ByVal p_dteAsOfDate As Date, _
    ByVal p_lUnits As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'         p_iCompanyID -- The Energy Company ID
'         p_iTankNum --- Tank number (delivery location)
'         p_iProductCode -- Product code
'         p_dteAsOfDate -- Expected date of delivery
'         p_lUnits -- Number of units to be priced
'         p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'         Return Type DataSet.Tables(0).Rows
'         ("average_price") This is the average price per unit – Currency (100ths; includes deviation only when price
plan)
'         ("pp_flag") 'Y' plan price was used, 'V' Variable plan price was used, 'N' no plan price was used - String
'         ("prev_price_out") - Previous price (if any) - Decimal
'         ("price_date_out") - The date that the price came into effect - Date
'         ("prev_price_date_out") - Previous Date (if any) - Date
'         ("remaining_gal") - remaining gallons - Decimal
'         ("error") - N/A for API – Long Integer
'         ("prod_base_price") - BPC price - Decimal
'         ("differential") - the price differential from the plan price (if plan price in use) - Decimal
'         ("deviation") - the deviation field from the tanks table - Decimal
'         ("deviation_date") - the deviation date field from the tanks table - Date
'         ("deviation_adp") - The value of the deviation after the account discount percent was applied - Decimal
'         ("discount_percent") - Discount percent from ACCOUNTS table (applied to price if non zero) - Double
'         ("pending_pp") - Publication Price is pending - String
'         ("followed_price_down") - Indicates that the plan was a lowest price type and the price was lower and
used. - String
'         ("as_of_date") - the date that we're pricing for. – Date
'         ("tax_amount") - product tax amount – Decimal
'         ("total_price") - total price – Decimal (average price + deviation whether price plan or not)
```

Return code	Description

96	Unknown Error occurred.

.NET Example

The following example shows how to look up a product price:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objPrice As System.Data.DataSet
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
objPrice = objDelivery.GetProductPrice(iCompanyID,lAccountNum,iTankNum,iProductCode,
dteDeliveryDate, lUnits, cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    If Not objPrice Is Nothing Then
        CProductPrice = objPrice.Tables(0).Rows(0).Item(0)
        SPlanPrice= objPrice.Tables(0).Rows(0).Item(1)
    Else
        ' Unable to get the product price
    End If
End If
```

XML Example

The following example shows how to look up a product price:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:GetProductPrice>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iTankNum>1</add:p_iTankNum>
      <add:p_iProductCode>3</add:p_iProductCode>
      <add:p_dteAsOfDate>2013-09-01</add:p_dteAsOfDate>
      <add:p_decUnits>100</add:p_decUnits>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetProductPrice>
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery.CalculateDeliveryUnits()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns the units and tax per unit for a given Company ID/Account number/Delivery location/Dollar amount

```
Public Function CalculateDeliveryUnits(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iProductCode As Short, _
    ByVal p_dteAsOfDate As Date, _
    ByVal p_decDeliveryAmount As Decimal, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As DCCustomers.Display.PhoneOrderPrice
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'         p_iCompanyID -- The Energy Company ID
'         p_iTankNum --- Tank number (delivery location)
'         p_iProductCode -- Product code
'         p_dteAsOfDate -- Expected date of delivery
'         p_decDeliveryAmount -- Dollar amount of delivery
'         p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'         Return Type PhoneOrderPrice data Object
'         phoneOrderPrice.Units -- Total units to deliver -- Decimal
'         phoneOrderPrice.TaxPerUnit - Tax rate per unit -- Decimal
'         The remaining fields are used in E3 application and can be ignored.
```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to calculate delivery units:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objUnits As PhoneOrderPrice
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
```

```
objUnits =  
objDelivery.CalculateDeliveyUnits(iCompanyID,lAccountNum,iTankNum,iProductCode,  
dteDeliveryDate,decDeliveryAmount,cSysUser,lRetCode)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
Else  
    ' Success  
    If Not objUnits Is Nothing Then  
        Units = objUnits.Units  
        UnitTax= objUnits.TaxPerUnit  
    Else  
        ' Unable to calculate delivery units  
    End If  
End If
```

XML Example

The following example shows how to look up a product price:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
xmlns:add="http://www.addsys.com/">  
  <soapenv:Header/>  
  <soapenv:Body>  
    <add:CalculateDeliveryUnits>  
      <add:p_iCompanyID>1122</add:p_iCompanyID>  
      <add:p_lAccountNum>1</add:p_lAccountNum>  
      <add:p_iTankNum>1</add:p_iTankNum>  
      <add:p_iProductCode>3</add:p_iProductCode>  
      <add:p_dteAsOfDate>2013-09-01</add:p_dteAsOfDate>  
      <add:p_decDeliveryAmount>100</add:p_decDeliveryAmount>  
      <add:p_sSysUser>WebUser</add:p_sSysUser>  
      <add:p_lErrorCode>0</add:p_lErrorCode>  
    </add:CalculateDeliveryUnits>  
  </soapenv:Body>  
</soapenv:Envelope>
```

cDelivery.GetTicketInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Gets delivery ticket detailed information belonging to a given account number/delivery location.

```
Public Function GetTicketInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iTankNum --- Tank number (delivery location); 0=All tanks or specific tank number
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Ticket status - ("ticket_status") -- Char(1) (U,A,P)
'          (1)Phone order indicator -- ("phone_order") -- Char(1) (Y/N)
'          (2)Ticket number -- ("ticket_num") -- Long integer
'          (2)Phone order units ("phone_order_units") - Char(1) (Y/N)
'          (3)Delivery date ("delivery_date") - date
'          (4)Driver ("driver_name") - Char(50)
'          (5)Delivery instructions - Char(250)
'          (6)Ticket text ("ticket_text") - Char(250)
'          (7)Tank number ("tank_num") -- 16-bit integer (short)
```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve the ticket information:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objTicket As System.Data.DataSet
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
objTicket =
objDelivery.GetTicketInfo(iCompanyID,lAccountNum,iTankNum,cSysUser,lRetCode)
```

```
‘
If lRetCode <> 0 Then
    ‘ Failure - Display error message
Else
    ‘ Success
    For Each row In objTicket.Tables(0).Rows
        Debug.Print “Ticket status: “ & Row (0) & “ Ticket number: “ & Row (1), etc.
    Next
End If
```

XML Example

The following example shows how to retrieve the ticket information:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:GetTicketInfo>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iTankNum>1</add:p_iTankNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetTicketInfo>
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery.GetDeliveryHistory()/cDelivery.GetDeliveryHistoryWithCredits()

- License Required: SmartConnect API Customer Look-up Business Edition**

Important remark: GetDeliveryHistory() method will not return delivery transactions with zero or negative units; you can use GetDeliveryHistoryWithCredits() method to include transactions with zero or negative units.

Description

Retrieves Delivery history for a Given Account Number/Tank number according to a given criteria.

```
Public Function GetDeliveryHistory(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_dteFromDate As Date, _
    ByVal p_dteToDate As Date, _
    ByVal p_lMaxRows As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'    p_iCompanyID -- The Energy Company ID
'    p_iTankNum --- 0 = All tanks, <> 0 specific Tank number (delivery location)
'    p_dteFromDate -- From Date
'    p_dteToDate -- To Date
'    p_lMaxRows -- Number of maximum number of deliveries to return from newest to oldest, send 0 to return
all selected deliveries
'    p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'    Return Type DataSet.Tables(0).Rows (0)Delivery date ("delivery_date") - Date
'        (1)Ticket number ("ticket_num") - Integer
'        (2)Units delivered("units") - number (2 decimals)
'        (3)Cost per unit ("cost_per_unit") - currency
'        (4)Total cost ("net_amount") - currency
'        (5)Tank number ("tank_num") - Short
'        (6)Product ("product") - Char(30)
'        (7) Degree day ("degree_day") - Long integer
'        (8) Gross amount ("gross_amount") – Currency
'        (9) Partial fill flag ("partial_fill") -- String Y/N
'        (10) In tank units (in_tank_units) – Decimal
'        (11) Partial fill reason ("part_fill_reason") -- 16-bit integer
'        (12) Partial fill reason description ("part_fill_reason_desc") -- String
'        (13) Runout flag ("runout") -- String Y/N
'        (14) Runout reason ("runout_reason") -- 16-bit integer
'        (15) Runout reason description ("runout_reason_desc") – String
'        (16) Transaction unique identifier ("trx_unique_key") – Integer
' lmd 27.oct.2015. DC. call# 344254ADD
'        (17) Percent Fill (percent_fill) -- Short Integer
```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve the delivery history for a given account number:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objDelHist As System.Data.DataSet
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
objDelHist =
objDelivery.GetDeliveryHistory(iCompanyID,lAccountNum,iTankNum,dteFromDate,
dteToDate, 0, cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objDelHist.Tables(0).Rows
        Debug.Print "Delivery date: " & Row (0) & " Ticket number: " & Row (1), etc.
    Next
End If
```

XML Example

The following example shows how to retrieve the delivery history for a given account number:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:GetDeliveryHistory>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iTankNum>1</add:p_iTankNum>
      <add:p_dteFromDate>2013-01-01</add:p_dteFromDate>
      <add:p_dteToDate>2013-12-31</add:p_dteToDate>
      <add:p_lMaxRows>5</add:p_lMaxRows>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetDeliveryHistory>
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery.LogPhoneOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

Description

Logs a delivery phone order for a given account number and delivery location.

```
Public Function LogPhoneOrder(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iDeliveryBy As Short, _
    ByVal p_dteDeliveryDate As Date, _
    ByVal p_iDeliveryTime As Short, _
    ByVal p_lUnits As Integer, _
    ByVal p_sInstructions As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iTankNum --- Tank number (delivery location)
'          p_iDeliveryBy -- 0=Deliver on, 1=Deliver by (future use)
'          p_dteDeliveryDate -- Delivery date
'          p_iDeliveryTime -- Delivery time 1=AM, 2=PM (future use)
'          p_lUnits -- Units to be delivered
'          p_sInstructions - Special instructions
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type Integer -- Confirmation number
'
```

Return code	Description
104	Unable to create a log header
106	Unable to create a phone order
565	Check flags (Phone_order='Y' or Ticket_Status = 'P')
96	Unknown Error occurred.

.NET Example

The following example shows how to log a phone order:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim lConfirmationNo as Integer
```



```
‘  
Set objDelivery = CreateObject("EnergyAPI.cDelivery")  
‘  
lConfirmationNo =  
objDelivery.LogPhoneOrder(iCompanyID,lAccountNum,iTankNum,0,dteDeliveryDate,1,150,"B  
E SURE TO LOCK GATE", cSysUser,lRetCode)  
‘  
If lRetCode <> 0 Then  
    ‘ Failure - Display error message  
Else  
    ‘ Success  
    ‘ Display confirmation number  
End If
```

cDelivery.LogPhoneOrderCC()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

This method will be called to log a delivery phone order for a given account number and delivery location when interacting with the credit card module.

Enterprise Signature:

Public Function LogPhoneOrderCC(ByVal p_iCompanyID As Short, ByVal p_lAccountNum As Integer, ByVal p_iTankNum As Short, ByVal p_iDeliveryBy As Short, ByVal p_dteDeliveryDate As Date, ByVal p_iDeliveryTime As Short, ByVal p_lUnits As Integer, ByVal p_sInstructions As String, ByVal p_sSysUser As String, ByVal pCCTrxHdrId As Integer, ByVal pAuthorizationAmt As Decimal, ByVal pCCCustomerHdrId As Integer, ByRef pErrorCode As Integer) As Integer

Web Service Signature:

Public Function LogPhoneOrderCC(ByVal p_iCompanyID As Short, ByVal p_lAccountNum As Integer, ByVal p_iTankNum As Short, ByVal p_iDeliveryBy As Short, ByVal p_dteDeliveryDate As Date, ByVal p_iDeliveryTime As Short, ByVal p_lUnits As Integer, ByVal p_sInstructions As String, ByVal p_sSysUser As String, ByVal pCCTrxHdrId As Integer, ByVal pAuthorizationAmt As Decimal, ByVal pCCCustomerHdrId As Integer, ByRef pErrorCode As Integer) As Integer

Input Parameters

```
'Accepts: AccountNum -- The Account Number Without Check Digit
'      p_iCompanyID -- The Energy Company ID
'      p_iTankNum --- Tank number (delivery location)
'      p_iDeliveryBy -- 0=Deliver By, 1=Deliver on (future use)
'      p_dteDeliveryDate -- Delivery date
'      p_iDeliveryTime -- Delivery time 1=AM, 2=PM (future use)
'      p_lUnits -- Units to be delivered
'      p_sInstructions - Special instructions
'      p_sSysUser - Calling program (IVRUser/WebUser)
'      pCCTrxHdrId -- always 0
'      pAuthorizationAmt -- as returned from call to cCreditCard.LoadCardInfo()
'      pCCCustomerHdrId -- as returned from call to cCreditCard.LoadCardInfo()
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'      Return Type Integer -- Confirmation number
```

cDelivery.LogPhoneOrder_V2()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition (if CC functionality is used)**

This method will be called to log a delivery phone order for a given account number and delivery location while optionally interacting with the credit card module. See also "[General Steps to Logging a Phone Order](#)"

Method Signature:

Public Function LogPhoneOrder_V2(ByVal pCompanyID As Short, ByVal pOrderInfo As DCEnergyAPI.Delivery.DeliveryOrder, ByRef pErrorCode As Integer) As Integer

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pOrderInfo	Object of type DeliveryOrder	Refer to explanation of structure above.
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
		32 bit integer	Confirmation number for the task performed

Return code	Description
1073	This customer's Must Charge Credit Card flag is set to Yes. You must pre-authorize the charge for this phone order
1445	The customer's source of charge is not setup for this type of transaction.
122	Invalid input parameters

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LogPhoneOrder_V2>
      <!--type: short-->
      <add:pCompanyId>2</add:pCompanyId>
      <!--type: string-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pOrderInfo>
        <!--type: int-->
        <add:AccountNum>25</add:AccountNum>
      </add:pOrderInfo>
    </add:LogPhoneOrder_V2>
  </soap:Body>
</soap:Envelope>
```

```
<!--type: short-->
<add:TankNum>1</add:TankNum>
<!--type: short-->
<add:DeliveryBy>0</add:DeliveryBy>
<!--type: dateTime-->
<add:DeliveryDate>2015-03-24T00:00:00</add:DeliveryDate>
<!--type: short-->
<add:DeliveryTime>1000</add:DeliveryTime>
<!--type: int-->
<add:Units>100</add:Units>
<!--type: Decimal-->
<add:SpecialPrice>175.0000</add:SpecialPrice>
<!--type: string-->
<add:Instructions>Some Instructions</add:Instructions>
<!--type: int-->
<add:CCTrxHdrId>4</add:CCTrxHdrId>
<!--type: decimal-->
<add:AuthorizationAmt>100</add:AuthorizationAmt>
<!--type: int-->
<add:CCCcustomerHdrId>5</add:CCCcustomerHdrId>
<!--type: boolean-->
<add:ChargeOnPosting>false</add:ChargeOnPosting>
<!--type: string-->
<add:PONumber>PONumber</add:PONumber>
</add:pOrderInfo>
<!--type: int-->
<add:pErrCode>0</add:pErrCode>
</add:LogPhoneOrder_V2>
</soap:Body>
</soap:Envelope>
```

cDelivery.ReschedulePhoneOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

Description

Allows the user to reschedule a previously logged phone order.

```
Public Function ReschedulePhoneOrder(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iDeliveryBy As Short, _
    ByVal p_dteDeliveryDate As Date, _
    ByVal p_iDeliveryTime As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iTankNum --- Tank number (delivery location)
'          p_iDeliveryBy -- 0=Deliver By, 1=Deliver on (future use)
'          p_dteDeliveryDate -- Delivery date
'          p_iDeliveryTime -- Delivery time 1=AM, 2=PM (future use)
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type Integer -- Confirmation number
'
```

Return code	Description
104	Unable to create a log header
107	Phone order not found
108	Unable to reschedule phone order
96	Unknown Error occurred.

.NET Example

The following example shows how to reschedule a phone order:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim lConfirmationNo as Integer
'
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
'
```

```
lConfirmationNo =  
objDelivery.RechedulePhoneOrder(iCompanyID,lAccountNum,iTankNum,0,dteDeliveryDate,1,  
cSysUser,lRetCode)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
Else  
    ' Success  
    ' Display confirmation number  
End If
```

cDelivery.ReschedulePhoneOrder_V2()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

Allows the user to reschedule a previously logged phone order.

Method Signature:

Public Function ReschedulePhoneOrder_V2(ByVal pCompanyID As Short, ByVal pOrderInfo As DCEnergyAPI.Delivery.DeliveryOrder, ByRef pErrorCode As Integer) As Integer

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pOrderInfo	Object of type DeliveryOrder	Refer to explanation of structure above.
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
		32 bit integer	Confirmation number for the task performed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ReschedulePhoneOrder_V2>
      <!--type: short-->
      <add:pCompanyId>2</add:pCompanyId>
      <!--type: string-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pOrderInfo>
        <!--type: int-->
        <add:AccountNum>25</add:AccountNum>
        <!--type: short-->
        <add:TankNum>1</add:TankNum>
        <!--type: short-->
        <add:DeliveryBy>0</add:DeliveryBy>
        <!--type: dateTime-->
        <add:DeliveryDate>2015-03-27T00:00:00</add:DeliveryDate>
        <!--type: short-->
        <add:DeliveryTime>1000</add:DeliveryTime>
      </add:pOrderInfo>
      <!--type: int-->
      <add:pErrCode>0</add:pErrCode>
    </add:ReschedulePhoneOrder_V2>
  </soap:Body>
</soap:Envelope>
```

cDelivery.CancelPhoneOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

Description

Cancels a previously logged phone order.

```
Public Function CancelPhoneOrder(ByVal p_iCompanyID As Short, _  
    ByVal p_lAccountNum As Integer, _  
    ByVal p_iTankNum As Short, _  
    ByVal p_sSysUser As String) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit  
'          p_iCompanyID -- The Energy Company ID  
'          p_iTankNum --- Tank number (delivery location)  
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: Return Type Integer -- 0 If Successful : <> 0 Error Number  
,
```

Return code	Description
104	Unable to create a log header
107	Phone order not found
96	Unknown Error occurred.

.NET Example

The following example shows how to cancel a phone order:

```
Dim objDelivery as Object  
Dim lRetCode as Integer  
,  
Set objDelivery = CreateObject("EnergyAPI.cDelivery")  
,  
lRetCode = objDelivery.CancelPhoneOrder(iCompanyID,lAccountNum,iTankNum,cSysUser)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
Else  
    ' Success  
    ' Phone order canceled  
End If
```


cDelivery.CancelPrintedTicket()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

Allows the user to delete a previously printed ticket.

***NOTE* THIS METHOD SHOULD ONLY BE CALLED IF THE TICKET STATUS ON THE TANK ="P"rinted**

Method Signature:

Public Function CancelPrintedTicket(ByVal pCompanyID As Short, ByVal pOrderInfo As DCEnergyAPI.Delivery.DeliveryOrder, ByRef pErrorCode As Integer) As Integer

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pOrderInfo	Object of type DeliveryOrder	Refer to explanation of structure above.
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
		32 bit integer	Confirmation number for the task performed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CancelPrintedTicket>
      <!--type: short-->
      <add:pCompanyId>2</add:pCompanyId>
      <!--type: string-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pOrderInfo>
        <!--type: int-->
        <add:AccountNum>25</add:AccountNum>
        <!--type: short-->
        <add:TankNum>1</add:TankNum>
        <!--type: int-->
        <add:TicketNum>12345</add:TicketNum>
        <!--type: string-->
        <add:CancelReason>Enter 50-char cancel reason here</add: CancelReason >
        <!--type: boolean-->
        <add>DeleteSLPPRecord>true</add: DeleteSLPPRecord>
      </add:pOrderInfo>
      <!--type: int-->
      <add:pErrCode>0</add:pErrCode>
    </add: CancelPrintedTicket >
  </soap:Body>
</soap:Envelope>
```

```
</soap:Body>  
</soap:Envelope>
```

cDelivery.GetDegreeDayInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns the Degree day information based on date or degree day.

if we receive a date, then we get the degree day associated with the date for the specified table.

if we receive a degree day, we return the date closest to the degree day (this is the same logic we use to calculate the estimated delivery date in E3 components (ie delivery display)).

```
Public Function GetDegreeDayInfo(ByVal p_iCompanyID As Short, _
    ByVal p_iDDTable As Byte, _
    ByVal p_dteDeliveryDate_In As Date, _
    ByVal p_lDegreeDay_In As Integer, _
    ByVal p_lHotWaterGallons As Integer, _
    ByVal p_decKFactor As Decimal, _
    ByVal p_sSeparateHWHeater As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As DCEnergyAPI.Delivery.DegreeDayDateInfo
```

Input Parameters

```
'Accepts: p_iCompanyID -- The Energy Company ID
'          p_iDDTable -- Degree day table (Climat zone)
'          p_dteDeliveryDate -- Delivery Date, if we receive a date, the API will get the degree day associated with the
date for the specified table. Pass '01/01/1900' if none
'          p_lDegreeDay -- Degree day, if we receive a degree day, the API will get the date closest to that degree day.
Pass zero if none
'          p_lHotWaterGallons -- Hot Water gallons, needed only if a degree day is passed.- Should pass 0 if none
'          p_decKFactor -- Kfactor, needed if we pass a degree day and we're looking for a date - should pass 0 if none
'          p_sSeparateHWHeater -- Separate Hot Water Heater (Y/N), needed if we pass a degree day and we're
looking for a date - should pass N if none
'          p_sSysUser - Calling program (IVRUser/WebUser)
' Note: Hot water gallons, Kfactor, and Separate Hot Water Heater are optional, and should be passed only if a
Degree day is passed and this information is available to perform degree day adjustment based on hot water
consumption. Otherwise you should pass 0, 0.0, and "N"
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DegreeDayDateInfo data class
'          DegreeDayDateInfo.DegreeDay -- Computed Degree day if we receive a date
'          DegreeDayDate.DeliveryDueDate -- Associated Date if we receive a degree day
```

.NET Example

The following example shows how to look up a product price:

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim objDDInfo As DegreeDayDate
```

```
'  
' Input variables  
Dim iCompanyID as Short = 1  
Dim iDDTable As Byte = 1  
Dim dteDeliveryDateIn as DateTime = New DateTime(1900,01,01)  
Dim lDgreeDay as Integer = 4200  
Dim lHotWaterGallons as Integer = 0  
Dim decKFactor As Integer = 0.0  
Dim sSeparateHWHeater as String = "N"  
'  
' Variables to store the results  
Dim DDOut As Integer = 0  
Dim DeliveryDueDate as DateTime = new DateTime(1900,01,01)  
  
Set objDelivery = CreateObject("EnergyAPI.cDelivery")  
'  
objDDInfo =  
objDelivery.GetDegreeDayInfo(iCompanyID,iDDTable,dteDeliveryDateIn,lDgreeDayIn,lHotW  
aterGallons,decKFactor,sSeparateHFHeater,cSysUser,lRetCode)  
'  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
Else  
    ' Success  
    If Not objDDInfo Is Nothing Then  
        DDOut = objDDInfo.DegreeDay  
        DeliveryDueDate= objDDInfo.DeliveryDueDate  
    Else  
        ' Unable to Get Degree Day Info  
    End If  
End If  
End If
```

cDelivery. ITRForecasting ()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Maintenance/Tank Monitor Business Edition**

Description

Updates the TANKS table with the new electronic tank reading. We use the new electronic tank reading date and the new electronic tank reading value in order to calculate and update the last electronic percent/unit reading, the last electronic tank reading date, the potential pending delivery flag and finally the next delivery date or the degree day next in the TANKS table. A log entry will be created upon a successful update.

```
Public Function ITRForecasting(ByVal p_iCompanyID As Short,
    ByVal p_sSysUser As String,
    ByVal p_lAccountNum As Integer,
    ByVal p_iTankNum As Short,
    ByVal p_dteITRDate As Date,
    ByVal p_decITRValue As Decimal) As Integer
```

Input Parameters

```
'Accepts: p_iCompanyID -- The Energy Company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
'          p_lAccountNum -- The Account Number Without Check Digit
'          p_iTankNum --- Tank number (delivery location)
'          p_dteITRDate - Enter inventory reading date
'          p_decITRValue = reading value in % or units.
```

Important Note: When entering the p_decITRValue = reading value in % or units, you will need to manually review the Product Supplemental table with the customer to get the list of product codes that use Delivery Input In-Tank Units as a percent or as a units value.

Return Values

```
'Returns: Return Type Integer -- 0 If Successful: <> 0 Error Number
'
```

Return code	Description
10018	No account/tank data for this account.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
1091	Electronic Tank Reading not allowed on this tank.
1093	Invalid Electronic Tank Reading Date.
1092	Electronic Tank Reading Date cannot be greater than the current date.
1094	Invalid Electronic Tank Reading Value.
1023	No automatic forecasting is available without a previous delivery.
1020	Units to forecast are less than reserve. Cannot reforecast.
1025	Next delivery date less than or equal to last delivery date. Cannot forecast.
1011	There is a problem updating data in a table. The process did not complete. Call your System

.NET Example

The following example shows how to update the TANK using the ITRForecasting method.

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim dteITRDate as DateTime
Dim decITRValue as decimal
Dim iCompanyID as Short

Set objDelivery = CreateObject("EnergyAPI.cDelivery")

lRetCode = objDelivery.ITRForecasting(iCompanyID, cSysUser,
lAccountNum,iTankNum,dteITRDate, decITRValue )
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    ' Tank updated
End If
```

XML Example

The following example shows how to update the TANK using the ITRForecasting method.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add: ITRForecasting >
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iTankNum>1</add:p_iTankNum>
      <add:p_dteITRDate>03/03/2011</add:p_dteITRDate >
      <add:p_decITRValue >250</add:p_decITRValue >
    </add: ITRForecasting >
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery.LoadScheduledProducts()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Maintenance/Tank Monitor Business Edition**

Description

Returns a Typed Dataset containing general information about the requested Scheduled Product(s).

```
Public Function LoadScheduledProducts
    (ByVal pCompanyId As Short
    , ByVal pAccountNum As Integer
    , ByVal pLocationNum As Short
    , ByVal pSpTankNum As Short
    , ByVal pSysUser As String)
    As DCCustomers.Display.DSWdmsScheduledProducts
```

Input Parameters

Variable	Data Type	Description
pCompanyId	Short (16 bit integer)	The E3 Company ID
pAccountNum	Integer (32 bit integer)	Account number (without check digit). Required value
pLocationNum	Short (16 bit Integer)	WDMS location number. If value is 0 data will be returned for all locations that match other criteria.
pSpTankNum	Short (16 bit Integer)	Scheduled Product Tank Number. If value is 0 data will be returned for all Scheduled Products that match other criteria
pSysUser	String	Calling Program (IVRUser/WebUser/WebTest/WebLive/etc...)

Return Values

Variable	Data Type	Description
	DCCustomers.Display.DSWdmsScheduledProducts : System.Data.DataSet	One or more rows containing information about the requested Scheduled Product(s).

.NET Example

The following example shows how to retrieve information for a specific Scheduled Product.

```
Dim objDelivery as Object
Dim dsRet as DCCustomers.Display.DSWdmsScheduledProducts
Dim lAccountNum as Integer
Dim iLocationNum as Short
Dim iSpTankNum as Short
Dim iCompanyID as Short
```

```
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
```

```
dsRet = objDelivery.LoadScheduledProducts(iCompanyID, lAccountNum, iLocationNum,  
iSpTankNum, cWebUser)
```

```
'Process he returned dataset  
If Not dsRet Is Nothing Then  
    DisplayData(dsRet)  
End If
```

XML Example

The following example shows how to retrieve information for a specific Scheduled Product.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:LoadScheduledProducts>  
      <add:pCompanyId>2</add:pCompanyId>  
      <add:pAccountNum>100</add:pAccountNum>  
      <add:pLocationNum>1</add:pLocationNum>  
      <add:pSpTankNum>1</add:pSpTankNum>  
      <add:pSysUser>WebUser</add:pSysUser>  
    </add:LoadScheduledProducts>  
  </soap:Body>  
</soap:Envelope>
```


cDelivery.LogScheduledProductPhoneOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Maintenance/Tank Monitor Business Edition**

Description

Updates the SP_HEADER table with the new phone order units and next delivery date, this information is then used to pull tickets using E3 functionality.

A log entry will be created upon successfully updating the Scheduled Product record.

```
Public Function LogScheduledProductPhoneOrder
    (ByVal pCompanyId As Short
    , ByVal pAccountNum As Integer
    , ByVal pLocationNum As Short
    , ByVal pSpTankNum As Short
    , ByVal pPhoneOrderUnits As Integer
    , ByVal pNextDeliveryDate As DateTime
    , ByVal pSysUser As String) As Integer
```

Input Parameters

Variable	Data Type	Description
pCompanyId	Short (16 bit integer)	The E3 Company ID
pAccountNum	Integer (32 bit integer)	Account number (without check digit). Required value.
pLocationNum	Short (16 bit Integer)	WDMS location number. Required Value.
pSpTankNum	Short (16 bit Integer)	Scheduled Product Tank Number. Required Value.
pPhoneOrderUnits	Integer (32 bit integer)	Delivery units that are to be logged with this phone order. Required Value.
pNextDeliveryDate	Date	The date when the delivery is to be made. Required Value.
pSysUser	String	Calling Program (IVRUser/WebUser/WebTest/WebLive/etc...)

Return Values

Variable	Data Type	Description
	Integer	Error code. 0 = Success; Non-zero = unsuccessful operation.

Return Codes

Return code	Description
831	Customer is terminated.
1388	The selected Location is not active.
1333	Product does not exist. (i.e. unable to find a valid combination of Account, Location, SP Tank Number)
1389	Cannot log Schedule Product Phone Order. Already has an existing ticket / order out. The "order_out" and/or "phone_order" flag(s) are turned on (set to "Y")

1374	The quantity delivered must not exceed the tank size. <i>The "phone_order_units" cannot be greater than the "size. Ignored if "size" = 0</i>
1390	Delivery is currently suspended. <i>The "next_delivery_date" cannot be greater than or equal to the "suspend_delivery" date and less than the "reinstate_delivery" date.</i>
984	The date cannot be earlier than today. <i>The "next_delivery_date" cannot be greater than the master file (database) date</i>

.NET Example

The following example shows how to log a Scheduled Product phone order.

```
Dim objDelivery as Object
Dim lRetCode as Integer
Dim lAccountNum as Integer
Dim iLocationNum as Short
Dim iSpTankNum as Short
Dim iPhoneOrderUnits as Short
Dim dtNextDeliveryDate as Date
Dim iCompanyID as Short
```

```
Set objDelivery = CreateObject("EnergyAPI.cDelivery")
```

```
lRetCode = objDelivery.LogScheduledProductPhoneOrder(iCompanyID, lAccountNum,
iLocationNum, iSpTankNum, iPhoneOrderUnits, dtNextDeliveryDate, cWebUser)
```

```
if lRetCode = 0 Then
    'Display success message
Else
    'Something Went Wrong
    DisplayMessage(lRetCode)
End If
```

XML Example

The following example shows how to log a Scheduled Product phone order.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LogScheduledProductPhoneOrder>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pAccountNum>100</add:pAccountNum>
      <add:pLocationNum>1</add:pLocationNum>
      <add:pSpTankNum>1000</add:pSpTankNum>
      <add:pPhoneOrderUnits>4567</add:pPhoneOrderUnits>
      <add:pNextDeliveryDate>2013-08-17</add:pNextDeliveryDate>
```

```
    <add:pSysUser>WebUser</add:pSysUser>
  </add:LogScheduledProductPhoneOrder>
</soap:Body>
</soap:Envelope>
```

cDelivery.ComputeTaxes()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to retrieve tax details and compute tax totals for given delivery (tank) location.

**Public Function ComputeTaxes(ByVal pCompanyID As Short,
 ByVal pSysUser As String,
 ByVal pAccountNum As Integer,
 ByVal pTankNum As Short,
 ByVal pPricePerUnit As Decimal,
 ByVal pTaxDate As Date,
 ByVal pUnits As Decimal,
 ByRef pTotalTaxAmount As Decimal,
 ByVal pConversionFactor As Decimal,
 ByRef pTotalRatePerUnit As Decimal,
 ByRef pErrorCode As Integer) As DCE3Common.DSTaxLines**

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	int (32 bit integer)	Source account # (no checkdigit)
	pTankNum	Short (16 bit integer)	Delivery location #
	pPricePerUnit	Decimal	Price per unit in cents .
	pTaxDate	Date	Tax date
	pUnits	Decimal	Units delivered
	pTotalTaxAmount	Decimal	Total tax amount computed by the system (dollars)
	pConversionFactor	Decimal	Conversion factor
	pTotalRatePerUnit	Decimal	Total rate/unit computed by the system (cents)
	pErrorCode	Int (32 bit integer)	0:success, <>0: error code
Return Values			
		DCE3Common.DSTaxLines	Data Structure containing tax detail lines.

DSTaxLines

Datatype/column name	Description
short tax_type	Tax #
short tax_type_pc	Tax type posting code
decimal taxable_amt	Taxable amount (in dollars)
decimal tax_rate	Tax rate (% when cent_or_p is 'P' or cents)
string cent_or_p	1 character code: 'C'ent or 'P'ercent
decimal tax_rate_per_unit	Rate per unit (in cents)
date tax_eff_date	effective date
decmlial tax_amt	Tax Amount (in dollars)
string jur_desc	Tax jurisdiction description
string cat_desc	Category description
string rate_desc	Rate description
string note	note

.NET Example

The following example shows how to retrieve tax information (details) for given account/tank/delivery

```

Dim DSTaxes As DCE3Common.DSTaxLines
Dim TotalTaxAmt As Decimal = 0.0
Dim TotalRatePerUnit As Decimal = 0.0
Dim ErrorCode As Integer = 0
Dim companyID As Short = 1600
Dim accountNum As Integer = 100
Dim tankNum As Short = 1
Dim pricePerUnitCents As Decimal = 448.76
Dim taxDate As Date = "01/16/2017"
Dim unitsDelivered As Decimal = 250.0
Dim conversionFactor As Decimal = 1.0
Using delivery As EnergyAPI.cDelivery = New EnergyAPI.cDelivery
    DSTaxes = delivery.ComputeTaxes(companyID, _
        "WebUser", _
        accountNum, _
        tankNum, _
        pricePerUnitCents, _
        taxDate, _
        unitsDelivered, _
        TotalTaxAmt, _
        conversionFactor, _
        TotalRatePerUnit, _
        ErrorCode)
End Using
If ErrorCode <> 0 Then
    ' RETRIEVE AND DISPLAY ERROR CODE
Else
    ' ITERATE THROUGH THE DSTaxes to get the tax details
    ' the total tax amount is in TotalTaxAmt, the total rate per unit in TotalRatePerUnit

```

End If

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:ComputeTaxes>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>100</add:pAccountNum>
      <add:pTankNum>1</add:pTankNum>
      <add:pPricePerUnit>448.76</add:pPricePerUnit>
      <add:pTaxDate>2017-01-16</add:pTaxDate>
      <add:pUnits>250.0</add:pUnits>
      <add:pTotalTaxAmount>0</add:pTotalTaxAmount>
      <add:pConversionFactor>1.0</add:pConversionFactor>
      <add:pTotalRatePerUnit>0</add:pTotalRatePerUnit>
      <add:pErrorCode>0</add:pErrorCode>
    </add:ComputeTaxes>
  </soapenv:Body>
</soapenv:Envelope>
```

cDelivery.GetDeliveryARAddons()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Delivery Ticket Business Edition**

This method will be called to retrieve a list of delivery AR Add-ons for the delivery location

Method Signature:

```
Public Function GetDeliveryARAddons(ByVal pCompanyID As Short, ByVal pSysUser As String, ByVal
pRequest As DCEnergyAPI.Delivery.DeliveryArAddonsRequest) As
DCEnergyAPI.Delivery.DeliveryArAddonsResponse
```

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pRequest	Object of type DeliveryArAddonsRequest	Refer to explanation of structure above.
Return Values			
	Data structure	DeliveryArAddonResponse	

XML document:**Request**

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetDeliveryARAddons xmlns="http://www.addsys.com/">
      <pCompanyID>short</pCompanyID>
      <pSysUser>string</pSysUser>
      <pRequest>
        <AccountNum>int</AccountNum>
        <TankNum>short</TankNum>
        <AsOfDate>dateTime</AsOfDate>
      </pRequest>
    </GetDeliveryARAddons>
  </soap12:Body>
</soap12:Envelope>
```

Response

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetDeliveryARAddonsResponse xmlns="http://www.addsys.com/">
      <GetDeliveryARAddonsResult>
```

```
<Errors>
  <string>string</string>
  <string>string</string>
</Errors>
<Addons>
  <DeliveryAddons>
    <Selected>unsignedByte</Selected>
    <AddonTransId>short</AddonTransId>
    <AddonHeaderId>int</AddonHeaderId>
    <AddonDetailId>int</AddonDetailId>
    <Description>string</Description>
    <RateType>unsignedByte</RateType>
    <RateTypeDesc>string</RateTypeDesc>
    <RateAmount>decimal</RateAmount>
    <AddonAmount>decimal</AddonAmount>
    <TaxFlagEntries>unsignedByte</TaxFlagEntries>
    <RatePerUnit>decimal</RatePerUnit>
    <Units>decimal</Units>
    <ProductAmount>decimal</ProductAmount>
    <AutomaticAddon>unsignedByte</AutomaticAddon>
    <AlreadyInPrice>unsignedByte</AlreadyInPrice>
    <TaxAmount>decimal</TaxAmount>
    <AddonPostingCode>short</AddonPostingCode>
    <TaxJurisdiction>short</TaxJurisdiction>
    <AddonPostingCodeDesc>string</AddonPostingCodeDesc>
  </DeliveryAddons>
  <DeliveryAddons>
    <Selected>unsignedByte</Selected>
    <AddonTransId>short</AddonTransId>
    <AddonHeaderId>int</AddonHeaderId>
    <AddonDetailId>int</AddonDetailId>
    <Description>string</Description>
    <RateType>unsignedByte</RateType>
    <RateTypeDesc>string</RateTypeDesc>
    <RateAmount>decimal</RateAmount>
    <AddonAmount>decimal</AddonAmount>
    <TaxFlagEntries>unsignedByte</TaxFlagEntries>
    <RatePerUnit>decimal</RatePerUnit>
    <Units>decimal</Units>
    <ProductAmount>decimal</ProductAmount>
    <AutomaticAddon>unsignedByte</AutomaticAddon>
    <AlreadyInPrice>unsignedByte</AlreadyInPrice>
    <TaxAmount>decimal</TaxAmount>
    <AddonPostingCode>short</AddonPostingCode>
    <TaxJurisdiction>short</TaxJurisdiction>
    <AddonPostingCodeDesc>string</AddonPostingCodeDesc>
  </DeliveryAddons>
</Addons>
</GetDeliveryARAddonsResult>
</GetDeliveryARAddonsResponse>
</soap12:Body>
</soap12:Envelope>
```


cService Class

Overview

This section describes the methods that enable the programmer to retrieve service location detailed information, service history as well as logging service work orders for a given account number and company ID. The following methods are covered in this section:

- GetGeneralInfo(): Retrieves service location(s) general information
- GetServiceCallInfo(): returns service call information
- GetServiceHistory(): retrieves service history for a given account number
- GetCallReason():Retrieves a collection of service call reasons for a given energy account number/service location/company ID.
- GetDaySegment():Retrieves a collection of day segments for a given energy company ID.
- GetAvailableSlot():Retrieves a collection of calendar scheduling slots for a given energy account, service location, call reason(s) and schedule date & time window.
- LogWorkOrder():Logs a service work order for a given account, service location, call reason and preferred date/time range (or selected scheduling slot). Also, if successful, returns a confirmation number (i.e. work order #) and scheduled date & time window
- LogWorkOrderNTC():Logs a service work order for a given account, service location, call reason and preferred date/time range (or selected scheduling slot). Also, if successful, returns a confirmation number (i.e. work order #) and scheduled date & time window. This method **replaces** LogWorkOrder() when the scheduling system being used is NT Calendar.
-
- RescheduleWorkOrder():Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window.
- RescheduleWorkOrderWithNotes():Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window. Allows for updating call notes.
- RescheduleWorkOrderNTC():Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window. This method **replaces** RescheduleWorkOrder() & RescheduleWorkOrderWithNotes() when the scheduling system being used is NT Calendar.
- CancelWorkOrder():Cancels a service work order for a given account, service location, work order

cService.GetGeneralInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Gets detailed service information for one or all service locations.

```
Public Function GetGeneralInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iServiceNum As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iServiceNum --- Service number -- 0=All service locations, <> 0 specific service location
'
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
```

```
'          Return Type DataSet.Tables(0).Rows
'          (0) Service location number ("service_num") - Short
'          (1) Service address ("service_address") - Char (250)
'          (2) Service contract ("service_contract") - Char (3)
'          (3) Contract renewal date ("contract_renewal_date") - Date
'          (4) Last cleaning date ("last_clean_date") - Date
'          (5) Service Category ("service_category") - Short
'          (6) Service Category Description ("service_category_desc") - char (40)
'          (7) Date Last Service ("date_last_service") - Date
'          (8) # of uncompleted calls ("number_of_calls") - Short
'          (9) Tank Warranty Amount ("tank_warr_amount") - Currency
'          (10) Tank Warranty Date ("tank_warr_date") - Date
'          (11) Service Contract SubLevel Code ("contract_sub_level") - Short
'          (12) Contract Description ("contract_description") - char (40)
'          (13) PMI ("cleaning_ticket") - Char (1) [Y/N]
'          (14) # of uncompleted Pegasus cleaning calls
```

```
("number_peg_cleaning_calls") - Short
```

.NET Example

The following example shows how to return the service location information:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvLocation As System.Data.DataSet
'
Set objService = CreateObject("EnergyAPI.cService")
'
```

```
objSrvLocation =  
objService.GetGeneralInfo(iCompanyID,lAccountNum,iServiceNum,cSysUser,lRetCode)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
Else  
    ' Success  
    For Each row In objSrvLocation.Tables(0).Rows  
        Debug.Print "Service #:" & Row (0) & "Address:" & Row (1), etc.  
    Next  
End If
```

cService.GetServiceCallInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves service call details for a given account number/service location

Public Function **GetServiceCallInfo**(ByVal p_iCompanyID As Short, _
 ByVal p_lAccountNum As Integer, _
 ByVal p_iServiceNum As Short, _
 ByVal p_sSysUser As String, _
 ByRef p_lErrorCode As Integer) As System.Data.DataSet

Input Parameters

'Accepts: p_lAccountNum -- The Account Number Without Check Digit
 ' p_iCompanyID -- The Energy Company ID
 ' p_iServiceNum --- Service number (Service location)
 ' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
 '
 ' Return Type DataSet.Tables(0).Rows
 ' (0) Work order number ("wo_number") - Long integer
 ' (1) Call Reason 1 ("call_reason_1") - Char (40)
 ' (2) Call Reason 2 ("call_reason_2") - Char (40)
 ' (3) Call Reason 3 ("call_reason_3") - Char (40)
 ' (4) Scheduled date & time ("scheduled_dt") - Date & time
 ' (5) Call date & time ("call_dt")
 ' (6) Service call text ("call_text") - Char (250)
 ' (7) Modified by ("modify_user") - char (20)
 ' (8) Scheduled To Time ("scheduled_to_time") - Char (5)
 ' (9) Requested Technician ("requested_tech") - char(30)
 ' (10) Call 1st Flag ("call_first") - Boolean
 ' (11) Call 1st phone number ("call_first_phone") - char (10)
 ' (12) Completed Time ("complete_time") - Char (5); military time (24 hour)
 ' (13) Pegasus Status ("pegasus_status") - char (10)
 ' (14) Pegasus Status Date/Time ("pegasus_status_date") Date & time
 ' (15) Targeted Arrival Date ("arrival_date") - Date & time
 ' (16) Targeted Arrival Time ("arrival_time") - char (5); military time (24 hour)
 ' (17) Assigned Technician ("technician_name") - char (30)

.NET Example

The following example shows how to return the service call details:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvCall As System.Data.DataSet
'
Set objService = CreateObject("EnergyAPI.cService")
```

```
‘  
objSrvCall =  
objService.GetServiceCallInfo(iCompanyID,lAccountNum,iServiceNum,cSysUser,lRetCode)  
‘  
If lRetCode <> 0 Then  
    ‘ Failure - Display error message  
Else  
    ‘ Success  
    For Each row In objSrvCall.Tables(0).Rows  
        Debug.Print “WO #:” & Row (0) & “Call Reason 1:” & Row (1), etc.  
    Next  
End If
```

cService.GetServiceHistory()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves service call history records for a given account number/service location

```
Public Function GetServiceHistory(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iServiceNum As Short, _
    ByVal p_dteFromDate As Date, _
    ByVal p_dteToDate As Date, _
    ByVal p_lMaxRows As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iServiceNum --- 0=All service locations, <> 0 Specific Service location
'          p_dteFromDate -- From Date
'          p_dteToDate -- To Date
'          p_lMaxRows -- Number of maximum number of deliveries to return from Newest to oldest delivery,
'                      Send 0 to return all selected history records
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Service location number ("service_num") -- Short integer
'                      (1)Date of service ("service_date") - Date
'                      (2)Work order number ("wo_number") - Long integer
'                      (3)Price ("price") - currency
'                      (4)Nature of call ("call_reason_1") - Char(40)
```

.NET Example

The following example shows how to retrieve service call history:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvHist As System.Data.DataSet
'
Set objService = CreateObject("EnergyAPI.cService")
'
objSrvHist =
objService.GetServiceHistory(iCompanyID,lAccountNum,iServiceNum,dteFromDate,
dteToDate, 0, cSysUser,lRetCode)
'
If lRetCode <> 0 Then
```

```
        ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvHist.Tables(0).Rows ()
        Debug.Print "Service #:" & Row (0) & "Service date:" & Row (1), etc.
    Next
End If
```

cService.GetCallReason()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of service call reasons for a given energy account number/service location/company ID.

```
Public Function GetCallReason (ByVal p_iCompanyID As Short, _
                                ByVal p_lAccountNum As Integer, _
                                ByVal p_iServiceNum As Short, _
                                ByVal p_sSysUser As String, _
                                ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iServiceNum --- Specific Service location
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows
'          (0) Call Reason ID ("reason") -- Short
'          (1) Call Reason Description ("description") – char(40)
'          (2) Calendar Flag ("calendar_schedule") – char(1) Y/N
'          (3) Priority ("priority") – char(10) "Critical","High","Medium","Low"
```



Call reasons are attached to service categories via the Service Category Maintenance in the Energy Client application. Therefore, the account must have a valid service category (1-255) at the service location level and the category must have one or more call reasons attached to it. Otherwise, no results will be returned.

.NET Example

The following example shows how to retrieve call reasons:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvRsn As System.Data.DataSet
'
Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvRsn = objService. GetCallReason (iCompanyID, lAccountNum, iServiceNum, cSysUser,
lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
```



```
Else
    ' Success
    For Each row In objSrvRsn.Tables(0).Rows
        Debug.Print "Reason #:" & Row (0) & "Reason Description:" & Row (1), etc.
    Next
End If
```

cService.GetDaySegment()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of day segments for a given energy company ID/Customer #/Service Location.

```
Public Function GetDaySegment(ByVal p_iCompanyID As Short, _
                               ByVal p_lAccountNum As Integer, _
                               ByVal p_iServiceNum As Short, _
                               ByVal p_sSysUser As String, _
                               ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iServiceNum --- Specific Service location
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows
'          (0) Day Segment ID ("day_segment_id") -- Short
'          (1) Day Segment Description ("description") – char (40)
'          (2) From Time ("from_time") – char (5)
'          (3) To Time ("to_time") – char (5)
```



When NT Calendar is Active

The day segments are retrieved from the Pegasus database. The day segments are defined at the company level. If no results are returned verify the existence of these records.

When Energy Scheduler is Active

The day segments are created/maintained by service centers via the Day Segment maintenance screen in the Energy Client Application. Service Centers are created/maintained via the Service Center maintenance screen in the Energy Client application. The account # must have a valid zone that is attached to a valid Service Center in order to get results. If no results are returned verify that the zone at the service location level is attached to a service center and that the service center has atleast one day segment defined.

.NET Example

The following example shows how to retrieve day segments:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvDaySeg As System.Data.DataSet
'
Set objService = CreateObject ("EnergyAPI.cService")
'
```

```
objSrvDaySeg = objService.GetDaySegment (iCompanyID, lAccountNum, iServiceNum,  
cSysUser, lRetCode)  
,
```

```
If lRetCode <> 0 Then
```

```
    ' Failure - Display error message
```

```
Else
```

```
    ' Success
```

```
    For Each row In objSrvDaySeg.Tables(0).Rows
```

```
        Debug.Print "Day Segment #:" & Row (0) & "Day Segment Description:" & Row  
        (1), etc.
```

```
    Next
```

```
Endif
```

cService.GetAvailableSlot()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves a collection of calendar scheduling slots for a given energy account, service location, call reason(s) and schedule date & time window.

```
Public Function GetAvailableSlot (ByVal p_iCompanyID As Short, _
                                   ByVal p_lAccountNum As Integer, _
                                   ByVal p_iServiceNum As Short, _
                                   ByVal p_sSysUser As String, _
                                   ByVal p_iCallReasonOne As Short, _
                                   ByVal p_iCallReasonTwo As Short, _
                                   ByVal p_iCallReasonThree As Short, _
                                   ByVal p_dteFromDate As Date, _
                                   ByVal p_sFromTime As String, _
                                   ByVal p_dteToDate As Date, _
                                   ByVal p_sToTime As String, _
                                   ByVal p_iDaysOfWeek As Short, _
                                   ByVal p_iMaxSlots As Short, _
                                   ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

'Accepts:

```
' p_iCompanyID -- The Energy Company ID
' p_lAccountNum -- The Account Number Without Check Digit
' p_iServiceNum --- Specific Service location
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_iCallReasonOne – first call reason (mandatory)
' p_iCallReasonTwo – 2nd call reason (0 if none)
' p_iCallReasonThree – 3rd call reason (0 if none)
' p_dteFromDate – Preferred “From Date”; must be greater than today
' p_sFromTime – Preferred “From Time”; military time (24hour)
' p_dteToDate – Preferred “To Date”; must be greater/equal From date
' p_sToTime – Preferred “To Time”; military time (24hour)
' p_iDaysOfWeek – bitmap corresponding to preferred days of the week
```



2^0 – Sunday (1), 2^1 – Monday (2), 2^2 – Tuesday (4), 2^3 – Wednesday (8)
2^4 – Thursday (16), 2^5 – Friday (32), 2^6 – Saturday (64)

Example: If days requested are Monday and Wednesday then p_iDaysOfWeek would equal 2^1+2^3 = 2+8=10

```
' p_iMaxSlots – maximum number of slots to return; 1-255
```



When NT Calendar is active the value for the above parameter should be between 1 and 10, as the NT Calendar will never return more than 10 available slots. Any value > 10 will be reset to 10 by the API.
When the Energy Scheduler is active the acceptable range will be between 1 and 255.

Return Values

'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number

'
' Return Type DataSet.Tables(0).Rows
' (0) Slot ID ("slot_id") – Short

***NOTE* “slot_id” this value has meaning only when the scheduling system is set to ‘Energy Scheduler’. Not applicable for ‘NT Calendar’.**

' (1) Schedule Date ("schedule_date") – Date
' (2) From Time ("earliest_time") – Char (5) 99:99
' (3) To Time ("latest_time") – Char (5) 99:99
' (4) # of Techs available/slot ("num_of_techs") – Short

.NET Example

The following example shows how to retrieve available calendar slots:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvAvSlot As System.Data.DataSet
'
Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvAvSlot = objService.GetAvailableSlot (iCompanyID, lAccountNum, iServiceNum,
cSysUser, 1, 0,0,"01/01/2004","10:00","01/03/2004","15:00",4, 5,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvAvSlot.Tables(0).Rows
        Debug.Print "Slot ID #:" & Row (0) & "Slot Description:" & Row (1), etc.
    Next
End If
```

cService.LogWorkOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Logs a service work order for a given account, service location, call reason and preferred date/time range (or selected scheduling slot). Also, if successful, returns a confirmation number (i.e. work order #) and scheduled date & time window

```
Public Function LogWorkOrder(ByVal p_iCompanyID As Short, _
                               ByVal p_lAccountNum As Integer, _
                               ByVal p_iServiceNum As Short, _
                               ByVal p_sSysUser As String, _
                               ByVal p_iCallReasonOne As Short, _
                               ByVal p_iCallReasonTwo As Short, _
                               ByVal p_iCallReasonThree As Short, _
                               ByVal p_sCallText As String, _
                               ByVal p_sLogText As String, _
                               ByVal p_sPerformToday as String, _
                               ByVal p_dteFromDate As Date, _
                               ByVal p_sFromTime As String, _
                               ByVal p_dteToDate As Date, _
                               ByVal p_sToTime As String, _
                               ByVal p_iDaysOfWeek As Short, _
                               ByVal p_sCallFirst String, _
                               ByVal p_sCallFirstPhone as String, _
                               ByVal p_lSlotID As Integer, _
                               ByVal p_bIsCleaningTicket as Boolean, _
                               ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

'Accepts:

```
' p_iCompanyID -- The Energy Company ID
' p_lAccountNum -- The Account Number Without Check Digit
' p_iServiceNum --- Specific Service location
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_iCallReasonOne – first call reason (mandatory)
' p_iCallReasonTwo – 2nd call reason (0 if none)
' p_iCallReasonThree – 3rd call reason (0 if none)
' p_sCallText – Call text
' p_sLogText – This text will be appended to basic Customer Log (Service WO template) text. This can
' be used to capture additional information (e.g. customer accepted diagnostic fee or chargeable
' warning). See section below describing Customer Log entry.
' p_sPerformToday – ‘Y’ immediate call to be done today, ‘N’ can be done in future
' p_dteFromDate – Preferred “From Date”; must be greater than today
' p_sFromTime – Preferred “From Time”; military time (24 hour)
' p_dteToDate – Preferred “To Date”; must be greater/equal From date
' p_sToTime – Preferred “To Time”; military time (24 hour)
' p_iDaysOfWeek – Bitmap corresponding to preferred days of the week
```



2^0 – Sunday (1), 2^1 – Monday (2), 2^2 – Tuesday (4), 2^3 – Wednesday (8)

2^4 - Thursday (16), 2^5 - Friday (32), 2^6 - Saturday (64)

Example: If days requested are Monday and Wednesday then $p\_iDaysOfWeek$ would equal $2^1 + 2^3 = 2 + 8 = 10$

' p_sCallFirst - 'Y' means customer must be called first prior to arrival
' p_sCallFirstPhone - phone number including area code
' p_lSlotID - Uniquely identifies the appointment time. A non-zero value would mean that a call to GetAvailableSlot was issued prior. If 0 the system will use the date/time ranges to re-schedule the Work Order. ***NOTE* p_lSlotID is only applicable when the scheduling system is 'Energy Scheduler'. Ignored when 'NT Calendar' is active (ie: send 0).**
' p_bIsCleaningTicket - True if call is flagged as a cleaning ticket, False if not.

Return Values

Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number

' Return Type DataSet.Tables(0).Rows
' (0) Confirmation # ("wo_number") - Long Integer
' (1) Schedule Date ("schedule_date") - Date
' (2) From Time ("from_time") - Char (5) 99:99
' (3) To Time ("to_time") - Char (5) 99:99
' (4) Booking Indicator ("booking_indicator") - Char (1) 'C'alendar booked, 'U'nbooked
' (5) Pegasus Status ("pegasus_status") - Char (10)
' (6) Pegasus Status date/time ("pegasus_status_date") - Date & Time
' (7) Targeted Arrival Date ("arrival_date") - Date & Time
' (8) Targeted Arrival Time ("arrival_time") - Char (5)
' ***NOTE* when NT Calendar is active, the targeted Arrival time will simply be the "To Time" as defined in the system for the day segment that was booked.**
' (9) Technician Name ("technician_name") - Char (30)

ATTENTION it should be noted that the From/To dates and From/To Times are 'preferred'. Therefore there is no guarantee that the requested date/time will be booked. The system will book/reserve the first most suitable available slot (date/time). The Dataset returned will ultimately have the confirmed date/time. Even if GetAvailableSlots() is called prior and a slot is selected, there is no guarantee that once LogWorkOrder() is called that the selected date/time is still available and will be booked. For this reason, a call to LogWorkOrder() can be made without having to first call GetAvailableSlots()

.NET Example

The following example shows how to logging a service work order:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvLogWO As Object
Dim sCallFirst as String
Dim sCallFirstPhone as String
'
sCallFirst="N"
sCallFirstPhone="5165558989"
Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvLogWO = objService. LogWorkOrder (iCompanyID, lAccountNum, iServiceNum,
cSysUser,1, 0,0,"this is the call text","this is the log
text","N","01/01/2004","10:00","01/03/2004","15:00",4,sCallFirst,sCallFirstPhone,0,False,lRet
Code)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvLogWO.Tables(0).Rows
        Debug.Print "Confirmation #:" & Row (0) & "Schedule date:" & Row (1), etc.
    Next
End If
```


cService.LogWorkOrderNTC()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Logs a service work order for a given account, service location, call reason and preferred date/time range (or selected scheduling slot). Also, if successful, returns a confirmation number (i.e. work order #) and scheduled date & time window.

NOTE This method **should be used in place of** LogWorkOrder() when the scheduling system being used is NT Calendar and you **do not** want the system (API) to book the next suitable slot if the requested date/time is not available but rather return code 181 (unavailable slots).

Public Function **LogWorkOrderNTC**(ByVal pCompanyID As Short
 , ByVal pAccountNum As Integer
 , ByVal pServiceNum As Short
 , ByVal pSysUser As String
 , ByVal pWorkOrderData As [DCEnergyAPI.Service.WorkOrderData](#)
 , ByRef pErrorCode As Integer) As DCEnergyAPI.Service.DSServiceWorkOrder

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Integer (32 bit Integer)	Account # (without checkdigit)
	pServiceNum	Short (16 bit integer)	ActionCode.Code (as retrieved by LoadCreditActionCodes)
	pSysUser	String	Calling program (IVRUser/WebUser)
	pWorkOrderData	DCEnergyAPI.Service.WorkOrderData	Data structure of type WorkOrderData

Return Values

Returns: pErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number

Return Type DataSet.Tables(0).Rows

- (0) Confirmation # ("wo_number") – Long Integer
- (1) Schedule Date ("schedule_date") – Date
- (2) From Time ("from_time") – Char (5) 99:99
- (3) To Time ("to_time") – Char (5) 99:99
- (4) Booking Indicator ("booking_indicator") – Char (1) 'C'alendar booked, 'U'nbooked
- (5) Pegasus Status ("pegasus_status") – Char (10)
- (6) Pegasus Status date/time ("pegasus_status_date") – Date & Time
- (7) Targeted Arrival Date ("arrival_date") – Date & Time
- (8) Targeted Arrival Time ("arrival_time") – Char (5)

NOTE when NT Calendar is active, the targeted Arrival time will simply be the "To Time" as defined in the system for the day segment that was booked.

(9) Technician Name ("technician_name") – Char (30)

DCEnergyAPI.Service.WorkOrderData

field name	Description
int WorkOrderNumber	0: when calling LogWorkOrderNTC Valid wo#: when calling RescheduleWorkOrderNTC
short CallReasonOne	1st call reason (mandatory). Must be 0 when calling RescheduleWorkOrderNTC
short CallReasonTwo	2nd call reason (0 if none) . Must be 0 when calling RescheduleWorkOrderNTC
short CallReasonThree	3rd call reason (0 if none) . Must be 0 when calling RescheduleWorkOrderNTC
string PerformToday	'Y' immediate call to be done today, 'N' can be done in future
DateTime FromDate	Preferred "From Date"; must be greater than today
DateTime ToDate	Preferred "To Date"; must be greater/equal From date
string FromTime	Preferred "From Time"; military time (24 hour)
string ToTime	Preferred "To Time"; military time (24 hour)
short DaysOfWeek	Bitmap corresponding to preferred days of the week 2^0 - Sunday (1), 2^1 - Monday (2), 2^2 - Tuesday (4), 2^3 - Wednesday (8), 2^4 - Thursday (16), 2^5 - Friday (32), 2^6 - Saturday (64)
bool IsCleaningTicket	True if call is flagged as a cleaning ticket, False if not.
DCEnergy.Service.CallInformationData CallInformation	Data structure of type CallInformationData

DCEnergyAPI.Service.CallInformationData

field name	Description
string CallText	Call notes (Applicable with <i>LogWorkOrderNTC()</i> , <i>RescheduleWorkOrderNTC()</i> + <i>UpdateCallInformation()</i>)
string LogText	This text will be appended to basic Customer Log (Service WO template) text. This can be used to capture additional information (e.g. customer accepted diagnostic fee or chargeable warning). See section below describing Customer Log entry. (Applicable with <i>LogWorkOrderNTC()</i> , <i>RescheduleWorkOrderNTC()</i> + <i>UpdateCallInformation()</i>)
string CallFirst	'Y' means customer must be called first prior to arrival (Applicable with <i>LogWorkOrderNTC()</i> , <i>RescheduleWorkOrderNTC()</i> + <i>UpdateCallInformation()</i>)
string CallFirstPhone	phone number including area code (Applicable with <i>LogWorkOrderNTC()</i> , <i>RescheduleWorkOrderNTC()</i> + <i>UpdateCallInformation()</i>)
int PreferredTechnician	Preferred technician (EmployeeID as returned from lookup <i>cService.LoadTechnicians()</i>) or 0. (Applicable with <i>LogWorkOrderNTC()</i> , <i>RescheduleWorkOrderNTC()</i> + <i>UpdateCallInformation()</i>)

XML Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LogWorkOrderNTC>
      <add:pCompanyID>1600</add:pCompanyID>
      <add:pAccountNum>100</add:pAccountNum>
      <add:pServiceNum>1</add:pServiceNum>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pWorkOrderData>
        <add:WorkOrderNumber>0</add:WorkOrderNumber>
        <add:CallReasonOne>91</add:CallReasonOne>
        <add:CallReasonTwo>0</add:CallReasonTwo>
        <add:CallReasonThree>0</add:CallReasonThree>
        <add:PerformToday>N</add:PerformToday>
        <add:FromDate>2017-11-10</add:FromDate>
        <add:ToDate>2017-11-10</add:ToDate>
        <add:FromTime>0800</add:FromTime>
        <add:ToTime>1159</add:ToTime>
        <add:DaysOfWeek>0</add:DaysOfWeek>
        <add:IsCleaningTicket>0</add:IsCleaningTicket>
        <add:CallInformation>
          <add:CallText>THIS IS CALL TEXT FROM API</add:CallText>
          <add:LogText>THIS IS LOG TEXT FROM API</add:LogText>
          <add:CallFirst>N</add:CallFirst>
          <add:CallFirstPhone></add:CallFirstPhone>
          <add: PreferredTechnician >123</add: PreferredTechnician>
        </add:CallInformation>
      </add:pWorkOrderData>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LogWorkOrderNTC>
  </soap:Body>
</soap:Envelope>
```

cService.RescheduleWorkOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window.

```
Public Function RescheduleWorkOrder (ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iServiceNum As Short, _
    ByVal p_sSysUser As String, _
    ByVal p_lWorkOrderNumber As Integer, _
    ByVal p_sLogText As String, _
    ByVal p_sPerformToday As String, _
    ByVal p_dteFromDate As Date, _
    ByVal p_sFromTime As String, _
    ByVal p_dteToDate As Date, _
    ByVal p_sToTime As String, _
    ByVal p_iDaysOfWeek As Short, _
    ByVal p_lSlotID As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

'Accepts:

```
' p_iCompanyID -- The Energy Company ID
' p_lAccountNum -- The Account Number Without Check Digit
' p_iServiceNum --- Specific Service location
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_lWorkOrderNumber – service W/O #
' p_sLogText – This text will be appended to basic Customer Log (Service WO template) text. This can
' be used to capture additional information (e.g. customer accepted diagnostic fee or chargeable
' warning). See section below describing Customer Log entry.
' p_sPerformToday – ‘Y’ immediate call to be done today, ‘N’ can be done in future
' p_dteFromDate – Preferred “From Date”; must be greater than today
' p_sFromTime – Preferred “From Time”; military time (24 hour)
' p_dteToDate – Preferred “To Date”; must be greater/equal From date
' p_sToTime – Preferred “To Time”; military time (24 hour)
' p_iDaysOfWeek – Bitmap corresponding to preferred days of the week
```



2^0 – Sunday (1), 2^1 – Monday (2), 2^2 – Tuesday (4), 2^3 – Wednesday (8)
2^4 – Thursday (16), 2^5 – Friday (32), 2^6 – Saturday (64)

Example: If days requested are Monday and Wednesday then p_iDaysOfWeek would equal $2^1 + 2^3 = 2 + 8 = 10$

' p_lSlotID – Uniquely identifies the appointment time. A non-zero value would mean that a call to GetAvailableSlot was issued prior. If 0 the system will use the date/time ranges to schedule the Work Order.

***NOTE* p_lSlotID is only applicable when the scheduling system is ‘Energy Scheduler’. Ignored when ‘NT Calendar’ is active (ie: send 0).**

Return Values

Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number

Return Type DataSet.Tables(0).Rows

- (0) Confirmation # ("wo_number") – Long Integer
- (1) Schedule Date ("schedule_date") – Date
- (2) From Time ("from_time") – Date & Time (extract time)
- (3) To Time ("to_time") – Date & Time (extract time)
- (4) Booking Indicator ("booking_indicator")– Char (1) 'C'alendar booked, 'U'nbooked
- (5) Pegasus Status ("pegasus_status") –Char (10)
- (6) Pegasus Status date/time ("pegasus_status_date") – Date & Time
- (7) Targeted Arrival Date ("arrival_date") – Date & Time
- (8) Targeted Arrival Time ("arrival_time") – Char (5)

***NOTE* when NT Calendar is active, the targeted Arrival time will simply be the "To Time" as defined in the system for the day segment that was booked.**

- (9) Technician Name ("technician_name") – Char (30)

ATTENTION it should be noted that the From/To dates and From/To Times are 'preferred'. Therefore there is no guarantee that the requested date/time will be booked. The system will book/reserve the first most suitable available slot (date/time). The Dataset returned will ultimately have the confirmed date/time. Even if GetAvailableSlots() is called prior and a slot is selected, there is no guarantee that once RescheduleWorkOrder() is called that the selected date/time is still available and will be booked. For this reason, a call to RescheduleWorkOrder() can be made without having to first call GetAvailableSlots()

.NET Example

The following example shows how to reschedule a service work order:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvReschWO As System.Data.DataSet
Dim sCallFirst as String
Dim sCallFirstPhone as String
Dim lSlotID as Integer
'
sCallFirst="N"
sCallFirstPhone="5165558989"
lSlotID =0
Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvReschWO = objService. RescheduleWorkOrder (iCompanyID, lAccountNum,
iServiceNum, cSysUser,1, 0,0,"this is the LOG text","this is the log
text","N","01/01/2004","10:00","01/03/2004","15:00",4,lSlotID,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvReschWO.Tables(0).Rows
        Debug.Print "Confirmation #:" & Row (0) & "Schedule date:" & Row (1), etc.
    Next
End If
```

cService.RescheduleWorkOrderWithNotes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window.

```
Public Function RescheduleWorkOrderWithNotes (ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iServiceNum As Short, _
    ByVal p_sSysUser As String, _
    ByVal p_lWorkOrderNumber As Integer, _
    ByVal p_sLogText As String, _
    ByVal p_sPerformToday As String, _
    ByVal p_dteFromDate As Date, _
    ByVal p_sFromTime As String, _
    ByVal p_dteToDate As Date, _
    ByVal p_sToTime As String, _
    ByVal p_iDaysOfWeek As Short, _
    ByVal p_lSlotID As Integer, _
    ByVal pCallText As String, _
    ByRef p_lErrorCode As Integer, _
    ByRef p_sErrMsg As String) As System.Data.DataSet
```

Input Parameters

'Accepts:

```
' p_iCompanyID -- The Energy Company ID
' p_lAccountNum -- The Account Number Without Check Digit
' p_iServiceNum --- Specific Service location
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_lWorkOrderNumber -- service W/O #
' p_sLogText -- This text will be appended to basic Customer Log (Service WO template) text. This can
' be used to capture additional information (e.g. customer accepted diagnostic fee or chargeable
' warning). See section below describing Customer Log entry.
' p_sPerformToday -- 'Y' immediate call to be done today, 'N' can be done in future
' p_dteFromDate -- Preferred "From Date"; must be greater than today
' p_sFromTime -- Preferred "From Time"; military time (24 hour)
' p_dteToDate -- Preferred "To Date"; must be greater/equal From date
' p_sToTime -- Preferred "To Time"; military time (24 hour)
' p_iDaysOfWeek -- Bitmap corresponding to preferred days of the week
```



2^0 - Sunday (1), 2^1 - Monday (2), 2^2 - Tuesday (4), 2^3 - Wednesday (8)
2^4 - Thursday (16), 2^5 - Friday (32), 2^6 - Saturday (64)

Example: If days requested are Monday and Wednesday then p_iDaysOfWeek would equal $2^1 + 2^3 = 2 + 8 = 10$

' p_lSlotID -- Uniquely identifies the appointment time. A non-zero value would mean that a call to GetAvailableSlot was issued prior. If 0 the system will use the date/time ranges to schedule the Work Order.

***NOTE* p_IslotID is only applicable when the scheduling system is 'Energy Scheduler'. Ignored when 'NT Calendar' is active (ie: send 0).**

' pCallText – Call text

Return Values

Returns: p_LErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number

' p_sErrCode (ByRef) – Error message under some conditions

'

' Return Type DataSet.Tables(0).Rows

'

' (0) Confirmation # ("wo_number") – Long Integer

'

' (1) Schedule Date ("schedule_date") – Date

'

' (2) From Time ("from_time") – Date & Time (extract time)

'

' (3) To Time ("to_time") – Date & Time (extract time)

'

' (4) Booking Indicator ("booking_indicator") – Char (1) 'C'alendar booked, 'U'nbooked

'

' (5) Pegasus Status ("pegasus_status") – Char (10)

'

' (6) Pegasus Status date/time ("pegasus_status_date") – Date & Time

'

' (7) Targeted Arrival Date ("arrival_date") – Date & Time

'

' (8) Targeted Arrival Time ("arrival_time") – Char (5)

'

' ***NOTE* when NT Calendar is active, the targeted Arrival time will simply be the "To Time" as defined in the system for the day segment that was booked.**

'

' (9) Technician Name ("technician_name") – Char (30)

ATTENTION it should be noted that the From/To dates and From/To Times are 'preferred'. Therefore there is no guarantee that the requested date/time will be booked. The system will book/reserve the **first** most suitable available slot (date/time). The Dataset returned will ultimately have the confirmed date/time. Even if GetAvailableSlots() is called prior and a slot is selected, there is no guarantee that once RescheduleWorkOrderWithNotes() is called that the selected date/time is still available and will be booked. For this reason, a call to RescheduleWorkOrderWithNotes() can be made without having to first call GetAvailableSlots()

.NET Example

The following example shows how to reschedule a service work order:

```
Dim objService as Object
Dim lRetCode as Integer
Dim sRetMsg As String = ""
Dim objSrvReschWO As System.Data.DataSet
Dim sCallFirst as String
Dim sCallFirstPhone as String
Dim lSlotID as Integer
'
sCallFirst="N"
sCallFirstPhone="5165558989"
lSlotID =0
Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvReschWO = objService. RescheduleWorkOrderWithNotes (iCompanyID, lAccountNum,
iServiceNum, cSysUser,1, 0,0,"this is the LOG text","this is the log
text","N","01/01/2004","10:00","01/03/2004","15:00",4,lSlotID,"updated call notes",lRetCode,
sRetMsg)
'
If lRetCode <> 0 OrElse sRetMsg <> "" Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvReschWO.Tables(0).Rows
        Debug.Print "Confirmation #:" & Row (0) & "Schedule date:" & Row (1), etc.
    Next
End If
```

cService.RescheduleWorkOrderNTC()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Reschedules a service work order for a given account, service location, work order and preferred date/time range. Also, if successful, returns a confirmation number (i.e. work order #) and revised scheduled date & time window.

NOTE This method **should be used in place of** RescheduleWorkOrder() & RescheduleWorkOrderWithNotes() when the scheduling system being used is NT Calendar and you **do not** want the system (API) to book the next suitable slot if the requested date/time is not available but rather return code 181 (unavailable slots).

Public Function **RescheduleWorkOrderNTC**(ByVal pCompanyID As Short
 , ByVal pAccountNum As Integer
 , ByVal pServiceNum As Short
 , ByVal pSysUser As String
 , ByVal pWorkOrderData As [DCEnergyAPI.Service.WorkOrderData](#)
 , ByRef pErrorCode As Integer
 , ByRef pErrMsg as String) As DCEnergyAPI.Service.DSServiceWorkOrder

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Integer (32 bit Integer)	Account # (without checkdigit)
	pServiceNum	Short (16 bit integer)	ActionCode.Code (as retrieved by LoadCreditActionCodes)
	pSysUser	String	Calling program (IVRUser/WebUser)
	pWorkOrderData	DCEnergyAPI.Service.WorkOrderData	

Return Values

Returns: pErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
 , pErrMsg (ByRef) – Error message under some conditions

,
 , Return Type DataSet.Tables(0).Rows
 , (0) Confirmation # ("wo_number") – Long Integer
 , (1) Schedule Date ("schedule_date") – Date
 , (2) From Time ("from_time") – Char (5) 99:99
 , (3) To Time ("to_time") – Char (5) 99:99
 , (4) Booking Indicator ("booking_indicator")– Char (1) 'C'alendar booked, 'U'nbooked
 , (5) Pegasus Status ("pegasus_status") –Char (10)

- ' (6) Pegasus Status date/time ("pegasus_status_date") – Date & Time
- ' (7) Targeted Arrival Date ("arrival_date") – Date & Time
- ' (8) Targeted Arrival Time ("arrival_time") – Char (5)

***NOTE* when NT Calendar is active, the targeted Arrival time will simply be the "To Time" as defined in the system for the day segment that was booked.**

- ' (9) Technician Name ("technician_name") – Char (30)

XML Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:RescheduleWorkOrderNTC>
      <add:pCompanyID>1600</add:pCompanyID>
      <add:pAccountNum>100</add:pAccountNum>
      <add:pServiceNum>1</add:pServiceNum>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pWorkOrderData>
        <add:WorkOrderNumber>8153</add:WorkOrderNumber>
        <add:CallReasonOne>0</add:CallReasonOne>
        <add:CallReasonTwo>0</add:CallReasonTwo>
        <add:CallReasonThree>0</add:CallReasonThree>
        <add:PerformToday>N</add:PerformToday>
        <add:FromDate>2017-11-10</add:FromDate>
        <add:ToDate>2017-11-10</add:ToDate>
        <add:FromTime>0800</add:FromTime>
        <add:ToTime>1159</add:ToTime>
        <add:DaysOfWeek>0</add:DaysOfWeek>
        <add:IsCleaningTicket>0</add:IsCleaningTicket>
        <add:CallInformation>
          <add:CallText>THIS IS THE CALL TEXT FROM API</add:CallText>
          <add:LogText>THIS IS LOG TEXT FROM API</add:LogText>
          <add:CallFirst>N</add:CallFirst>
          <add:CallFirstPhone></add:CallFirstPhone>
          <add:PreferredTechnician>0</add:PreferredTechnician >
        </add:CallInformation>
      </add:pWorkOrderData>
      <add:pErrorCode>0</add:pErrorCode>
      <add:pErrMsg></add:pErrMsg>
    </add:RescheduleWorkOrderNTC>
  </soap:Body>
</soap:Envelope>
```

cService.CancelWorkOrder()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Cancels a service work order for a given account, service location, work order.

```
Public Function CancelWorkOrder (ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iServiceNum As Short, _
    ByVal p_sSysUser As String, _
    ByVal p_lWorkOrderNumber as Integer, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts:
'      p_iCompanyID -- The Energy Company ID
'      p_lAccountNum -- The Account Number Without Check Digit
'      p_iServiceNum --- Specific Service location
'      p_sSysUser - Calling program (IVRUser/WebUser)
'      p_lWorkOrderNumber – service W/O #
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'      Return Type Integer (not currently used; future use)
```

.NET Example

The following example shows how to rescheduling a service work order:

```
Dim objService as Object
Dim lRetCode as Integer
Dim lRet as Integer
Dim lWorkOrder as Integer

lWorkOrder =12345
Set objService = CreateObject ("EnergyAPI.cService")
lRet = objService. CancelWorkOrder (iCompanyID, lAccountNum, iServiceNum, cSysUser,
lWorkOrder, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    MsgBox "Work Order 12345 was successfully canceled."
End If
```

cService.GetServiceWaitTime()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Provides a mechanism to retrieve call wait time and service notes information.

```
Public Function GetServiceWaitTime (ByVal p_iCompanyID As Short, _
    ByVal p_iCallReason As Short, _
    ByVal p_sZone As String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts:
'      p_iCompanyID -- The Energy Company ID
'      p_iCallReason -- Call Reason
'      p_sZone --- Service Zone
'      p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'      Return Type DataSet.Tables(0).Rows
'      (0) Call Reason ("call_reason") – Short
'      (1) Service Zone ("zone") – Char(10)
'      (6) Service Notes ("service_notes") – Char(255)
'      (2) From Date ("from_date") – Date
'      (3) From Time ("from_time") – Time (24hour;military)
'      (4) To Date ("to_date") – Date
'      (5) To Time ("to_time") – Time (24hour;military)
```

.NET Example

The following example shows how to access service wait time information:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvWaitTime As System.Data.DataSet
Dim iCallReason as Short
Dim sZone as string

iCallReason =91
sZone="12345"

Set objService = CreateObject ("EnergyAPI.cService")
'
objSrvWaitTime = objService. GetServiceWaitTime (iCompanyID,
iCallReason,sZone,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
```

```
        ' Failure - Display error message
Else
    ' Success
    For Each row In objSrvWaitTime.Tables(0).Rows ()
        Debug.Print "Call Reason #:" & Row (0) & "From Date: " & Row (2) & "From
        Time: " & Row (3) & "To Date:" & Row (4) & " To Time: " & Row (5)
    Next
End If
```

cService.SaveServiceWaitTime()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Provides a mechanism for saving call wait time and service notes to the Energy table by call reason and zone.

```
Public Function SaveServiceWaitTime (ByVal p_iCompanyID As Short, _
                                     ByVal p_iCallReason As Short, _
                                     ByVal p_sZone As String, _
                                     ByVal p_dteFromDate as Date, _
                                     ByVal p_sFromTime as String, _
                                     ByVal p_dteToDate as Date, _
                                     ByVal p_sToTime as String, _
                                     ByVal p_sServiceNotes as string, _
                                     ByVal p_sSysUser As String, _
                                     ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts:
'      p_iCompanyID -- The Energy Company ID
'      p_iCallReason -- Call Reason
'      p_sZone --- Service Zone
'      p_dteFromDate -- From Date
'      p_sFromTime -- From Time (AM/PM format or 24hour)
'      p_dteToDate -- To Date
'      p_sToTime -- To Time (AM/PM format or 24hour)
'      p_sServiceNotes -- notes
'      p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'      Return Type Integer: Unused (possible future use)
```

.NET Example

The following example shows how to save service call wait information:

```
Dim objService as Object
Dim lRetCode as Integer
Dim objSrvWaitTime As System.Data.DataSet
Dim iCallReason as Short
Dim sZone as string

iCallReason =91
sZone="12345"
```



```
Set objService = CreateObject ("EnergyAPI.cService")
'
Call objService. SaveServiceWaitTime (iCompanyID, iCallReason,
sZone,"05/19/2004","08:00PM","05/20/2004","10:00PM","This is my call Note",
cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success.
    msgbox "Service Wait Time information successfully saved."
Endif
```

cService.DeleteServiceWaitTime()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Provides a mechanism for deleting call wait time information by call reason and zone.

```
Public Function DeleteServiceWaitTime (ByVal p_iCompanyID As Short, _
                                         ByVal p_iCallReason As Short,
                                         ByVal p_sZone As String, _
                                         ByVal p_sSysUser As String,
                                         ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts:
'      p_iCompanyID -- The Energy Company ID
'      p_iCallReason -- Call Reason
'      p_sZone --- Service Zone
'      p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful: <> 0 Error Number
'
'      Return Type Integer: Unused (possible future use)
```

.NET Example

The following example shows how to save service call wait information:

```
Dim objService as Object
Dim lRetCode as Integer
Dim iCallReason as Short
Dim sZone as string

iCallReason =91
sZone="12345"

Set objService = CreateObject ("EnergyAPI.cService")
'
Call objService. DeleteServiceWaitTime (iCompanyID, iCallReason, sZone,
cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success.
    msgbox "Service Wait Time information successfully deleted."
Endif
```

cService.UpdateCallInformation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Provides a mechanism for updating information on an existing call/work order.

Public Function **UpdateCallInformation**(ByVal pCompanyID As Short,
 ByVal pAccountNum As Integer,
 ByVal pServiceNum As Integer,
 ByVal pWorkOrderNumber As Integer,
 ByVal pSysUser As String,
 ByVal pCallInformation As [DCEnergyAPI.Service.CallInformationData](#)) As Integer

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Integer	E3 Account number without check digit
	pServiceNum	Short (16 bit integer)	Service location #
	pWorkOrderNumber	Integer	Valid w/o#
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCallInformation	DCEnergyAPI.Service.CallInformationData	Call information to update
Return Values			
		Integer	0 if successful, <> 0 Error Number

XML Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:UpdateCallInformation>
      <add:pCompanyID>1600</add:pCompanyID>
      <add:pAccountNum>100</add:pAccountNum>
      <add:pServiceNum>1</add:pServiceNum>
      <add:pWorkOrderNumber>8153</add:pWorkOrderNumber>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCallInformation>
        <add:CallText>Call Notes are entered here</add:CallText>
        <add:LogText>Log text for customer log entry</add:LogText>
      </add:pCallInformation>
    </add:UpdateCallInformation>
  </soap:Body>
</soap:Envelope>
```

```
<add:CallFirst>Y</add:CallFirst>
<add:CallFirstPhone>9998887777</add:CallFirstPhone>
<add:PreferredTechnician>12345</add:PreferredTechnician>
</add:pCallInformation>
</add:UpdateCallInformation>
</soap:Body>
</soap:Envelope>
```

NOTE it is important to send the current data for any item you are not intending to modify when calling this procedure!!!

cService.LoadTechnicians()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Service Order Business Edition**

Description

Provides a mechanism for loading active technicians.

Public Function **LoadTechnicians**(ByVal pCompanyID As Short,
ByVal pSysUser As String,
ByRef pErrorCode As Integer) As [DCEnergyAPI.Service.TechnicianInformation](#)()

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Integer	0 if successful, <> 0 Error Number
Return Values			
		DCEnergyAPI.Service.TechnicianInformation []	Array of type TechnicianInformation .

DCEnergyAPI.Service.TechnicianInformation

field name	Description
int EmployeeID	Unique employee #
string TechnicianName	Full name of technician

XML Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadTechnicians>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadTechnicians>
  </soap:Body>
</soap:Envelope>
```

CTransaction Class

Overview

This section describes the methods that enable the programmer to retrieve AR transaction records and accept payments entered by the end-user. The following methods are covered in this section:

- `GetARTransactions()`: retrieves a collection of transactions for a given criteria set
- `AcceptPayment()`: accepts payments entered by the user
- `GetNewUploadTransactionDataSet()`: returns empty data structure that can be used with the `UploadTransactions` call.
- `UploadTransactions()`: Allows transactions to be loaded into Energy – will not calculate taxes or do any reprojection.

cTransaction.GetARTransactions() / GetARTransactionsWithNoCreditActions()

- License Required: SmartConnect API Customer Look-up Business Edition**

Important remark: GetARTransactions() method will include credit actions that are transactions with posting codes greater than the maximum credit posting code; you can use GetARTransactionsWithNoCreditActions() method to exclude the credit actions.

Description

Retrieves AR transaction records for a given company ID/account number.

```
Public Function GetARTransactions(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iServiceNum As Short, _
    ByVal p_iLocationNum As Short, _
    ByVal p_dteFromDate As Short, _
    ByVal p_dteToDate As Short, _
    ByVal p_lMaxRows As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
' p_iCompanyID -- The Energy Company ID
' p_iTankNum --- 0= All tanks , <> 0 Specific tank number (delivery location)
' p_iServiceNum --- 0 = All Service location, 0<> Specific service location
' p_iLocation --- 0 = All Wdms locations, <> 0 Specific Wdms location
' p_dteFromDate -- From event date
' p_dteToDate -- To event date
' p_lMaxRows -- Maximum number of transactions - 0 = All selected transactions
' p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
' Return Type DataSet.Tables(0).Rows (0) Reference number ("reference_num") -- Long integer
' (1) Event date ("event_date") -- Date
' (2) Posting date ("posting_date") -- Date
' (3) Posting code ("posting_code") -- Short integer
' (4) Posting code description ("post_code_desc") -- Char(30)
' (5) Number of units ("units") --- Number with 2 decimals
' (6) Total amount ("net_amount") -- Currency
```

i We return payments (credits) as debits. The only way to tell the difference is from the posting code. You will have to get the maximum debit and the maximum credit codes from the client.

From 1 to maxDebit is a debit
 From maxDebit + 1 to maxCredit is a credit
 From maxCredit + 1 to the end are to be ignored.

```

'          (7) Degree day ("degree_day") -- Long integer
'          (8) Driver number ("driver_num") -- Short integer
'          (9) Driver name ("driver_name") -- Char(50)
'          (10) Truck number ("truck_num") -- Short integer
'          (11) Truck name ("truck_name") -- Char(50)
'          (12) Due date ("due_by_date") -- Date
'          (13) Has Document Flag ("has_documents Boolean") -- Bool
'          (14) Transaction Key ("trx_unique_key") -- Integer
'          (15) Tank Location ("tank_num") -- Short Integer
'          (16) Service Location ("service_num") -- Short Integer
'          (17) Location ("location_num") -- Short Integer
'          (18) In Tank Units (in_tank_units) -- Decimal
'          (19) Percent Fill (percent_fill) -- Short Integer
'          (20) Open Amount (open_amt) -- Decimal
'          (21) PO Number (po_number) -- String
'          (22) Keyoff Flag (keyoff_flag) -- String (Y/N)
'          (23) Invoice number (invoice_num) 32-bit Integer

```

Return code	Description
96	Unknown Error occurred.

.NET Example

The following example shows how to retrieve AR transaction records:

```

Dim objTransaction as Object
Dim lRetCode as Integer
Dim objTrxHist As System.Data.DataSet
'
Set objTransaction = CreateObject("EnergyAPI.cTransaction")
'
objTrxHist =
objTransaction.GetARTransactions(iCompanyID,lAccountNum,iTankNum,0,0,dteFromDate,
dteToDate, 0, cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objTrxHist.Tables(0).Rows
        Debug.Print "Reference #:" & Row (0) & "Event date:" & Row (1), etc.
    Next
End If

```

XML Example

The following example shows how to retrieve AR transaction records:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>

```



```
<add:GetARTransactions>
  <add:p_iCompanyID>2</add:p_iCompanyID>
  <add:p_lAccountNum>2</add:p_lAccountNum>
  <add:p_iTankNum>0</add:p_iTankNum>
  <add:p_iServiceNum>0</add:p_iServiceNum>
  <add:p_iLocationNum>0</add:p_iLocationNum>
  <add:p_dteFromDate>1990-01-01</add:p_dteFromDate>
  <add:p_dteToDate>2010-12-31</add:p_dteToDate>
  <add:p_lMaxRows>50</add:p_lMaxRows>
  <add:p_sSysUser>WebUser</add:p_sSysUser>
  <add:p_lErrorCode>0</add:p_lErrorCode>
</add:GetARTransactions>
</soap:Body>
</soap:Envelope>
```

cTransaction.GetARTransactionsPPD()

License Required: SmartConnect API Prompt Pay Discount information

Description

Retrieves AR transaction records including prompt pay discount amount and due date for a given company ID/account number.

```
Public Function GetARTransactionsPPD(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iTankNum As Short, _
    ByVal p_iServiceNum As Short, _
    ByVal p_iLocationNum As Short, _
    ByVal p_dteFromDate As Short, _
    ByVal p_dteToDate As Short, _
    ByVal p_lMaxRows As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'         p_iCompanyID -- The Energy Company ID
'         p_iTankNum --- 0= All tanks , <> 0 Specific tank number (delivery location)
'         p_iServiceNum --- 0 = All Service location, 0<> Specific service location
'         p_iLocation --- 0 = All Wdms locations, <> 0 Specific Wdms location
'         p_dteFromDate -- From event date
'         p_dteToDate -- To event date
'         p_lMaxRows -- Mximum number of transactions - 0 = All selected transactions
'         p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'         Return Type DataSet.Tables(0).Rows (0) Reference number ("reference_num") -- Long integer
'         (1) Event date ("event_date") -- Date
'         (2) Posting date ("posting_date") -- Date
'         (3) Posting code ("posting_code") -- Short integer
'         (4) Posting code description ("post_code_desc") -- Char(30)
'         (5) Number of units ("units") --- Number with 2 decimals
'         (6) Total amount ("net_amount") -- Currency
```

i We return payments (credits) as debits. The only way to tell the difference is from the posting code. You will have to get the maximum debit and the maximum credit codes from the client.

From 1 to to maxDebit is a debit

From maxDebit + 1 to maxCredit is a credit

```
'         (7) Degree day ("degree_day") -- Long integer
'         (8) Driver number ("driver_num") -- Short integer
'         (9) Driver name ("driver_name") -- Char(50)
'         (10) Truck number ("truck_num") -- Short integer
'         (11) Truck name ("truck_name") -- Char(50)
'         (12) Due date ("due_by_date") -- Date
```

- ' (13) Has Document Flag ("has_documents Boolean") – Bool
- ' (14) Transaction Key ("trx_unique_key") – Integer
- ' (15) Tank Location ("tank_num") -- Short Integer
- ' (16) Service Location ("service_num") -- Short Integer
- ' (17) Location ("location_num") -- Short Integer
- ' (18) In Tank Units (in_tank_units) -- Decimal
- ' (19) Percent Fill (percent_fill) -- Short Integer
- ' (20) Open Amount (open_amt) -- Decimal
- ' (21) PO Number (po_number) – String
- ' (22) Keyoff Flag (keyoff_flag) -- String (Y/N)
- ' (23) Invoice number (invoice_num) 32-bit Integer
- ' (24) Discount Amount -- Decimal
- ' (25) Discount until date -- Date
- ' (26) Transaction Due Date -- Date
- ' (27) Unpaid Amount due - Decimal

Return code	Description
96	Unknown Error occurred.

XML Example

The following example shows how to retrieve AR transaction records:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetARTransactionsPPD>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>100</add:p_lAccountNum>
      <add:p_iTankNum>0</add:p_iTankNum>
      <add:p_iServiceNum>0</add:p_iServiceNum>
      <add:p_iLocationNum>0</add:p_iLocationNum>
      <add:p_dteFromDate>2017-10-01</add:p_dteFromDate>
      <add:p_dteToDate>2017-10-31</add:p_dteToDate>
      <add:p_lMaxRows>50</add:p_lMaxRows>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetARTransactionsPPD>
  </soap:Body>
</soap:Envelope>
```

cTransaction.AcceptPayment()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

Accepts a payment using checking account information or a credit card number and expiration date (note – this does not integrate with the Energy/E3 Credit Card Verification module);

```
Public Function AcceptPayment(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iPaymentMethod As Short, _
    ByVal p_sPaymentMethodInfo As String, _
    ByVal p_iCCEExpirationMonth As Short, _
    ByVal p_iCCEExpirationYear As Short, _
    ByVal p_cPaymentAmount As Currency, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
' p_iCompanyID -- The Energy Company ID
' p_iPaymentMethod -- 2=Check, 3=Credit card
' p_sPaymentMethodInfo -- Checking account number or Credit card number
' p_iCCEExpirationMonth -- Credit card expiration month 1-12
' p_iCCEExpirationYear -- Credit card expiration Year (2-digit number -- 00-50)
```

For non- credit card payments. Please eneter a expiration month of 12 and exp year on 50

```
' p_cPaymentAmount -- Payment amount in dollars
' p_sSysUser - Calling program (IVRUser/WebUser)
'
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
' Return Type Integer -- Confirmation number
'
```

Return code	Description
96	Unknown error occurred.
102	Invalid database connection requested
113	Invalid expiration month
114	Invalid expiration year
115	Unable to get a new batch number
116	Unable to create a payment batch header
117	Invalid payment posting code
118	Unable to create the payment detail

.NET Example

The following example shows how to enter a payment transaction:

```
Dim objTransaction as Object
Dim lRetCode as Integer
Dim lConfirmationNo As Integer
'
Set objTransaction = CreateObject("EnergyAPI.cTransaction")
'
lConfirmationNo =
objTransaction.AcceptPayment(iCompanyID,lAccountNum,3,"5556667778889",12,03,250.00,
cSysUser,lRetCode)
'
If (lRetCode <> 0) Or (lConfirmationNo = 0) Then
    ' Failure - Display error message
Else
    ' Success
    Debug.print "Confirmation number=" & lConfirmationNo
End If
```

XML Example

The following example shows how to enter a payment transaction:

```
< <soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:AcceptPayment>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iPaymentMethod>3</add:p_iPaymentMethod>
      <add:p_sPaymentMethodInfo>Test</add:p_sPaymentMethodInfo>
      <add:p_iCCExpirationMonth>12</add:p_iCCExpirationMonth>
      <add:p_iCCExpirationYear>49</add:p_iCCExpirationYear>
      <add:p_cPaymentAmount>1.01</add:p_cPaymentAmount>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:AcceptPayment>
  </soap:Body>
</soap:Envelope>
```

cTransaction.AcceptPaymentDiscount()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method is provided in the Web service to accept a payment and allow for an additional amount to post to the account for prompt pay discount.

```
Public Function AcceptPaymentDiscount(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iPaymentMethod As Short, _
    ByVal p_sPaymentMethodInfo As String, _
    ByVal p_iCCEExpirationMonth As Short, _
    ByVal p_iCCEExpirationYear As Short, _
    ByVal p_cPaymentAmount As Currency, _
    ByVal p_cDiscountAmount As Currency, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'    p_iCompanyID -- The Energy Company ID
'    p_iPaymentMethod -- 2=Check, 3=Credit card
'    p_sPaymentMethodInfo -- Checking account number or Credit card number
'    p_iCCEExpirationMonth -- Credit card expiration month 1-12
'    p_iCCEExpirationYear -- Credit card expiration Year (2-digit number -- 00-50)
```

ⓘ For non- credit card payments. Please eneter a expiration month of 12 and exp year on 50

```
'    p_cPaymentAmount -- Payment amount in dollars
'    p_cDiscountAmount -- Prompt pay discount amount in dollars
'    p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'    Return Type Integer -- Confirmation number
'
```

Return code	Description
96	Unknown error occurred.
102	Invalid database connection requested
113	Invalid expiration month
114	Invalid expiration year
115	Unable to get a new batch number
116	Unable to create a payment batch header
117	Invalid payment posting code
118	Unable to create the payment detail

XML Example

The following example shows how to enter a payment transaction with discount:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:AcceptPaymentDiscount>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iPaymentMethod>3</add:p_iPaymentMethod>
      <add:p_sPaymentMethodInfo>Disc</add:p_sPaymentMethodInfo>
      <add:p_iCCExpirationMonth>12</add:p_iCCExpirationMonth>
      <add:p_iCCExpirationYear>49</add:p_iCCExpirationYear>
      <add:p_cPaymentAmount>1.02</add:p_cPaymentAmount>
      <add:p_cDiscountAmount>0.02</add:p_cDiscountAmount>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:AcceptPaymentDiscount>
  </soap:Body>
</soap:Envelope>
```

cTransaction.AcceptPaymentReference()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method is provided in the Web service to accept a payment and allow for an additional amount to post to the account for prompt pay discount as well as a reference number.

```
Public Function AcceptPaymentReference(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iPaymentMethod As Short, _
    ByVal p_sPaymentMethodInfo As String, _
    ByVal p_iCCEExpirationMonth As Short, _
    ByVal p_iCCEExpirationYear As Short, _
    ByVal p_cPaymentAmount As Currency, _
    ByVal p_cDiscountAmount As Currency, _
    ByVal p_iReferenceNumber As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As Integer
```

Input Parameters

```
'Accepts: p_lAccountNum -- The Account Number Without Check Digit
'          p_iCompanyID -- The Energy Company ID
'          p_iPaymentMethod -- 2=Check, 3=Credit card
'          p_sPaymentMethodInfo -- Checking account number or Credit card number
'          p_iCCEExpirationMonth -- Credit card expiration month 1-12
'          p_iCCEExpirationYear -- Credit card expiration Year (2-digit number -- 00-50)
```

i For non- credit card payments. Please eneter a expiration month of 12 and exp year of 50

```
'          p_cPaymentAmount -- Payment amount in dollars
'          p_cDiscountAmount -- Prompt pay discount amount in dollars
'          p_cDiscountAmount -- Payment reference number -- between 0 and 8388350
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'          Return Type Integer -- Confirmation number
```

Return code	Description
96	Unknown error occurred.
102	Invalid database connection requested
113	Invalid expiration month
114	Invalid expiration year
115	Unable to get a new batch number
116	Unable to create a payment batch header
117	Invalid payment posting code
118	Unable to create the payment detail

XML Example

The following example shows how to enter a payment transaction with reference:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:AcceptPaymentReference>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_iPaymentMethod>3</add:p_iPaymentMethod>
      <add:p_sPaymentMethodInfo>****1234</add:p_sPaymentMethodInfo>
      <add:p_iCCExpirationMonth>12</add:p_iCCExpirationMonth>
      <add:p_iCCExpirationYear>49</add:p_iCCExpirationYear>
      <add:p_cPaymentAmount>1.49</add:p_cPaymentAmount>
      <add:p_cDiscountAmount>0.25</add:p_cDiscountAmount>
      <add:p_iReferenceNumber>654321</add:p_iReferenceNumber>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:AcceptPaymentReference>
  </soap:Body>
</soap:Envelope>
```

cTransaction.GetNewUploadTransactionDataSet ()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Retrieves an empty dataset that can be populated with the data –
This dataset is used to call the UploadTransactions method.

Public Function **GetNewUploadTransactionDataSet**(ByVal p_iCompanyID As Short,
ByVal p_sSysUser As String,
ByRef p_lErrorCode As Integer) As DSARTransactions

Input Parameters

'Accepts: p_iCompanyID -- The Energy Company ID
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_lErrorCode – ByRef Argument – 0 = no error, <> 0 = Error Code
,

Return Values

'Returns: a new DSARTransactions typeddataset (EnergyAPI.DSARTransactions)

None. Returns an empty object of type DataSet

.NET Example

The following code demonstrates using the GetNewUploadTransactionDataSet **call**.to get a new DSArTransactions dataset.

```
Dim oTrans as EnergyAPI.cTransaction = new EnergyAPI.cTransaction
Dim oTransactions as EnergyAPI.DSArTransaction
oTransactions = oTrans.GetNewUploadTransactionDataSet(iCompanyID,sSysUser, lErrorCode)
```

XML Example

The following example shows how get a new transaction set:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetNewUploadTransactionDataSet>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetNewUploadTransactionDataSet>
  </soap:Body>
</soap:Envelope>
```

cTransaction.UploadTransactions()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Allows transactions to be loaded into Energy – will not calculate taxes or do any reprojection.

```
Public Function UploadTransactions(ByVal p_iCompanyID As Short,
    ByVal pDSTransactions As DSARTransactions,
    ByVal p_sSysUser As String,
    ByRef p_lErrorCode As Integer) As DSARTransactions
```

Input Parameters

'Accepts: p_iCompanyID -- The Energy Company ID
 ' pDSTransactions – data set (DSARTransactions) containing transaction data to upload
 ' p_sSysUser - Calling program (IVRUser/WebUser)
 ' p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Count (the error code is stored in the DSARTransaction row with the error...

Return Values

'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Count (if not zero, some rows in the DSARTransaction dataSet will have error codes.
 DSARTransactions – the input DS but with updated error codes, ...

Return code	Description
96	Unknown Error Occurred.
208	Invalid customer number.
212	Invalid event date.
213	Invalid accounting date.
214	Invalid posting date.
215	Invalid posting code.
216	Invalid reference number.
217	The event date and the accounting date must be within 125 days of each other.
218	The posting date and the event date must be within 127 days of each other.
219	Unable to update account record -- transaction skipped.
220	Unable to create transaction record -- transaction skipped.
221	Unable to create trans delivery record -- transaction skipped.
222	Unable to get new transaction key -- transaction skipped.
223	Invalid division.
224	Invalid type.
233	Invalid term ID
234	Invalid EFT transaction status

.NET Example

```

Dim oTransactions as EnergyAPI.cTransaction = new EnergyAPI.cTransaction
Dim oTransactionDS as EnergyAPI.DSARTransactions
...
sSysUser="WebUser"
iCompanyID =999
oTransactions = oTrans.GetNewUploadTransactionDataSet(iCompanyID,sSysUser, lErrorCode)

' Loop To Populate DSARTransactions with your data...
'EXAMPLE ONLY meant to illustrate how to add data to a strongly typed dataset DSARTransactions
Dim oneTransaction As EnergyAPI.DSARTransactions.ARTransactionsRow
Dim ii As Integer = 0

For ii = 1 To 10
    oneTransaction = oTransactionDS.ARTransactions.NewARTransactionsRow()
    oneTransaction.account_num = ii * 10000
    oneTransaction.accounting_date = DateAdd(DateInterval.Day, ii + 1, Now)
    oneTransaction.acquisition_payment = "N"
    ' ... rest of the fields....
    oneTransaction.error_code = 0
    ' add row to Dataset
    oTransactionDS.ARTransactions.Rows.Add(oneTransaction)
Next
' upload transactions
oTransactionDS=oTransactions.UploadTransactions(iCompanyID, oTransactionDS, sSysUser,
lErrorCode)
' any errors?
If errorCode <> 0 Then

    For Each oneTransaction In oTransactionDS.Tables(0).Rows
        If oneTransaction.error_code <> 0 Then
            Debug.Print(oUtility.GetErrorDescription(oneTransaction.error_code))
        End If
    Next

End If

```

XML Example

The following example shows how to upload an AR transaction record:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:UploadTransactions>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:pDSTransactions>
        <!--You may enter ANY elements at this point-->
        <ARTransactions>
          <account_num>2</account_num>
          <posting_date>2012-02-01</posting_date>

```

```
<event_date>2012-02-01</event_date>
<accounting_date>2012-02-01</accounting_date>
<posting_code>10</posting_code>
<batch_num>111</batch_num>
<net_amount>100</net_amount>
<gross_amount>100</gross_amount>
<gross_plus_embedded>100</gross_plus_embedded >
<reference_num>1</reference_num>
<tank_num>1</tank_num>
<service_num>0</service_num>
<location_num>0</location_num>
<division>1</division>
<type>2</type>
<partial_fill>N</partial_fill>
<runout>N</runout>
<keyoff_flag>N</keyoff_flag>
<trx_type>D</trx_type>
<comment>Calvin Test</comment>
<data_conversion>Y</data_conversion>
<update_balance>Y</update_balance>
<error_code>0</error_code>
<error_description></error_description>
</ARTransactions>
</add:pDSTransactions>
<add:p_sSysUser>WebUser</add:p_sSysUser>
<add:p_lErrorCode>0</add:p_lErrorCode>
</add:UploadTransactions>
</soap:Body>
</soap:Envelope>
```

gross_amount, net_amount and gross_plus_embedded must **always** be equal.

cTransaction.LoadPreDefinedComments()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

Loads Pre-defined comments used in E3's "Enter Payments" component from database.

Public Function LoadPreDefinedComments
 (ByVal pCompanyId As Short,
 ByVal pSysUser As String)
 As DCAddCommon.DSNumCodeDescription

Input Parameters

Accepts: pCompanyId -- The Energy Company ID
 pSysUser - Calling program (IVRUser/WebUser)

Return Values

Returns: data set containing pre-defined code & description

Return code	Description
N/A	N/A

.NET Example

```
Dim oTrans as EnergyAPI.cTransaction = new EnergyAPI.cTransaction
Dim oComments DCAddCommon.DSNumCodeDescription
```

```
oComments = oTrans. LoadPreDefinedComments (iCompanyId, sSysUser)
```

Display/use as needed...

XML Example

The following example shows how to load Pre-defined Comments:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadPreDefinedComments>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadPreDefinedComments>
  </soap:Body>
</soap:Envelope>
```

cTransaction.AcceptPayment_V2()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method is provided in the Web service to accept a payment and allow for an additional amount to post to the account for prompt pay discount as well as a reference number, this new method also allows the user to select a pre-defined comment or enter any other comment.

- The Event & Accounting dates were added to the “AcceptPaymentData” structure in version 14.01.01 of SmartConnect, the minimum database version needed to use the Event & Accounting dates is 12.01. To use SmartConnect v14.01.01 with a database version lower than 12.01 the properties should **not** be set to a value greater than or equal to 1960-12-31, which is the same as not setting the property values.

Public Function AcceptPayment_V2
 (ByVal pCompanyId As Short,
 ByVal pPaymentInfo As DCEnergyAPI.Transaction.AcceptPaymentData,
 ByVal pSysUser As String, ByRef pErrorCode As Integer) As Integer

Input Parameters

Accepts:

- pCompanyId -- The Energy Company ID
- pPaymentInfo – payment information data structure
- pSysUser - Calling program (IVRUser/WebUser)

Payment Information Data Structure “DCEnergyAPI.cTransaction.AcceptPaymentData”

AccountNum	E3 Account number
PaymentMethod	2=Check, 3=Credit card
PaymentMethodInfo	Checking account number or Credit card number
CCExpirationMonth	Credit card expiration month 1-12
CCExpirationYear	card expiration Year (2-digit number -- 00-50)
	i For non- credit card payments. Please enter a expiration month of 12 and exp year of 50
PaymentAmount	Payment amount in dollars
DiscountAmount	Prompt pay discount amount in dollars
ReferenceNumber	Payment reference number – between 0 and 8388350
PreDefinedComment	IF for the pre-defined comment used in E3’s “Enter Payments” (use value from LoadPreDefinedComments method)
UserComment	Any free form text limited to 100 characters
EventDate	Event date for the payment transaction. If not specified default will be 31 st December 1960, which defaults to the master file date when the payment transaction is created
AccountingDate	Accounting date for the payment transaction. If not specified default will be 31 st December 1960, which defaults to the master file date when the payment transaction is created
FinancialInstitution	Financial Institution ID. Use LoadFinancialInstitutions() API below to return the list of available financial institutions that accept check deposits and/or enabled as card issuer. Default value = 999 Online Payment

Return Values

Returns: pErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
 Return Type Integer -- Confirmation number

Return code	Description
96	Unknown error occurred.
102	Invalid database connection requested
113	Invalid expiration month
114	Invalid expiration year
115	Unable to get a new batch number
116	Unable to create a payment batch header
117	Invalid payment posting code
118	Unable to create the payment detail
605	The Accounting date must be greater than the Cutoff date.
606	The Accounting date must be less than or equal to the Posting date.
607	The Event date must be greater than the Accounting minus the maximum day limit established on this database
608	The Event date must be less than the Accounting date plus the maximum day limit established on this database

XML Example

The following example shows how to enter a payment transaction version 2:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:AcceptPayment_V2>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pPaymentInfo>
        <add:AccountNum>1</add:AccountNum>
        <add:PaymentMethod>3</add:PaymentMethod>
        <add:PaymentMethodInfo>test</add:PaymentMethodInfo>
        <add:CCEExpirationMonth>12</add:CCEExpirationMonth>
        <add:CCEExpirationYear>49</add:CCEExpirationYear>
        <add:PaymentAmount>1.01</add:PaymentAmount>
        <add:DiscountAmount>.01</add:DiscountAmount>
        <add:ReferenceNumber>654321</add:ReferenceNumber>
        <add:PreDefinedComment>2</add:PreDefinedComment>
        <add:UserComment>Test txn</add:UserComment>
        <add:EventDate>2014-11-13</add:EventDate>
        <add:AccountingDate>2014-11-13</add:AccountingDate>
        <add:FinancialInstitution>2</add:FinancialInstitution>
      </add:pPaymentInfo>
    </add:AcceptPayment_V2>
  </soap:Body>
</soap:Envelope>
```



```
</add:pPaymentInfo>
<add:pSysUser>WebUser</add:pSysUser>
<add:pErrorCode>0</add:pErrorCode>
</add:AcceptPayment_V2>
</soap:Body>
</soap:Envelope>
```

cTransaction.LoadFinancialInstitutions()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method is provided to return the list of active financial institutions that are used either as a card issuer or a check deposit location.

The Financial institution ID (Int-16) can be specified in the Payment information data structure when posting a payment transaction using AcceptPayment_V2() API.

Public Function LoadFinancialInstitutions

(ByVal pCompanyId As Short,

ByVal pSysUser As String, ByRef pErrorCode As Integer) As Integer

Input Parameters

Accepts:

pCompanyId -- The Energy Company ID

pPaymentInfo – payment information data structure

pSysUser - Calling program (IVRUser/WebUser)

Return Values

Returns: pErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number

Return Type:Array (of FinancialInstitution)

FinancialInstitution data structure:

InternalID	Financial Institution ID (1-32767)
Description	Financial Institution description
CashCheckDepositLocation	0/1 (true/false); the value is set to true if your company deposits cash and/or checks at this financial institution.
CreditCardIssuer	0/1 (true/false); the value is set to true if this financial institution issues credit cards used by your customers to make payment.

XML Example

The following example shows how to retrieve the list of available financial institutions :

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadFinancialInstitutions>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:AcceptPayment_V2>
  </soap:Body>
</soap:Envelope>
```

cTransaction.GetARTransactions_V2()

- License Required: SmartConnect API Customer Look-up Business Edition**

Important remark: GetARTransactions_V2() method will NOT include credit actions that are transactions with posting codes greater than the maximum credit posting code; you can use the original GetARTransactions() method to include the credit actions.

Description

Retrieves AR transaction records for a given company ID/account number, event date and other criteria.

```
Public Function GetARTransactions_V2(ByVal pCompanyID As Short, _
    ByVal pSysUser As String, _
    ByRef pErrorCode As Integer, _
    ByVal pSelCrit As DCEnergyAPI.Transaction.ArTransactionsSelCrit) _
    As DCEnergyAPI.Transaction.ARTransactionData()
```

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pSelCrit	Object of type DCEnergyAPI.Transaction.ArTr ansactionsSelCrit	See definition for details
Return Values			
	pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: DCEnergyAPI.Transaction.ARTransactionData			
	Custom data structure		See definition above

i We return payments (credits) as debits. The only way to tell the difference is from the posting code. You will have to get the maximum debit and the maximum credit codes from the client.

From 1 to to maxDebit is a debit
 From maxDebit + 1 to maxCredit is a credit
 From maxCredit + 1 to the end are to be ignored.

Class ArTransactionsSelCrit Definition:

Datatype/column name	Description
<code>int</code> AccountNum	E3 account number without check digit
<code>short[]</code> DeliveryLocations	A 16-bit integer array of Delivery locations (Tank numbers). Null or array with no elements = All Delivery locations (Tank numbers)
<code>short[]</code> ServiceLocations	A 16-bit integer array of Service locations. Null or array with no elements = All Service locations
<code>short[]</code> WDMSLocations	A 16-bit integer array of WDMS locations. Null or array with no

	elements = All WDMS locations
DateTime FromDate	From date, only transactions after this date will be retrieved
DateTime ToDate	To date, only transactions before this date will be retrieved
int MaxRecordsToReturn	Maximum number of records/transactions to return. <=0 = All transactions

XML document:

The following is the xml document representation of the class object as defined above.

```
<?xml version="1.0" encoding="utf-8"?>
<ArTransactionsSelCrit>
  <AccountNum>int</AccountNum>
  <DeliveryLocations>
    <!--Optional Element(s)-->
    <short>short</short>
  </DeliveryLocations>
  <ServiceLocations>
    <!--Optional Element(s)-->
    <short>short</short>
  </ServiceLocations>
  <WDMSLocations>
    <!--Optional Element(s)-->
    <short>short</short>
  </WDMSLocations>
  <FromDate>dateTime</FromDate>
  <ToDate>dateTime</ToDate>
  <MaxRecordsToReturn>int</MaxRecordsToReturn>
</ArTransactionsSelCrit>
```

Class ARTransactionData Definition:

Datatype/column name	Description
int AccountNum	E3 account number, without check digit
int TrxUniqueKey	Unique identifier for the transaction by account num
DateTime EventDate	The date the transaction occurred, e.g., (mm/dd/yyyy), the actual delivery date, payment date, etc
short PostingCode	Posting code on the transaction. This can be a debit or credit posting code. It can also be the credit action code or termination reason code
short BatchNum	The batch number in which the transaction was posted.
int ReferenceNum	The reference number on the transaction, such as a ticket or check number.
decimal NetAmount	The dollar amount of the transaction. Both debit and credit transactions will appear here. Whether they appear as positive numbers or negative numbers depends upon a system option that is set during installation. If the option is on, then regular credit transactions appear as negative and reversed credit transactions appear as positive. If the option is off, then regular credit transactions appear as positive and reversed credit transactions appear as negative. To inquire about changing this option, contact ADD Systems Customer Support.

decimal OpenAmt	The open dollar amount of the transaction that is currently open, i.e., not yet reconciled for an open item (keyoff =Y) customer. If a transaction has been fully reconciled, this field is blank. Open Item fields are only active if you are viewing transactions for an open item account. Note that whether open amounts appear as positives or negatives depends on a system option that is set during installation. If the option is on, then regular credit transactions appear as negative and reversed credit transactions appear as positive. If the option is off, then regular credit transactions appear as positive and reversed credit transactions appear as negative. To inquire about changing this option, contact ADD Systems Customer Support.
decimal Units	The number of units (not converted) delivered.
short TankNum	Identifies the tank number on the transaction.
short ServiceNum	Identifies the service location number on the transaction.
short LocationNum	Identifies the WDMS location number on the transaction.
decimal GrossAmount	The total sales dollars for the product excluding taxes.
short Division	The division on the transaction.
short Type	The customer type on the transaction.
short Category	The customer category on the transaction.
int CostCenter	The cost center number used on the transaction. This may be the same as the division number or on databases using General Ledger, the GL number.
int DegreeDay	If the transaction was a delivery, this value represents the degree day of the delivery.
decimal Ppg	[Price/Unit] → Price per gallon
decimal Deviation	The effective deviation on the transaction. This value represents the difference between the tank's base price code price (without the tank deviation) and the price the customer actually paid. This difference can be the result of a tank price deviation, account discount percent, posting price override, or a price protection plan. If the actual price is lower than the BPC price, then the value in this field should be negative. If the actual price is higher than the BPC price, it should be positive.
string Runout	(Y/N) For delivery transactions, this flag indicates whether the delivery was posted as a runout.
string PartialFill	(Y/N) For delivery transactions, this flag indicates whether the delivery was posted as a partial fill.
short DeliveryReason	Non-delivery code. If a delivery attempt was made but was unsuccessful, a code from the Delivery Reason Codes table may appear in this column.
decimal InTankUnits	Represents the in-tank units present after a partial fill, if the driver measured the tank and if the product is not in the Product Supplemental Table. Or, if it is, it has been set up as "units".
byte PercentFill	The in-tank percent present after a partial fill, if the driver measured the tank and if the product is set up as "percent" in the Product Supplemental Table.
string Comment	Free-form text comment entered when a payment transaction is posted via the Enter Payments window.
string LastMaintenanceUserId	The user ID assigned to the person who performed the last modification.
DateTime LastMaintenanceDt	The date and time of the last modification. This can include not only

	user modifications, but also an action taken by the system, such as reconciling open items or including the transaction on a statement.
DateTime TrxInserted	The date and time the transaction was inserted into the database.
string CreatedBy	The user ID assigned to the person who created the transaction.
short SalesType	The sales type on the transaction, if one was entered.
short CostType	The cost type on the transaction, if one was entered.
short DepartmentCode	The department code on the transaction, if one was entered.
short TermId	The user-defined ID number from the Terms Code Table. This code indicates the customer's payment terms.
string EftTrxStatus	The status of the transaction, if it was an EFT transaction. One of the following status codes may appear in this field: N = Not EFT U = Undrafted D = Drafted P = Pending Draft E = Payment resulting from EFT
string EftDueDate	For EFT customers, the system-calculated date that the payment is due to be drafted. It is calculated based on the Terms Code on the EFT customer record.
decimal DiscountAmt	The calculated terms discount applied to the transaction. This could be based on a percentage of sale or cents per unit. The discount amount is calculated when payment is posted.
short ItsProdNum	Inventory Tracking System product number assigned to the tank record
decimal UnitsUsed	Units sold to customer, e.g., ITS, cylinder or meter reading sales
short UnitsReturned	Units returned (<i>ITS only</i>)
int CardSiteNum	The CARD site number
int CardNum	The CARD customer's card number
string PublicationPrice	Flag indicating that the customer has publication pricing. Publication pricing is set in Maintain Customers (Y/N)
int DocLinkDocId	The latest document id for the customer
int TransLinkKey	trx_unique_key * adjustment_code
int TransAdkjustKey	Count of transactions from TRANS_ADJUSTED
string SalesTypeDesc	Sales type description
string CostTypeDesc	Cost type description
string DepartmentCodeDesc	Department code description
string PostingCodeDesc	Posting code description
string Source	[TLS] → Source of transaction (Tank, Service, Location)
string AdjustmentStatus	Adjusted, Pending
decimal RunningBalance	The customer's A/R balance as of that transaction. The running balance appears in the transaction grid when it is first opened. Note that if the system option: "TRX Disp-Blank run bal when grid filtered/sorted" is enabled, the running balance will not be displayed, if transaction data is filtered using a specific posting code or transaction type, or if it is sorted by any column. When unchecked, the running balance will be displayed, except if the user summarizes transactions by reference number or invoice number, in which case, it will be blank.  Sorting transactions in any order other than the default sort

	order makes it difficult to understand the running balance. The nature of a running balance requires that transactions be in descending posting date order, to preserve the integrity of the balance. Therefore, if you change the sort order, for example, sorting by Reference #, Posting Code, or Department Code, be aware that the running balance for each transaction will be correct as of that transaction's posting date, although the order of transactions may cause it to appear inaccurate.
<code>string</code> DivisionDesc	Division description
<code>string</code> TypeDesc	Customer type description
<code>string</code> CategoryDesc	Customer category description
<code>short</code> EftBatchNumber	If the EFT status is 'P', then the EFT batch number will be displayed in this field.
<code>string</code> ItsProductDesc	ITS product description
<code>decimal</code> ItsNetChange	units_used minus units_returned
<code>decimal</code> UsedPpu	Used PPU
<code>DateTime</code> PostingDate	The date the transaction was posted.
<code>DateTime</code> AccountingDate	The accounting date of the transaction.
<code>DateTime</code> InvoiceDate	Invoice date and time (for CARD transactions only). Assigned by system when FEECARD is run.
<code>string</code> Salesperson	The salesperson on the account
<code>int</code> InvoiceNum	This field indicates the Invoice number, if one was assigned to the transaction. When enabled, the System Option: "Outsource-Use doc viewer ID as inv# in flat files", will cause the posting programs to use the unique Document Viewer ID number as the Invoice Number. This will guarantee, for all practical purposes, that all invoice numbers are unique. This also provides a direct link between the transaction and the billing document on which it appears. Other options can populate this field without this option, namely a STRTAB option to consolidate Card invoices and a STRTAB option to create an invoice number for IVO accounts.
<code>string</code> XOptionFlag	Suppress invoice (Y/N)
<code>string</code> PoNumber	This field displays the purchase order number on the transaction. For delivery transactions, the PO Number will be included with the delivery details if one has been assigned to the delivery location or if it was entered in the Delivery Order screen. For Card System transactions, the PO number will be included with the transaction details if one was entered either electronically (via APOST) or manually (via Card System Transaction Input/Review).
<code>short</code> TaxJurisdiction	Tax jurisdiction for the account
<code>short</code> CommentNum	Internal number assigned to predefined comment
<code>string</code> InvoiceComment	character field for invoice comments
<code>byte</code> DeliveryType	The delivery type is calculated by evaluating the tank setup at the time of delivery. Valid values are 0, 1, 2, 3, 5, 7, 9, 11, 17, 19, 25, 32, 34 See "DeliveryTypeDesc" for descriptions
<code>decimal</code> KFactor	K-factor on the delivery transaction
<code>short</code> DriverNum	The driver number assigned to the delivery. Only the driver code will appear here. Driver codes and driver names are available in the DRIVER_HEADER table

short TruckNum	The number of the truck used for the transaction
short RunoutReason	One of a list of predefined reason codes for runout
short PartFillReason	The reason for a partial fill
short PaymentMethod	Payment method code, set up in the Payment Methods component (e.g., check, credit card, cash, etc.)
short PaymentType	Identifies whether payment is a regular payment, security deposit, pre-buy, EFT, online or energy assistance. See Payment Types table
short CcType	Type of credit card (e.g., VISA, MasterCard)
short PaymentMethodInfo	Indicates the Financial institution for the Payment Method set up in Financial Institutions component
byte MtrType	The meter type on the transaction, as defined in the Meter Type Conversion Table
decimal CurrMtrRdg	The current meter reading on the transaction
decimal PrevMtrRdg	The meter reading prior to current
string MtrRdgEst	The estimated meter reading for this transaction (Y/N)
string CardVehNum	The customer's vehicle number for CARD transactions
decimal CardOdometer	The odometer reading which may have been entered at the pump, for CARD transactions
int CardTrxNum	The CARD transaction number
decimal Ppu	The price per unit/gallon of product that was delivered.
string DeliveryTypeDesc	The delivery type is calculated by evaluating the tank setup at the time of delivery. Delivery types can be as follows: 0 = Auto-K-Factor 1 = Auto-Days Between Delivery 2 = Will Call-K-Factor 3 = Will Call-Days Between Delivery 4 = Auto-Route Table 5 = Auto-Route Table 7 = Will Call-Route Table 9 = Auto-Usage Rate 11 = Will Call-Usage Rate 17 = Auto-Day of Week 19 = Will Call-Day of Week 25 = Auto-Day of Week 32 = Auto-K-Factor/Variable Hot Water 34 = Will Call-K-Factor/Variable Hot Water
string SalespersonDesc	Name of the salesperson on the account
string TaxJurisdictionDesc	Description of the tax county of the transaction
string CommentDescription	This column shows a transaction's pre-defined comments. The Pre-Defined Transaction Comments component is found in System Maintenance – Bookkeeping – Accounts Receivable . They can be assigned in the following components – Create Installment Plans, Enter Delivery Billing, Enter Payments, and Find and Adjust Transactions.
byte CardHour	The hour at which product was pumped for CARD transactions
byte CardMinute	The minute at which product was pumped for CARD transactions
string CardTime	The time at which the product was pumped (combination of CardHour & CardMinute) only applicable to card transactions
string GasboyTrxFlag	Flag indicating whether this was a Gasboy transaction

int CcTrxHdrId	Unique number generated for each credit card transaction header record. This column links to the CC_TRANS_HDR_HIST table
string PubPriceStatus	[Price Status] → The notation "PPP" in this column means that this transaction is pending publication pricing, i.e., a price has not been established yet. The notation "PP" in this column indicates that the transaction has been updated with publication pricing.
decimal MeteredUnits	Change in meter reading between the prior & current meter reading
decimal CalcNetAmount	[Net Amount] → Calculated net amount.
decimal CalcOpenAmt	[Open Amount] → Calculated open amount
decimal CalcGrossAmount	[Gross Amount] → Calculated gross amount
decimal EffectOnBalance	Effect on the AR balance
string PostingCodeDescLong	The posting code description, e.g., #2 fuel oil, parts, labor, etc.
string CcStatus	For credit card related transactions, this column displays the credit card status for the transaction. Examples of a status are charge not submitted, charge submitted, and voided.
string CcValidationId	This column shows the credit card validation ID for credit card transactions.
bool AllowAdjust	Allow adjustment
bool AllowCcCharge	Allow CC charge
string TrxType	D = debit transaction; C = credit transaction
int LdtrkEventId	If the order was processed in Load Track, this field indicates the Load Track Event ID that was assigned to the event. This ID can be used to search for more details in Display & Adjust Wholesale Events and in Bill of Lading Display.
string MinimumCharge	Flag (Y/N) indicating whether the sale was billed at the minimum charge amount or not
int CcBatchNum	This value represents the number of the Credit Card Batch in which this transaction was processed.
string TermCodeDesc	Term code description
string KeyoffFlag	When "Y", indicates that this account is an open item account (Y/N)
decimal AmountForCcCharge	Amount allowed for CC charge
int JeNum	This column is available if the system option "GP-Enable multiple JE export in one acct. period" is turned on. When JRNL is run for the purpose of creating journal entries, all transactions within that JRNL report are assigned a journal entry number. For example, if there are 300 transactions reported, and the next journal entry number available is 388, then all transactions will have the same journal entry number 388. This column shows the system generated journal entry number for this transaction.
int JeLineNum	When JRNL is run for the purpose of creating journal entries, all transactions within that JRNL report are assigned a journal entry line number. For example, if there are 300 transactions reported, each transaction will have a unique journal entry line number assigned between 1 and 300. This column shows the system generated journal entry line number for the transaction.
string DebitGLNum	This column displays the debit GL number for the transaction, usually representing the A/R portion of the journal entry.
string CreditGLNum	This column displays the credit GL number for the transaction, usually representing the product sale portion of the journal entry.
string UnitsGLNum	The units GL number is shown in this column and represents the GL

	number for units for the product.
<code>string</code> CardSite	Description for CardSiteNum
<code>DateTime</code> DueByDate	Pay by this date to receive a discount
<code>string</code> DriverName	Driver name for the "DriverNum"
<code>string</code> TruckName	Truck name for the "TruckNum"

XML document:

The following is the xml document representation of the class object as defined above.

```
<?xml version="1.0" encoding="utf-8"?>
<ARTransactionData>
  <AccountNum>int</AccountNum>
  <TrxUniqueKey>int</TrxUniqueKey>
  <EventDate>dateTime</EventDate>
  <PostingCode>short</PostingCode>
  <BatchNum>short</BatchNum>
  <ReferenceNum>int</ReferenceNum>
  <NetAmount>decimal</NetAmount>
  <OpenAmt>decimal</OpenAmt>
  <Units>decimal</Units>
  <TankNum>short</TankNum>
  <ServiceNum>short</ServiceNum>
  <LocationNum>short</LocationNum>
  <GrossAmount>decimal</GrossAmount>
  <Division>short</Division>
  <Type>short</Type>
  <Category>short</Category>
  <CostCenter>int</CostCenter>
  <DegreeDay>int</DegreeDay>
  <Ppg>decimal</Ppg>
  <Deviation>decimal</Deviation>
  <Runout>string</Runout>
  <PartialFill>string</PartialFill>
  <DeliveryReason>short</DeliveryReason>
  <InTankUnits>decimal</InTankUnits>
  <PercentFill>unsignedByte</PercentFill>
  <Comment>string</Comment>
  <LastMaintenanceUserId>string</LastMaintenanceUserId>
  <LastMaintenanceDt>dateTime</LastMaintenanceDt>
  <TrxInserted>dateTime</TrxInserted>
  <CreatedBy>string</CreatedBy>
  <SalesType>short</SalesType>
  <CostType>short</CostType>
  <DepartmentCode>short</DepartmentCode>
  <TermId>short</TermId>
  <EftTrxStatus>string</EftTrxStatus>
  <EftDueDate>string</EftDueDate>
  <DiscountAmt>decimal</DiscountAmt>
  <ItsProdNum>short</ItsProdNum>
  <UnitsUsed>decimal</UnitsUsed>
  <UnitsReturned>short</UnitsReturned>
  <CardSiteNum>int</CardSiteNum>
  <CardNum>int</CardNum>
  <PublicationPrice>string</PublicationPrice>
```

```
<DocLinkDocId>int</DocLinkDocId>
<TransLinkKey>int</TransLinkKey>
<TransAdkjustKey>int</TransAdkjustKey>
<SalesTypeDesc>string</SalesTypeDesc>
<CostTypeDesc>string</CostTypeDesc>
<DepartmentCodeDesc>string</DepartmentCodeDesc>
<PostingCodeDesc>string</PostingCodeDesc>
<Source>string</Source>
<AdjustmentStatus>string</AdjustmentStatus>
<RunningBalance>decimal</RunningBalance>
<DivisionDesc>string</DivisionDesc>
<TypeDesc>string</TypeDesc>
<CategoryDesc>string</CategoryDesc>
<EftBatchNumber>short</EftBatchNumber>
<ItsProductDesc>string</ItsProductDesc>
<ItsNetChange>decimal</ItsNetChange>
<UsedPpu>decimal</UsedPpu>
<PostingDate>dateTime</PostingDate>
<AccountingDate>dateTime</AccountingDate>
<InvoiceDate>dateTime</InvoiceDate>
<Salesperson>string</Salesperson>
<InvoiceNum>int</InvoiceNum>
<XOptionFlag>string</XOptionFlag>
<PoNumber>string</PoNumber>
<TaxJurisdiction>short</TaxJurisdiction>
<CommentNum>short</CommentNum>
<InvoiceComment>string</InvoiceComment>
<DeliveryType>unsignedByte</DeliveryType>
<KFactor>decimal</KFactor>
<DriverNum>short</DriverNum>
<TruckNum>short</TruckNum>
<RunoutReason>short</RunoutReason>
<PartFillReason>short</PartFillReason>
<PaymentMethod>short</PaymentMethod>
<PaymentType>short</PaymentType>
<CcType>short</CcType>
<PaymentMethodInfo>short</PaymentMethodInfo>
<MtrType>unsignedByte</MtrType>
<CurrMtrRdg>decimal</CurrMtrRdg>
<PrevMtrRdg>decimal</PrevMtrRdg>
<MtrRdgEst>string</MtrRdgEst>
<CardVehNum>string</CardVehNum>
<CardOdometer>decimal</CardOdometer>
<CardTrxNum>int</CardTrxNum>
<Ppu>decimal</Ppu>
<DeliveryTypeDesc>string</DeliveryTypeDesc>
<SalespersonDesc>string</SalespersonDesc>
<TaxJurisdictionDesc>string</TaxJurisdictionDesc>
<CommentDescription>string</CommentDescription>
<CardHour>unsignedByte</CardHour>
<CardMinute>unsignedByte</CardMinute>
<CardTime>string</CardTime>
<GasboyTrxFlag>string</GasboyTrxFlag>
<CcTrxHdrId>int</CcTrxHdrId>
<PubPriceStatus>string</PubPriceStatus>
<MeteredUnits>decimal</MeteredUnits>
```

```

    <CalcNetAmount>decimal</CalcNetAmount>
    <CalcOpenAmt>decimal</CalcOpenAmt>
    <CalcGrossAmount>decimal</CalcGrossAmount>
    <EffectOnBalance>decimal</EffectOnBalance>
    <PostingCodeDescLong>string</PostingCodeDescLong>
    <CcStatus>string</CcStatus>
    <CcValidationId>string</CcValidationId>
    <AllowAdjust>boolean</AllowAdjust>
    <AllowCcCharge>boolean</AllowCcCharge>
    <TrxType>string</TrxType>
    <RptOpenDb>decimal</RptOpenDb>
    <RptOpenCr>decimal</RptOpenCr>
    <RptNetDb>decimal</RptNetDb>
    <RptNetCr>decimal</RptNetCr>
    <LdtrkEventId>int</LdtrkEventId>
    <MinimumCharge>string</MinimumCharge>
    <CcBatchNum>int</CcBatchNum>
    <TermCodeDesc>string</TermCodeDesc>
    <KeyoffFlag>string</KeyoffFlag>
    <AmountForCcCharge>decimal</AmountForCcCharge>
    <JeNum>int</JeNum>
    <JeLineNum>int</JeLineNum>
    <DebitGlNum>string</DebitGlNum>
    <CreditGlNum>string</CreditGlNum>
    <UnitsGlNum>string</UnitsGlNum>
    <CardSite>string</CardSite>
    <DueByDate>dateTime</DueByDate>
    <DriverName>string</DriverName>
    <TruckName>string</TruckName>
</ARTransactionData>

```

.NET Example

The following example shows how to retrieve AR transaction records:

```

short e3CompanyID = 1;    // The E3 AR Database #
string scUser = "WebUser";    // Identifier Calling Application
int errCode = 0; //Return Code For Errors
DCEnergyAPI.Transaction.ArTransactionsSelCrit selCrit = new DCEnergyAPI.Transaction.ArTrans
actionsSelCrit();
DCEnergyAPI.Transaction.ARTransactionData[] arTrxResults = null;

// Populate Selection Criteria
selCrit.AccountNum = 100;
selCrit.DeliveryLocations = null; //Optional Field
//selCrit.ServiceLocations    //Optional Field
selCrit.WDMSLocations = new short[] { 1 };
selCrit.FromDate = new DateTime(2017, 01, 01);    //Default = 12/31/1960
selCrit.ToDate = new DateTime(2017, 01, 31);    //Default = 12/31/2049
selCrit.MaxRecordsToReturn = 100; //Default value (0) = All

using (EnergyAPI.cTransaction SmartConnectTransactionsObject = new EnergyAPI.cTransaction()
)

```

```

{
    arTrxResults = SmartConnectTransactionsObject.GetARTransactions_V2
                    (e3CompanyID, scUser, ref errCode, selCrit);
}

if (errCode != 0)
{
    //Non-
    zero Error Code, we need to do something. Maybe get error description from SmartConnect log
    it & display to user....
}

if (arTrxResults != null)
{
    foreach (DCEnergyAPI.Transaction.ARTransactionData trx in arTrxResults)
    {
        //Do something with the return data....
    }
}

```

XML Example

The following example shows how to retrieve AR transaction records:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-
envelope" xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetARTransactions_V2>
      <add:pCompanyID>1</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
      <add:pSelCrit>
        <add:AccountNum>100</add:AccountNum>
        <add:DeliveryLocations>
        </add:DeliveryLocations>
        <add:ServiceLocations>
        </add:ServiceLocations>
        <add:WDMSLocations>
          <add:short>1</add:short>
        </add:WDMSLocations>
        <add:FromDate>2017-01-01</add:FromDate>
        <add:ToDate>2017-01-31</add:ToDate>
        <add:MaxRecordsToReturn>100</add:MaxRecordsToReturn>
      </add:pSelCrit>
    </add:GetARTransactions_V2>
  </soap:Body>
</soap:Envelope>

```

cTransaction.CreateAdjustmentTransferEntry()

Description

To be used to create data entries in the E3 system that will be processed via various E3 functions/interfaces.

NOTE currently only supports E3 General Interface File Creation: “Refunds – Energy to AP”.

```
Public Function CreateAdjustmentTransferEntry(ByVal pCompanyID As Short, ByVal pSysUser As String, ByVal pAdjustmentTransferData As DCEnergyAPI.Transaction.AdjustmentTransferData) As Integer
```

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAdjustmentTransferData	Object of type DCEnergyAPI.Transaction.AdjustmentTransferData	See definition below for details.
Return Type: Long Integer			
			0 - if Successful<> 0 Error Number

DCEnergyAPI.Transaction.AdjustmentTransferData

Datatype/column name	Description
int AccountNum	E3 account number, without check digit
Enum.AdjustmentTransferTypes AdjustmentTransferType	Refund =6
int ReferenceNumber	Transaction Reference Number
int TrxUniqueKey	trx_unique_key as returned by UploadTransactions() or GetARTransactions_V2() ***
decimal Amount	Refund Amount

*** TrxUniqueKey: if no value is supplied (ie: 0) then no integrity validation can/will occur.

```
public enum AdjustmentTransferTypes
{
    [XmlEnum("6")]
    Refund = 6
}
```

When using XML, use the numeric representation and not the name.

Refunds - Energy to AP

Important remark: The actual refund transaction needs to be created via the "UploadTransactions" method. All other account level data required for refunds will be retrieved by this method. E3's "General Interface File Creation" must still be used to generate the flat file.

XML Example

The following example shows how to create a refund entry:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-
envelope" xmlns:add="http://www.addsys.com/">
  <soap:Header />
  <soap:Body>
    <add:CreateAdjustmentTransferEntry>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <!--Optional:-->
      <add:pAdjustmentTransferData>
        <add:AccountNum>1</add:AccountNum>
        <add:AdjustmentTransferType>6</add:AdjustmentTransferType>
        <add:Amount>-10</add:Amount>
        <add:TrxUniqueKey>-713</add:TrxUniqueKey>
        <add:ReferenceNumber>54321</add:ReferenceNumber>
      </add:pAdjustmentTransferData>
    </add:CreateAdjustmentTransferEntry>
  </soap:Body>
</soap:Envelope>
```

In the above example, since the `trx_unique_key` is provided the system will verify that:

- 1) the transaction for the account exists.
- 2) that the posting code matches the refund posting code as defined in the system.
- 3) that the amount sent matches what is posted to the account.
- 4) that the reference # sent matches what is on the transaction.

cTransaction.GetAssetDeliveryDetails()

Description

To be used to retrieve asset delivery details from the E3 system.

```
GetAssetDeliveryDetails(ByVal p_iCompanyID As Short, ByVal p_sSysUser As String, p_iAccountNum As Integer, p_iTrxUniqueKey As Integer) As DCEnergyAPI.Transaction.AssetDeliveryDetails
```

Input Parameters

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	p_iCompanyID	Short (16 bit integer)	The Energy Company ID
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_iAccountNum	Integer	Account number
	p_iTrxUniqueKey	Integer	Unique Transaction Key
Return Type: DCEnergyAPI.Transaction.AssetDeliveryDetails			

DCEnergyAPI.Transaction.AssetDeliveryDetails

Datatype/column name	Description
int AccountNum	E3 account number, without check digit
int TransactionUniqueKey	trx_unique_key as returned by UploadTransactions() or GetARTransactions_V2() ***
AssetDeliveryDetail[] AssetDeliveryDetailArray	Array of AssetDeliveryDetail class
ApiException Exception	Validation exceptions

DCEnergyAPI.Transaction.AssetDeliveryDetail

Datatype/column name	Description
decimal AssetLatitude	Latitude of asset
decimal AssetLongitude	Longitude of asset
string BarCode	The bar code
string Description	Description of transaction
decimal LastEngineHourReading	Last engine hour reading value (future use)
DateTime LastMaintenanceDate	Last maintenance date
string LastMaintenanceUserId	Last maintenance userId
short NonFillReason	Non-fill reason (future use)
int Odometer	Odometer value
DateTime ScanBeginDate	Scan begin date
DateTime ScanEndDate	Scan end date
short SequenceNumber	Sequence number

decimal Units	Units

DCEnergyAPI.ApiException

Datatype/column name	Description
int ErrorNumber	Error number
string ErrorMessage	Error description

XML Example

The following example shows how to retrieve asset delivery details:

Input:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:GetAssetDeliveryDetails>
      <add:p_iCompanyID>1800</add:p_iCompanyID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_iAccountNum>4523</add:p_iAccountNum>
      <add:p_iTrxUniqueKey>-3040</add:p_iTrxUniqueKey>
    </add:GetAssetDeliveryDetails>
  </soapenv:Body>
</soapenv:Envelope>
```

Output:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetAssetDeliveryDetailsResponse xmlns="http://www.addsys.com/">
      <AssetDeliveryDetailsRoot xmlns="http://www.addsys.com/transaction">
        <AccountNumber>4523</AccountNumber>
        <TransactionUniqueKey>-3040</TransactionUniqueKey>
        <AssetDeliveryDetails>
          <AssetDeliveryDetail>
            <AssetLatitude>0.000000</AssetLatitude>
            <AssetLongitude>0.000000</AssetLongitude>
            <BarCode>bar code 1-2-3-4-5-5</BarCode>
            <Description>bar code 1-2-3-4-5-5 -
123456789012345678901234567</Description>
            <LastEngineHourReading>0</LastEngineHourReading>
            <LastMaintenanceDate>2017-06-02T10:02:10.233</LastMaintenanceDate>
            <LastMaintenanceUserId>DESIGN17U</LastMaintenanceUserId>
            <NonFillReason>0</NonFillReason>
            <Odometer>0</Odometer>
            <ScanBeginDate>1960-12-31T00:00:00</ScanBeginDate>
            <ScanEndDate>1960-12-31T05:59:00</ScanEndDate>
            <SequenceNumber>1</SequenceNumber>
            <Units>150</Units>
          </AssetDeliveryDetail>
        </AssetDeliveryDetails>
      </AssetDeliveryDetailsRoot>
    </GetAssetDeliveryDetailsResponse>
  </soap:Body>
</soap:Envelope>
```

```
<AssetLongitude>0.000000</AssetLongitude>
<BarCode>003333457633-f</BarCode>
<Description></Description>
<LastEngineHourReading>0</LastEngineHourReading>
<LastMaintenanceDate>2017-06-02T10:02:10.296</LastMaintenanceDate>
<LastMaintenanceUserId>DESIGN17U</LastMaintenanceUserId>
<NonFillReason>0</NonFillReason>
<Odometer>0</Odometer>
<ScanBeginDate>1960-12-31T00:00:00</ScanBeginDate>
<ScanEndDate>1960-12-31T20:01:00</ScanEndDate>
<SequenceNumber>2</SequenceNumber>
<Units>100</Units>
</AssetDeliveryDetail>
</AssetDeliveryDetails>
</AssetDeliveryDetailsRoot>
</GetAssetDeliveryDetailsResponse>
</soap:Body>
</soap:Envelope>
```

cLookup Class

Overview

This section describes the methods that enable the programmer to retrieve company and division names. The following methods are covered in this section:

- **GetCompanyName():** Retrieves one or all company names
- **GetDivisionName():** Gets one or all division names that exist in the database for a company attribute
- **GetLookupStoredValues():** returns a list of descriptions to be presented to the end user (for example in a dropdown list).
- **ValidateZipCode():** validates the passed zip code and returns true/false based on outcome.

cLookUp.GetCompanyName()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves one or all company names.

```
Public Function GetCompanyName(ByVal p_iCompanyID As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_iCompanyID -- 0 = All Companies, <> 0 Specific company ID
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Company number ("company_no") - Short integer
'          (1)Company name ("company_name") -- char(50)
'
```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve company name(s):

```
Dim objLookUp as Object
Dim lRetCode as Integer
Dim objCompany As System.Data.DataSet
'
Set objLookUp = CreateObject("EnergyAPI.cLookUp")
objCompany = objLookUp.GetCompanyName(iCompanyID,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objCompany.Tables(0).Rows
        Debug.Print "Company ID:" & Row (0) & "Name:" & Row (1)
    Next
End If
```

XML Example

The following example shows how to retrieve company name(s):

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCompanyName>
      <add:p_iCompanyID>0</add:p_iCompanyID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetCompanyName>
  </soap:Body>
</soap:Envelope>
```

cLookUp.GetDivisionName()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Gets one or all division names that exist in the database for a company attribute

```
Public Function GetDivisionName(ByVal p_iCompanyID As Short, _
    ByVal p_iDivisionNum As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

Input Parameters

```
'Accepts: p_iCompanyID -- Company ID
'          p_iDivisionNum -- 0 = All divisions, <> 0 Specific division #
'          p_sSysUser - Calling program (IVRUser/WebUser)
```

Return Values

```
'Returns: p_lErrorCode (ByRef) -- 0 If Successful : <> 0 Error Number
'
'          Return Type DataSet.Tables(0).Rows (0)Division number ("division") - Short integer
'          (1)Division name ("division_name") -- char(40)
'
```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve division name(s) for a given company ID:

```
Dim objLookUp as Object
Dim lRetCode as Integer
Dim objDivision As System.Data.DataSet
'
Set objLookUp = CreateObject("EnergyAPI.cLookUp")
'
objDivision = objLookUp.GetDivisionName(iCompanyID,0,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objDivision.Tables(0).Rows
        Debug.Print "Division number :" & Row (0) & " Division name:" & Row (1)
    Next
End If
```

XML Example

The following example shows how to retrieve division name(s) for a given company ID:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetDivisionName>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_iDivisionNum>1</add:p_iDivisionNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetDivisionName>
  </soap:Body>
</soap:Envelope>
```

cLookup.GetLookupStoredValues()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Returns a list of descriptions for end user selection.

```
Public Function GetLookupStoredValues(ByVal p_iCompanyID As Short, _
    ByVal p_sLookupValues As String, _
    ByVal p_sStoredValues as String, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_sLookupValues	String	String value as retrieved from GetCustMasterFields/GetTankFields/GetMeterFields Etc.. (ie: DataSet.Tables(0).Rows(0).Item(5))
	p_sStoredValues	String	String value as retrieved from GetCustMasterFields/GetTankFields/GetMeterFields Etc.. (ie: DataSet.Tables(0).Rows(0).Item(6))
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type : System.Data.DataSet

- (0) Stored value ("stored_value") – Char(255)
The value that will be saved to the database.
- (1) Lookup value ("lookup_value") – Char(255)
The description as seen by the end user.
- (2) Status ("status") – char(1)
'A' – Active item, 'I' – Inactive item. When in add mode, the record set
must be filtered to only display "active" items.

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve a list of lookup/stored values based on the values stored in the account record fields:

```
Dim objLookUp as Object
Dim lRetCode as Integer
Dim objFieldList As System.Data.DataSet
```



```
Set objFieldList = CreateObject("EnergyAPI.cLookup")
'
objFieldList =
objLookup.GetLookupStoredValues(iCompanyID,sLookup,sStored,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFieldList.Tables(0).Rows
        Debug.Print "Stored Value:" & Row (0) & " Lookup value:" & Row (1) & "
        Status: " & Row (2)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of lookup/stored values based on the values stored in the account record fields:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetLookupStoredValues>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_sLookupValues>1</add:p_sLookupValues>
      <add:p_sStoredValues>1</add:p_sStoredValues>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetLookupStoredValues>
  </soap:Body>
</soap:Envelope>
```

cLookup.ValidateZipCode()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Will validate the inputted zip code and returns true/false.

```
Public Function ValidateZipCode(ByVal p_sZipCode as string, _
    ByVal p_sSysUser As String, _
    ByRef p_iCompanyID As Short, _
    ByRef p_iDivision As Short) As Integer
    ,
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_sZipCode	String	Value as inputted by end user.
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_iCompanyID (ByRef)	Short	Valid company ID
p_iDivision (ByRef)	Short	Valid division ID

Return Type : Long integer 0 = Success, <> 0 Error number

if zip code exists in the ENERGY_DIVISION_ZIP_CODE table, the API call will return the company ID and Division number

Return code	Description
96	Unknown Error Occurred.
126	Invalid zip code entered. Please try again

.NET Example

The following example shows how to validate an inputted zip code:

```
Dim objLookUp as Object
Dim lRetCode as Integer
Dim iCompanyID as Short
Dim iDivisionID as Short
Set objLookUp = CreateObject("EnergyAPI.cLookUp")
,
lRetCode = objLookUp.ValidateZipCode(sZipEntered,cSysUser,iCompanyID, iDivisionID)
,
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    ' Use iCompanyID && iDivisionID as returned by this API call to create a new account
End If
```

XML Example

The following example shows how to validate an inputted zip code:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ValidateZipCode>
      <add:p_sZipCode>10010</add:p_sZipCode>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_iDivision>1</add:p_iDivision>
    </add:ValidateZipCode>
  </soap:Body>
</soap:Envelope>
```

cUtility Class

Overview

This section describes the server functions exposed by the Energy API. Only one server function is included in the current release of the API:

- `GetServerDateTime()`: returns the current date & time of the server
- `GetErrorDescription()`: returns the error description pertaining to an error code returned by another method
- `SetLogVerbosity()`: Controls output to the standard windows 2000 log file
- `GetDatabaseVersion()`: gets the current release level for the database
- `GetEnergyAPIVersion()`: gets the Energy API current version
- `SetDataCacheRefreshTimeout()`: Sets the time interval in hours after which the data cache will be refreshed
- `GetDataCacheRefreshTimeout()`: Gets the time interval in hours and the date time when the data cache is set to be refreshed.

cUtility.GetServerDateTime()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves the current date and time of the server.

Public Function GetServerDateTime(Optional ByVal p_iCompanyID As Short = 0) As Date

Input Parameters

Input parameter: p_iCompanyID --> Company ID the user is connected to – 0 will return the server date & time where ENERGY_MASTER database is located.

Return Values

Return Type Date = Server date & time (MM/DD/YYYY HH:MM:SS)

.NET Example

The following example shows how to retrieve the server date & time:

```
Dim objUtility as Object
Dim dteServerDateTime As Date
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
dteServerDateTime = objUtility.GetServerDateTime(iCompanyID)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    ' Server Date & Time retrieved
End If
```

XML Example

The following example shows how to retrieve the server date & time:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetServerDateTime>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
    </add:GetServerDateTime>
  </soap:Body>
</soap:Envelope>
```

cUtility.GetErrorDescription()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Retrieves the error description for a given error code

Public Function GetErrorDescription(ByVal p_LErrorCode As Integer) As String

Input Parameters

' Input parameter: p_LErrorCode --> The error code - Long integer

Return Values

' Returns: Error description - Null string if not found

.NET Example

The following example shows how to retrieve an error description:

```
Dim objUtility as Object
Dim sErrorDescription As String
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
sErrorDescription = objUtility.GetErrorDescription(LErrorCode)
'
If Len(Trim$(sErrorDescription)) = 0 Then
    ' Error Code not found -
Else
    ' Success
    ' Debug.Print sErrorDescription
End If
```

XML Example

The following example shows how to retrieve an error description:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetErrorDescription>
      <add:p_LErrorCode>109</add:p_LErrorCode>
    </add:GetErrorDescription>
  </soap:Body>
</soap:Envelope>
```

cUtility.SetLogVerbosity()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This function controls output to the standard windows 2000 log file.

Public Function SetLogVerbosity(ByVal p_sSysUser As String, ByVal p_iLogVerbosity As Short) As Integer

Input Parameters

Input parameters

```
'
' p_sSysUser - Calling program (IVRUser/WebUser)
' p_iLogVerbosity: 0 thru 4, 4 is most output
' 0: normal operation -- Output to log file disabled - *THIS IS THE DEFAULT SETTING*
' 1: normal operation -- severe errors reported
' 2: normal operation -- all errors and warnings and significant events reported
' 3: extended output -- all actions reported
' 4: system debug mode, prints status on all actions in detail
```

Return Values

' Returns: Error code - 0=success, <> 0 → failure

.NET Example

The following example shows how to set the Energy API Log verbosity:

```
Dim objUtility as Object
Dim lErrorCode As Integer
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
lErrorCode = objUtility.SetLogVerbosity(sSysUser, 2)
'
If lErrorCode = 0 Then
    ' Success
Else
    ' Failure
    ' Debug.Print "Error code=" & lErrorCode
End If
```

XML Example

The following example shows how to set the Energy API Log verbosity:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
```

```
<add:SetLogVerbosity>
  <add:p_sSysUser>WebUser</add:p_sSysUser>
  <add:p_iLogVerbosity>3</add:p_iLogVerbosity>
</add:SetLogVerbosity>
</soap:Body>
</soap:Envelope>
```


cUtility.GetDatabaseVersion()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This function returns the current release level of the database.

Public Function GetDatabaseVersion() As String

Input Parameters

Input parameters

'
' p_iCompanyID -- Company ID – Short integer
' p_sSysUser - Calling program (IVRUser/WebUser)

Return Values

' Returns: Database version “Major.Minor”

.NET Example

The following example shows how to retrieve the database version:

```
Dim objUtility as Object
Dim sDBVersion As String
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
sDBVersion = objUtility.GetDatabaseVersion()
'
Debug.Print sDBVersion
```

XML Example

The following example shows how to retrieve the database version:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetDatabaseVersion>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
    </add:GetDatabaseVersion>
  </soap:Body>
</soap:Envelope>
```

cUtility.GetEnergyAPIVersion()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This function returns the Energy API current version.

Public Function GetEnergyAPIVersion() As String

Input Parameters

Input parameters
,
' None

Return Values

' Returns: Energy API version "Major.Minor.Revision"

.NET Example

The following example shows how to retrieve the Energy API version:

```
Dim objUtility as Object
Dim sVersion As String
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
sVersion = objUtility.GetEnergyAPIVersion()
'
Debug.Print sVersion
```

XML Example

The following example shows how to retrieve the Energy API version:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetEnergyAPIVersion/>
  </soap:Body>
</soap:Envelope>
```

cUtility.ClearCacheData()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method clears (deletes) the data cache file

Public Sub ClearCacheData()

Input Parameters

Input parameters

'p_bAllInstances → if true the method will try & delete all instances (local app fodler, assembly folder, windows fodler) of the datacache file, else only the first instance (the used by the application) will be deleted

Return Values

' Returns:None

.NET Example

The following example shows how to clear the data cache:

```
Dim objUtility as Object
'
Set objUtility = CreateObject("EnergyAPI.cUtility")
'
objUtility.ClearCacheData(False)
```

XML Example

The following example shows how to clear the data cache:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ClearCacheData>
      <add:p_bAllInstances>true</add:p_bAllInstances>
    </add:ClearCacheData>
  </soap:Body>
</soap:Envelope>
```

cUtility.SetDataCacheRefreshTimeout()

- **License Required: None**

Description

Sets the data cache refresh timeout interval in hours, this value determines the time interval at which the data cache will be refreshed. Default value is 12 hours.

```
Public Sub SetDataCacheRefreshTimeout(ByVal pHours As Short)
```

Input Parameters

Variable	Data Type	Description
pHours	Short (16 bit integer)	The time interval in hours.

Return Values

Variable	Data Type	Description

.NET Example

The following example shows how to set the data cache timeout.

```
Dim objUtility as Object  
Dim iHours As Short
```

```
Set objUtility = CreateObject("EnergyAPI.cUtility")
```

```
objUtility.SetDataCacheRefreshTimeout(iHours)
```

XML Example

The following example shows how to set the data cache timeout.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:SetDataCacheRefreshTimeout>  
      <add:pHours>7</add:pHours>  
    </add:SetDataCacheRefreshTimeout>  
  </soap:Body>  
</soap:Envelope>
```

cUtility.GetDataCacheRefreshTimeout()

- **License Required: None**

Description

Sets the data cache refresh timeout interval in hours, this value determines the time interval at which the data cache will be refreshed. Default value is 12 hours.

Public Function GetDataCacheRefreshTimeOut(**ByRef** pHours **As Short**) **As DateTime**

Input Parameters

Variable	Data Type	Description
pHours	Short (16 bit integer)	The time interval in hours.

Return Values

Variable	Data Type	Description
	DateTime	The date time after which the data cache is considered state.

.NET Example

The following example shows how to get the data cache timeout.

```
Dim objUtility as Object
Dim iHours As Short
Dim dteVal As DateTime
```

```
Set objUtility = CreateObject("EnergyAPI.cUtility")
```

```
dteVal = objUtility.GetDataCacheRefreshTimeout(iHours)
```

XML Example

The following example shows how to get the data cache timeout.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetDataCacheRefreshTimeOut>
      <add:pHours>0</add:pHours>
    </add:GetDataCacheRefreshTimeOut>
  </soap:Body>
</soap:Envelope>
```

cCustomerLog Class

Overview

This section describes the method(s) that allow the programmer to create a new entry in the customer log . The following method is covered in this section:

- CreateLogEntry(): Creates a log entry
- CreateLogEntryWithAttachment(): Creates a log entry & gives the call the option to save the attachment to the log header
- LoadLogAttachmentBytes():Loads a customer log attachment from the E3 database and returns the binary data in a byte array
- LoadLogAttachmentFile():Loads a customer log attachment from the E3 database and saves the file to local disk.
- SaveLogAttachmentBytes():Saves a customer log attachment to the E3 database using the binary data supplied in the byte array
- SaveLogAttachmentFile():Saves a customer log attachment specified in the pAttachmentPath parameter to the E3 database.
- RemoveLogAttachment():Removes a customer log attachment from the E3 database
- CleanupLocalResource():Cleanups / Deletes the local file after the calling program no longer needs it.

cCustomerLog.CreateLogEntry()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to create a new customer log entry

```
Public Function CreateLogEntry(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_lTemplateID As Integer, _
    Optional ByVal p_iLocationType As Short = 0, _
    Optional ByVal p_iLocation As Short = 0, _
    Optional ByVal p_sFollowUpRequired As String = "N", _
    Optional ByVal p_dteDueDate As Date = "01/01/1900", _
    Optional ByVal p_iFollowUpResID As Short = 0, _
    Optional ByVal p_sNote1 As String = "", _
    Optional ByVal p_sNote2 As String = "", _
    Optional ByVal p_sNote3 As String = "", _
    Optional ByVal p_sNote4 As String = "", _
    Optional ByVal p_sNotePath As String = "", _
    Optional ByVal p_iAlertFlag As Short = 0, _
    Optional ByVal p_vDetails As Object = Nothing) As Integer
```

Input Parameters

```
'Accepts: p_iCompanyID -- Company ID – Short integer
'         p_lAccountNum – Account number without the check digit – Long integer
'         p_sSysUser - Calling program (IVR/WebUser)
'         p_lTemplateID – Template ID (user or system defined template ID)
'         p_iLocationType – Short integer - 0=None, 1=Delivery, 2=Service, 3=WDMS
'         p_iLocation – Short integer – Delivery, Service or WDMS location number
'         p_sFollowUpRequired – Char(1) – FollowUp required Y/N – Default is “N”
'         p_dteDueDate – Date – default = “01/01/1900”
'         p_iFollowUpResID – Short integer – FollowUp Resource ID – Default =0
'         p_sNote1 – CHAR(255) – Log entry note 1 - Default = none
'         p_sNote2 – CHAR(255) – Log entry note 2 - Default = none
'         p_sNote3 – CHAR(255) – Log entry note 3 - Default = none
'         p_sNote4 – CHAR(255) – Log entry note 4 - Default = none
'         p_sNotePath – CHAR(150) – File Attachment full path - Default = none
'         p_iAlertFlag – Short integer – Alert flag 0/1 – Default = 0
'         p_vDetails – 2D array --> p_vDetails(0,ii) = Field sequence #, p_vDetails (0,ii) = value
'         (e.g old value), p_vdetails(1,ii)=Value 1 (e.g. new value)
```

Return Values

```
'Returns: -- 0 If Successful : <> 0 Error Number
'
```

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to create a customer log entry:

```

Sub CreateLogEntry()

    Dim iCompanyID As Short = 4
    Dim lAccountNum As Integer = 1
    Dim sSysUser As String = "WebUser"
    Dim lTemplateID As Integer = 2
    Dim iLocationType As Short = 1 'Optional
    Dim iLocation As Short = 0 'Optional
    Dim sFollowUpRequired As String = "N" 'Optional
    Dim dteDueDate As Date = #12/31/1960# 'Optional
    Dim iFollowUpResID As Short = 0 'Optional
    Dim sNote1 As String = "SomeNote1" 'Optional
    Dim sNote2 As String = "SomeNote2" 'Optional
    Dim sNote3 As String = "SomeNote3" 'Optional
    Dim sNote4 As String = "SomeNote4" 'Optional
    Dim sNotePath As String = "NotePath" 'Optional
    Dim iAlertFlag As Short = 0 'Optional
    Dim p_vDetails(,) As Object = Nothing 'Optional

    Dim oCustLogEntry As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog

    ReDim p_vDetails(5, 6) '5 = Number Of Elements; 6 = Number Of Rows In
LOG_TEMPLATE_DETAIL Where log_template_header_id = Your Template ID

    '***** Field For Sequence #: 1 *****
    p_vDetails(0, 0) = 1 'LOG_TEMPLATE_DETAIL.sequence_no This Is The Key
(Part Of It) To The LOG_TEMPLATE_DETAIL/LOG_DETAIL

    'Fields Bellow Are Optional (255 Max Length) Based On The Template
    '
    ' Data Type Is Determined By
LOG_TEAMPLATE_DETAIL.datatype_id (But Always Passed In As A String
    '
    ' Min/Max Required Are Based On Meta-Data From
LOG_TEMPLATE_DETAIL Table (Validation Left Up To The UI NOT The API)
    p_vDetails(1, 0) = "Some Optional Value Here" 'LOG_DETAIL.value
    p_vDetails(2, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_1
    p_vDetails(3, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_2
    p_vDetails(4, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_3
    p_vDetails(5, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_4

    '***** Field For Sequence #: 1 *****
    'Same For Other Fields. Fields That Are Not Required Need Not Be Added To The
Array (Must Be Resized Accordingly)

    oCustLogEntry.CreateLogEntry(iCompanyID, lAccountNum, sSysUser, lTemplateID,
iLocationType, iLocation, sFollowUpRequired, dteDueDate, iFollowUpResID, sNote1,
sNote2, sNote3, sNote4, sNotePath, iAlertFlag, p_vDetails)

    oCustLogEntry.Dispose()
    oCustLogEntry = Nothing

End Sub

```



```

public void CreateLogEntry()
{
    short iCompanyID = 4;
    int lAccountNum = 1;
    string sSysUser = "WebUser";
    int lTemplateID = 2;
    short iLocationType = 1; //Optional
    short iLocation = 0; //Optional
    string sFollowUpRequired = "N"; //Optional
    System.DateTime dteDueDate = DateTime.Parse("12/31/1960"); //Optional
    short iFollowUpResID = 0; //Optional
    string sNote1 = "SomeNote1"; //Optional
    string sNote2 = "SomeNote2"; //Optional
    string sNote3 = "SomeNote3"; //Optional
    string sNote4 = "SomeNote4"; //Optional
    string sNotePath = "NotePath"; //Optional
    short iAlertFlag = 0; //Optional
    object[,] p_vDetails = null; //Optional
    EnergyAPI.CCcustomerLog oCustLogEntry = new EnergyAPI.CCcustomerLog();

    p_vDetails = new object[6, 7]; //5 = Number Of Elements; 6 = Number Of Rows
    In LOG_TEMPLATE_DETAIL Where log_template_header_id = Your Template ID

    //***** Field For Sequence #: 1 *****
    p_vDetails[0, 0] = 1; //LOG_TEMPLATE_DETAIL.sequence_no This Is The Key (Part
    Of It) To The LOG_TEMPLATE_DETAIL/LOG_DETAIL

    //Fields Bellow Are Optional (255 Max Length) Based On The Template
    //      Data Type Is Determined By
    LOG_TEMPLATE_DETAIL.datatype_id (But Always Passed In As A String
    //      Min/Max Required Are Based On Meta-Data From
    LOG_TEMPLATE_DETAIL Table (Validation Left Up To The UI NOT The API)
    p_vDetails[1, 0] = "Some Optional Value Here"; //LOG_DETAIL.value
    p_vDetails[2, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_1
    p_vDetails[3, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_2
    p_vDetails[4, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_3
    p_vDetails[5, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_4

    //***** Field For Sequence #: 1 *****
    //Same For Other Fields. Fields That Are Not Required Need Not Be Added To
    The Array (Must Be Resized Accordingly)

    oCustLogEntry.CreateLogEntry(iCompanyID, lAccountNum, sSysUser, lTemplateID,
    iLocationType, iLocation, sFollowUpRequired, dteDueDate, iFollowUpResID, sNote1,
    sNote2, sNote3, sNote4, sNotePath, iAlertFlag, p_vDetails);

    oCustLogEntry.Dispose();
    oCustLogEntry = null;
}

```

XML Example

The following example shows how to create a customer log with one detail record.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <soap:Header/>
  <soap:Body>
    <add:CreateLogEntry>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_lAccountNum>102</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lTemplateID>5</add:p_lTemplateID>
      <add:p_iLocationType>1</add:p_iLocationType>
      <add:p_iLocation>1</add:p_iLocation>
      <add:p_sFollowUpRequired>N</add:p_sFollowUpRequired>
      <add:p_dteDueDate>2013-12-31</add:p_dteDueDate>
      <add:p_iFollowUpResID>0</add:p_iFollowUpResID>
      <add:p_sNote1>SmartConnect test text 1</add:p_sNote1>
      <add:p_sNote2>text 2</add:p_sNote2>
      <add:p_sNote3>text 3</add:p_sNote3>
      <add:p_sNote4>text 4</add:p_sNote4>
      <add:p_sNotePath></add:p_sNotePath>
      <add:p_iAlertFlag>0</add:p_iAlertFlag>
      <add:p_s_DSDetails>
        <xs:schema id="DSLogDetails" xmlns=""
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-
com:xml-msdata">
          <xs:element name="DSLogDetails" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
            <xs:complexType>
              <xs:choice minOccurs="0" maxOccurs="unbounded">
                <xs:element name="LogDetails">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="sequence_number" type="xs:int" minOccurs="0" />
                      <xs:element name="field_value" type="xs:string" minOccurs="0" />
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:choice>
            </xs:complexType>
          </xs:element>
        </xs:schema>
        <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
```

```
<DSLogDetails>
  <LogDetails>
    <sequence_number>1</sequence_number>
    <field_value>Yes</field_value>
  </LogDetails>
</DSLogDetails>
</diffgr:diffgram>
</add:p_s_DSDetails>
</add:CreateLogEntry>
</soap:Body>
</soap:Envelope>
```

cCustomerLog.CreateLogEntryWithAttachment()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to create a new customer log entry with the option that the attachment specified in the log entry can be saved to the log header. **NOTE: it is the caller's responsibility to ensure that the attachment is accessible by the SmartConnect web service.**

```
Public Function CreateLogEntryWithAttachment(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_lTemplateID As Integer, _
    ByVal p_iLocationType As Short, _
    ByVal p_iLocation As Short, _
    ByVal p_sFollowUpRequired As String, _
    ByVal p_dteDueDate As Date, _
    ByVal p_iFollowUpResID As Short, _
    ByVal p_sNote1 As String, _
    ByVal p_sNote2 As String, _
    ByVal p_sNote3 As String, _
    ByVal p_sNote4 As String, _
    ByVal p_sNotePath As String, _
    ByVal p_iAlertFlag As Short, _
    ByVal p_vDetails As Object, _
    ByVal pSaveAttachmentToDb As Boolean) As Integer
```

Input Parameters

```
'Accepts: p_iCompanyID -- Company ID – Short integer
'      p_lAccountNum – Account number without the check digit – Long integer
'      p_sSysUser - Calling program (IVR/WebUser)
'      p_lTemplateID – Template ID (user or system defined template ID)
'      p_iLocationType – Short integer - 0=None, 1=Delivery, 2=Service, 3=WDMS
'      p_iLocation – Short integer – Delivery, Service or WDMS location number
'      p_sFollowUpRequired – Char(1) – FollowUp required Y/N – Default is “N”
'      p_dteDueDate – Date – default = “01/01/1900”
'      p_iFollowUpResID – Short integer – FollowUp Resource ID – Default =0
'      p_sNote1 – CHAR(255) – Log entry note 1 - Default = none
'      p_sNote2 – CHAR(255) – Log entry note 2 - Default = none
'      p_sNote3 – CHAR(255) – Log entry note 3 - Default = none
'      p_sNote4 – CHAR(255) – Log entry note 4 - Default = none
'      p_sNotePath – CHAR(150) – File Attachment full path - Default = none
'      p_iAlertFlag – Short integer – Alert flag 0/1 – Default = 0
'      p_vDetails – 2D array --> p_vDetails(0,ii) = Field sequence #, p_vDetails (0,ii) = value
'      (e.g old value), p_vdetails(1,ii)=Value 1 (e.g. new value)
'      pSaveAttachmentToDb – Boolean – If set to true the attachment specified in “p_sNotePath” will be saved to
the E3 database
```

Return Values

```
'Returns: -- 0 If Successful : <> 0 Error Number
```

Return code	Description
96	Unknown Error Occurred.
122	Invalid parameters specified

.NET Example

The following example shows how to create a customer log entry with an attachment:

```

Sub CreateLogEntryWithAttachment()

    Dim iCompanyID As Short = 4
    Dim lAccountNum As Integer = 1
    Dim sSysUser As String = "WebUser"
    Dim lTemplateID As Integer = 2
    Dim iLocationType As Short = 1
    Dim iLocation As Short = 0
    Dim sFollowUpRequired As String = "N"
    Dim dteDueDate As Date = #12/31/1960#
    Dim iFollowUpResID As Short = 0
    Dim sNote1 As String = "SomeNote1"
    Dim sNote2 As String = "SomeNote2"
    Dim sNote3 As String = "SomeNote3"
    Dim sNote4 As String = "SomeNote4"
    Dim sNotePath As String = "PathAccessibleToSmartConnect\AttachmentName.txt"
    Dim iAlertFlag As Short = 0
    Dim p_vDetails(,) As Object = Nothing
    Dim saveAttachmentToDb As Boolean = True

    Dim oCustLogEntry As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog

    ReDim p_vDetails(5, 6) '5 = Number Of Elements; 6 = Number Of Rows In
LOG_TEMPLATE_DETAIL Where log_template_header_id = Your Template ID

    '***** Field For Sequence #: 1 *****
    p_vDetails(0, 0) = 1 'LOG_TEMPLATE_DETAIL.sequence_no          This Is The Key
(Part Of It) To The LOG_TEMPLATE_DETAIL/LOG_DETAIL

    'Fields Bellow Are Optional (255 Max Length) Based On The Template
    '                               Data Type Is Determined By
LOG_TEMPLATE_DETAIL.datatype_id (But Always Passed In As A String
    '                               Min/Max Required Are Based On Meta-Data From
LOG_TEMPLATE_DETAIL Table (Validation Left Up To The UI NOT The API)
    p_vDetails(1, 0) = "Some Optional Value Here" 'LOG_DETAIL.value
    p_vDetails(2, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_1
    p_vDetails(3, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_2
    p_vDetails(4, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_3
    p_vDetails(5, 0) = "Some Other Optional Value Here" 'LOG_DETAIL.value_4

    '***** Field For Sequence #: 1 *****
    'Same For Other Fields. Fields That Are Not Required Need Not Be Added To The
Array (Must Be Resized Accordingly)

    oCustLogEntry.CreateLogEntryWithAttachment(iCompanyID, lAccountNum, sSysUser,
lTemplateID, iLocationType, iLocation, sFollowUpRequired, dteDueDate,

```

```
iFollowUpResID, sNote1, sNote2, sNote3, sNote4, sNotePath, iAlertFlag, p_vDetails,
saveAttachmentToDb)
```

```
oCustLogEntry.Dispose()
oCustLogEntry = Nothing
```

```
End Sub
```

```
public void CreateLogEntryWithAttachment()
{
    short iCompanyID = 4;
    int lAccountNum = 1;
    string sSysUser = "WebUser";
    int lTemplateID = 2;
    short iLocationType = 1;
    short iLocation = 0;
    string sFollowUpRequired = "N";
    System.DateTime dteDueDate = DateTime.Parse("12/31/1960");
    short iFollowUpResID = 0;
    string sNote1 = "SomeNote1";
    string sNote2 = "SomeNote2";
    string sNote3 = "SomeNote3";
    string sNote4 = "SomeNote4";
    string sNotePath = "PathAccessibleToSmartConnect\\AttachmentName.txt";
    short iAlertFlag = 0;
    object[,] p_vDetails = null;
    bool saveAttachmentToDb = true;

    EnergyAPI.CCustomerLog oCustLogEntry = new EnergyAPI.CCustomerLog();

    p_vDetails = new object[6, 7]; //5 = Number Of Elements; 6 = Number Of Rows
    In LOG_TEMPLATE_DETAIL Where log_template_header_id = Your Template ID

    //***** Field For Sequence #: 1 *****
    p_vDetails[0, 0] = 1; //LOG_TEMPLATE_DETAIL.sequence_no This Is The Key (Part
    Of It) To The LOG_TEMPLATE_DETAIL/LOG_DETAIL

    //Fields Bellow Are Optional (255 Max Length) Based On The Template
    //                               Data Type Is Determined By
    LOG_TEMPLATE_DETAIL.datatype_id (But Always Passed In As A String
    //                               Min/Max Required Are Based On Meta-Data From
    LOG_TEMPLATE_DETAIL Table (Validation Left Up To The UI NOT The API)
    p_vDetails[1, 0] = "Some Optional Value Here"; //LOG_DETAIL.value
    p_vDetails[2, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_1
    p_vDetails[3, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_2
    p_vDetails[4, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_3
    p_vDetails[5, 0] = "Some Other Optional Value Here"; //LOG_DETAIL.value_4

    //***** Field For Sequence #: 1 *****
    //Same For Other Fields. Fields That Are Not Required Need Not Be Added To
    The Array (Must Be Resized Accordingly)

    oCustLogEntry.CreateLogEntryWithAttachment(iCompanyID, lAccountNum, sSysUser,
    lTemplateID, iLocationType, iLocation, sFollowUpRequired, dteDueDate,
```

```
iFollowUpResID, sNote1, sNote2, sNote3, sNote4, sNotePath, iAlertFlag, p_vDetails,
saveAttachmentToDb);
```

```
oCustLogEntry.Dispose();
oCustLogEntry = null;
```

```
}
```

XML Example

The following example shows how to create a customer log with one detail record.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <soap:Header/>
  <soap:Body>
    <add:CreateLogEntry>
      <add:p_iCompanyID>1122</add:p_iCompanyID>
      <add:p_lAccountNum>102</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lTemplateID>5</add:p_lTemplateID>
      <add:p_iLocationType>1</add:p_iLocationType>
      <add:p_iLocation>1</add:p_iLocation>
      <add:p_sFollowUpRequired>N</add:p_sFollowUpRequired>
      <add:p_dteDueDate>2013-12-31</add:p_dteDueDate>
      <add:p_iFollowUpResID>0</add:p_iFollowUpResID>
      <add:p_sNote1>SmartConnect test text 1</add:p_sNote1>
      <add:p_sNote2>text 2</add:p_sNote2>
      <add:p_sNote3>text 3</add:p_sNote3>
      <add:p_sNote4>text 4</add:p_sNote4>
      <add:p_sNotePath></add:p_sNotePath>
      <add:p_iAlertFlag>0</add:p_iAlertFlag>
    <add:p_s_DSDetails>
      <xs:schema id="DSLogDetails" xmlns=""
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-
com:xml-msdata">
        <xs:element name="DSLogDetails" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
          <xs:complexType>
            <xs:choice minOccurs="0" maxOccurs="unbounded">
              <xs:element name="LogDetails">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="sequence_number" type="xs:int" minOccurs="0" />
                    <xs:element name="field_value" type="xs:string" minOccurs="0" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:choice>
```

```
</xs:complexType>
</xs:element>
</xs:schema>
<diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
  <DSLogDetails>
    <LogDetails>
      <sequence_number>1</sequence_number>
      <field_value>Yes</field_value>
    </LogDetails>
  </DSLogDetails>
</diffgr:diffgram>
</add:p_s_DSDetails>
  <add:pSaveAttachmentToDb>0</add:pSaveAttachmentToDb>
</add:CreateLogEntry>
</soap:Body>
</soap:Envelope>
```


cCustomerLog.LoadLogAttachmentBytes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to return an existing E3 Customer log entry's attachment in binary format.

Method Signature:

```
Public Function LoadLogAttachmentBytes(ByVal pCompanyId As Short, ByVal pSysUser As String, ByVal pAccountNum As Integer, ByVal pLogHeaderId As Integer, ByVal pFollowupResolvedAttachment As Boolean, ByVal pSequenceNo As Short, ByVal pRepetitionNo As Short, ByRef pAttachmentPath As String, ByRef pAttachmentExtension As String) As Byte()
```

Input Parameters

Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit integer)	E3 Account number without check digit
	pLogHeaderId	Integer (32 bit integer)	The Header ID of the log entry from which the attachment needs to be loaded
	pFollowupResolvedAttachment	Boolean	If set to true the data returned will be the follow-up resolved attachment, else the standard attachment on the log header
	pSequenceNo	Short (16 bit integer)	The Sequence Number of the detail record for which the attachment needs to be loaded. Pass 0 to load attachment from Log Header
	pRepetitionNo	Short (16 bit integer)	Identifies which detail entry to return, used for templates that allow repetition of details, default value = 1 for templates that don't allow repetition

Return Values

	pAttachmentPath	String	The log Attachment path that is stored in E3, only applies to an attachment on the header
	pAttachmentExtension	String	The file extension of the log attachment that is stored in E3
Return Type: Byte()			
			Byte array containing the attachment in binary format

Return Codes

Return code	Description

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadLogAttachmentBytes>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLogHeaderId>123</add:pLogHeaderId>
      <add:pFollowupResolvedAttachment>true</add:pFollowupResolvedAttachment>
      <add:pSequenceNo>1</add:pSequenceNo>
      <add:pRepetitionNo>1</add:pRepetitionNo>
      <add:pAttachmentPath></add:pAttachmentPath>
      <add:pAttachmentExtension></add:pAttachmentExtension>
    </add:LoadLogAttachmentBytes>
  </soap:Body>
</soap:Envelope>
```

cCustomerLog.LoadLogAttachmentFile()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

Loads a customer log attachment from the E3 database and saves the file to local disk.

NOTE: A call to "CleanupLocalResource" should be made to cleanup resources

Method Signature:

```
Public Function LoadLogAttachmentFile(ByVal pCompanyId As Short, ByVal pSysUser As String, ByVal pAccountNum As Integer, ByVal pLogHeaderId As Integer, ByVal pFollowupResolved Attachment As Boolean, ByVal pSequenceNo As Short, ByVal pRepetitionNo As Short, ByRef pAttachmentPath As String, ByRef pAttachmentExtension As String, ByRef pOutputPath As String) As Long
```

Input Parameters

Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit integer)	E3 Account number without check digit
	pLogHeaderId	Integer (32 bit integer)	The Header ID of the log entry from which the attachment needs to be loaded
	pFollowupResolvedAttachment	Boolean	If set to true the data returned will be the follow-up resolved attachment, else the standard attachment on the log header
	pSequenceNo	Short (16 bit integer)	The Sequence Number of the detail record for which the attachment needs to be loaded. Pass 0 to load attachment from Log Header
	pRepetitionNo	Short (16 bit integer)	Identifies which detail entry to return, used for templates that allow repetition of details, default value = 1 for templates that don't allow repetition

Return Values

	pAttachmentPath	String	The log Attachment path that is stored in E3, only applies to an attachment on the header
	pAttachmentExtension	String	The file extension of the log attachment that is stored in E3
	pOutputPath	String	The path where the Customer Log Attachment will be saved to, if a path is not specified the system will auto generate a path. NOTE: If path is specified it the

			caller's responsibility to ensure that SmartConnect as sufficient access to access & write to the path
Return Type: Byte()			
			Byte array containing the attachment in binary format

Return Codes

Return code	Description

XML document:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadLogAttachmentFile>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLogHeaderId>123</add:pLogHeaderId>
      <add:pFollowupResolvedAttachment>true</add:pFollowupResolvedAttachment>
      <add:pSequenceNo>1</add:pSequenceNo>
      <add:pRepetitionNo>1</add:pRepetitionNo>
      <add:pAttachmentPath></add:pAttachmentPath>
      <add:pAttachmentExtension></add:pAttachmentExtension>
      <add:pOutputPath></add:pOutputPath>
    </add:LoadLogAttachmentFile>
  </soap:Body>
</soap:Envelope>

```

cCustomerLog.SaveLogAttachmentBytes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

Saves a customer log attachment to the E3 database using the binary data supplied in the byte array

Method Signature:

```
Public Function SaveLogAttachmentBytes(ByVal pCompanyId As Short, ByVal pSysUser As String,
ByVal pAccountNum As Integer, ByVal pLogHeaderId As Integer, ByVal pFollowupResolvedAttachment As Boolean,
ByVal pSequenceNo As Short, ByVal pRepetitionNo As Short,
ByVal pAttachmentPath As String, ByVal pFileBytes As Byte()) As Integer
```

Input Parameters

Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit integer)	E3 Account number without check digit
	pLogHeaderId	Integer (32 bit integer)	The Header ID of the log entry from which the attachment needs to be saved
	pFollowupResolvedAttachment	Boolean	If set to true the attachment data will be saved to the follow-up resolved attachment, else the standard attachment on the log header
	pSequenceNo	Short (16 bit integer)	The Sequence Number of the detail record for which the attachment needs to be saved. Pass 0 to save attachment from Log Header
	pRepetitionNo	Short (16 bit integer)	Identifies which detail entry to which the attachment will be saved, used for templates that allow repetition of details, default value = 1 for templates that don't allow repetition
	pAttachmentPath	String	The log Attachment path that is stored in E3.
	pFileBytes	Byte()	A Byte Array of the Log Attachment that will be saved to the Customer Log Record

Return Values

Return Type: Integer		
Error Code	32 bit integer	Non-Zero Error if the operation cannot be completed

Return Codes

Return code	Description
122	Invalid input parameters.
1011	There is a problem updating data in a table. The process did not complete. Call your System Administrator.
1496	File attachments are not permitted for this log type.
1499	The modification of a system generated log is not permitted.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveLogAttachmentBytes>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLogHeaderId>123</add:pLogHeaderId>
      <add:pFollowupResolvedAttachment>true</add:pFollowupResolvedAttachment>
      <add:pSequenceNo>1</add:pSequenceNo>
      <add:pRepetitionNo>1</add:pRepetitionNo>
      <add:pAttachmentPath>SomeDir\SomeFile.txt</add:pAttachmentPath>
      <add:pFileBytes>cid:216454850456</add:pFileBytes>
    </add:SaveLogAttachmentBytes>
  </soap:Body>
</soap:Envelope>
```

cCustomerLog.SaveLogAttachmentFile()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

Saves a customer log attachment specified in the pAttachmentPath parameter to the E3 database.

NOTE: A call to "CleanupLocalResource" should be made to cleanup resources

NOTE: it is the caller's responsibility to ensure that the attachment is accessible by the SmartConnect web service.

Method Signature:

Public Function SaveLogAttachmentFile(**ByVal** pCompanyId **As Short**, **ByVal** pSysUser **As String**, **ByVal** pAccountNum **As Integer**, **ByVal** pLogHeaderId **As Integer**, **ByVal** pFollowupResolved Attachment **As Boolean**, **ByVal** pSequenceNo **As Short**, **ByVal** pRepetitionNo **As Short**, **ByVal** pAttachmentPath **As String**) **As Integer**

Input Parameters

Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit integer)	E3 Account number without check digit
	pLogHeaderId	Integer (32 bit integer)	The Header ID of the log entry from which the attachment needs to be saved
	pFollowupResolvedAttachment	Boolean	If set to true the attachment data will be saved to the follow-up resolved attachment, else the standard attachment on the log header
	pSequenceNo	Short (16 bit integer)	The Sequence Number of the detail record for which the attachment needs to be saved. Pass 0 to save attachment from Log Header
	pRepetitionNo	Short (16 bit integer)	Identifies which detail entry to which the attachment will be saved, used for templates that allow repetition of details, default value = 1 for templates that don't allow repetition
	pAttachmentPath	String	The path to the file which will be saved to the Customer Log Attachment.

Return Values

Return Type: Integer		
Error Code	32 bit integer	Non-Zero Error if the operation cannot be

		completed

Return code	Description
122	Invalid input parameters
1011	There is a problem updating data in a table. The process did not complete. Call your System Administrator.
1496	File attachments are not permitted for this log type.
1499	The modification of a system generated log is not permitted.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveLogAttachmentFile>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLogHeaderId>123</add:pLogHeaderId>
      <add:pFollowupResolvedAttachment>true</add:pFollowupResolvedAttachment>
      <add:pSequenceNo>1</add:pSequenceNo>
      <add:pRepetitionNo>1</add:pRepetitionNo>
      <add:pAttachmentPath>SomeDir\SomeFile.txt</add:pAttachmentPath>
    </add:SaveLogAttachmentFile>
  </soap:Body>
</soap:Envelope>
```


cCustomerLog.RemoveLogAttachment()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

Removes a customer log attachment from the E3 database

Method Signature:

```
Public Function RemoveLogAttachment(ByVal pCompanyId As Short, ByVal pSysUser As String, ByVal pAccountNum As Integer, ByVal pLogHeaderId As Integer, ByVal pFollowupResolvedAttachment As Boolean, ByVal pSequenceNo As Short, ByVal pRepetitionNo As Short) As Integer
```

Input Parameters

Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit integer)	E3 Account number without check digit
	pLogHeaderId	Integer (32 bit integer)	The Header ID of the log entry from which the attachment needs to be removed
	pFollowupResolvedAttachment	Boolean	If set to true the follow-up resolved attachment will be removed, else the standard attachment on the log header
	pSequenceNo	Short (16 bit integer)	The Sequence Number of the detail record for which the attachment needs to be removed. Pass 0 to load attachment from Log Header
	pRepetitionNo	Short (16 bit integer)	Identifies which detail entry to which the attachment will be removed, used for templates that allow repetition of details, default value = 1 for templates that don't allow repetition

Return Values

Return Type: Integer		
Error Code	32 bit integer	Non-Zero Error if the operation cannot be completed

Return Codes

Return code	Description
122	Invalid input parameters

1011	There is a problem updating data in a table. The process did not complete. Call your System Administrator.
1496	File attachments are not permitted for this log type.
1499	The modification of a system generated log is not permitted.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:RemoveLogAttachment>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLogHeaderId>123</add:pLogHeaderId>
      <add:pFollowupResolvedAttachment>true</add:pFollowupResolvedAttachment>
      <add:pSequenceNo>1</add:pSequenceNo>
      <add:pRepetitionNo>1</add:pRepetitionNo>
    </add:RemoveLogAttachment>
  </soap:Body>
</soap:Envelope>
```

cCustomerLog.CleanupLocalResource()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

Cleanups / Deletes the local file after the calling program no longer needs it.

NOTE: it is the caller's responsibility to ensure that the path is accessible by the SmartConnect web service.

Method Signature:

```
Public Sub CleanupLocalResource(ByVal pSysUser As String, ByVal pResourcePath As String)
```

Input Parameters

Accepts	Variable	Data type	Description
	pSysUser	String	Calling program (IVRUser/WebUser)
	pResourcePath	String	Path for the local file that needs to be deleted

Return Values

Return Type:

Return code	Description

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CleanupLocalResource>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pResourcePath>SomeDir\SomeFile.doc</add:pResourcePath>
    </add:CleanupLocalResource>
  </soap:Body>
</soap:Envelope>
```

cCustomerLog.FindCustomerLogEntries()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to find customer log entries give a selection criteria. The functionality mimicks the behavior seen in E3.

Public Function FindCustomerLogEntries(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pSelectionCriteria As [CustomerLogSelectionCriteria](#), ByRef pErrorCode As Integer) As [DCEnergyAPI.CustomerLog.CustomerLogHeader\(\)](#)

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pSelectionCriteria	CustomerLogSelectionCriteria	Class object
pErrorCode	Integer (32bit Integer)	0: no error <>0: error * <i>Utility.GetErrorDescription()</i> to retrieve the description.*

Return Values

Array of type [CustomerLogHeader](#)

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:FindCustomerLogEntries>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <!--Optional:-->
      <add:pSelectionCriteria>
        <!--Optional:-->
        <add:TemplatesToSearch>
          <!--Zero or more repetitions:-->
          <add:int>136</add:int>
        </add:TemplatesToSearch>
        <add:IncludeGeneralCustomer>true</add:IncludeGeneralCustomer>
        <add:IncludeDeliveryLocations>false</add:IncludeDeliveryLocations>
        <add:IncludeServiceLocations>false</add:IncludeServiceLocations>
        <add:IncludeWDMSLocations>false</add:IncludeWDMSLocations>
        <add:LogsCreatedOrModifiedFromDate>1900-01-01</add:LogsCreatedOrModifiedFromDate>
        <add:LogsCreatedOrModifiedToDate>1900-01-01</add:LogsCreatedOrModifiedToDate>
        <add:LogsCreatedOrModifiedPrevMonths>0</add:LogsCreatedOrModifiedPrevMonths>
        <add:LogsCreatedOrModifiedPrevDays>0</add:LogsCreatedOrModifiedPrevDays>
        <add:FollowupStatusNotRequired>true</add:FollowupStatusNotRequired>
        <add:FollowupStatusRequired>true</add:FollowupStatusRequired>
        <add:FollowupStatusRequiredDate>2020-02-23</add:FollowupStatusRequiredDate>
        <add:FollowupStatusResolved>true</add:FollowupStatusResolved>
        <!--Optional:-->
        <add:FollowupResources>
        </add:FollowupResources>
        <add:IncludeAllAlerts>true</add:IncludeAllAlerts>
        <add:IncludeAlertedOnly>false</add:IncludeAlertedOnly>
        <add:IncludeNotAlertedOnly>false</add:IncludeNotAlertedOnly>
        <!--Optional:-->
        <add:FullAccount></add:FullAccount>
        <!--Optional:-->
        <add:DivisionsToInclude>
          <!--Zero or more repetitions:-->
          <!-- <add:int>?</add:int>-->
        </add:DivisionsToInclude>
        <!--Optional:-->
        <add:CollectorsToInclude>
          <!--Zero or more repetitions:-->
        </add:CollectorsToInclude>
      </add:pSelectionCriteria>
    </add:FindCustomerLogEntries>
  </soap:Body>
</soap:Envelope>
```

```
</add:pSelectionCriteria>
<add:pErrorCode>0</add:pErrorCode>
</add:FindCustomerLogEntries>
</soap:Body>
</soap:Envelope>
```

VB.NET Example:

```
Private Sub btnLoadLogs_Click(sender As Object, e As EventArgs) Handles btnLoadLogs.Click
    Dim customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
    Dim myLogs As DCEnergyAPI.CustomerLog.CustomerLogHeader()
    Dim selectionCrit As New EnergyAPI.CustomerLogSelectionCriteria
    Dim errorCode As Integer = 0
    Dim templates As Integer() = New Integer() {136, 137}
    Dim fullAccountNum As String = "1008"
    Dim iCompanyID As Short = 2
    '
    With selectionCrit
        .FollowupStatusNotRequired = True
        .FollowupStatusResolved = True
        .TemplatesToSearch = templates
        .FullAccount = fullAccountNum
    End With
    myLogs = customerLog.FindCustomerLogEntries(iCompanyID, "WebUser", selectionCrit,
errorCode)
    DataGridView1.DataSource = myLogs
End Sub
```

Search log entries for one account (1008) and templates 136 and 137 only.

cCustomerLog.LoadCustomerLogDetails()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to retrieve customer log details for a given log header record if details exist.

Public Function LoadCustomerLogDetails(ByVal pCompanyID As Short, ByVal pSysUser As String, ByVal pAccountNum As Integer, ByVal pLogHeaderID As Integer, ByVal pLogTemplateHeaderID As Integer, ByRef pErrorCode As Integer) As

[DCEnergyAPI.CustomerLog.CustomerLogDetail\(\)](#)

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pAccountNum	Integer (32bit Integer)	Account # (no checkdigit)
pLogHeaderID	Integer (32bit Integer)	Log header id
pLogTemplateHeaderID	Integer (32bit Integer)	Log template id
pErrorCode	Integer (32bit Integer)	0: no error <>0: error# * <i>Utility.GetErrorDescription()</i> to retrieve the description.*

Return Values

Array of type [CustomerLogDetail](#) or Empty Array (not all log templates save detail information)

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerLogDetails>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>4572</add:pAccountNum>
      <add:pLogHeaderID>1999999959</add:pLogHeaderID>
      <add:pLogTemplateHeaderID>136</add:pLogTemplateHeaderID>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCustomerLogDetails>
  </soap:Body>
</soap:Envelope>
```

VB.NET Example:

```
Private Sub btnLoadDetails_Click(sender As Object, e As EventArgs) Handles btnLoadDetails.Click
  Dim customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
  Dim myDetails As DCEnergyAPI.CustomerLog.CustomerLogDetail()
  Dim errorCode As Integer = 0
  Dim AccountNum As String = "100" -- without the checkdigit
  Dim logHeaderID as Integer = 1999999352
  Dim logTemplateID as Integer = 136
  Dim iCompanyID as Short = 2
  myDetails = customerLog.LoadCustomerLogDetails(iCompanyID, "WebUser", CInt(AccountNum),
logHeaderID, logTemplateID, errorCode)
  DataGridView2.DataSource = myDetails
End Sub
```


cCustomerLog.UpdateCustomerLogEntry()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to update a customer log header **and/or** details records for one given log header record.

Public Function UpdateCustomerLogEntry(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pCustomerLogHeader As DCEnergyAPI.CustomerLog.CustomerLogHeader, ByRef pCustomerLogDetails As DCEnergyAPI.CustomerLog.CustomerLogDetail(), ByRef pErrorCount As Integer) As Boolean

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pCustomerLogHeader	CustomerLogHeader	Log header information
pCustomerLogDetails	Array of CustomerLogDetail	Log detail information
pErrorCount*	Integer (32bit Integer)	0: no errors <>0: # of errors that occurred.

*pErrorCount**: Unlike the *pErrorCode* parameter a non-zero value means that one or more errors occurred while trying to save the header and/or detail record... Interrogate **pCustomerLogHeader.ErrorCode** and iterate through each **pCustomerLogDetails.ErrorCode** (when applicable) to determine which record/field failed. You can make a call to **Utility.GetErrorDescription()** to retrieve the description for **.ErrorCode**.

ATTENTION The expectation is the FindCustomerLogEntries() was used to retrieve existing data. Regardless, it is expected that when calling UpdateCustomerLogEntry() all pertinent data (modified & unmodified) must be included.

Return Values

Boolean

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:UpdateCustomerLogEntry>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <!--Optional:-->
      <add:pCustomerLogHeader>
        <add:AccountNum>100498</add:AccountNum>
        <!--Optional:-->
        <add:FullAccount>1004986</add:FullAccount>
        <add:LogHeaderID>1999999991</add:LogHeaderID>
        <add:LogTemplateHeaderID>136</add:LogTemplateHeaderID>
        <add:LocationType>0</add:LocationType>
        <add:LocationNum>0</add:LocationNum>
        <!--Optional:-->
        <add:FollowupRequired>N</add:FollowupRequired>
        <add:DueDate>1900-01-01</add:DueDate>
        <add:FollowupResource>0</add:FollowupResource>
        <add:ResolvedDate>1900-01-01</add:ResolvedDate>
        <add:AlertFlag>>false</add:AlertFlag>
        <!--Optional:-->
        <add>Note>THIS IS MY NEW NOTE</add>Note>
        <!--Optional:-->
        <add:ResolvedNote></add:ResolvedNote>
        <!--Optional:-->
        <add:LastMaintenanceUserID>WebUser</add:LastMaintenanceUserID>
        <!--Optional:-->
        <add:LastMaintenanceDate>Feb 25 2016 2:29:13:016PM</add:LastMaintenanceDate>
        <add:AllowAttachments>>true</add:AllowAttachments>
        <add:AllowModificationToExistingNotes>0</add:AllowModificationToExistingNotes>
        <!--Optional:-->
        <add:AttachmentFileType>xls</add:AttachmentFileType>
        <!--Optional:-->
        <add>NoteAttachmentBinaryFormat></add>NoteAttachmentBinaryFormat>
        <!--Optional:-->
        <add>NoteAttachmentFullPath></add>NoteAttachmentFullPath>
        <!--Optional:-->
        <add>NoteAttachmentBinary></add>NoteAttachmentBinary>
        <add>NoteAttachmentRemove>>false</add>NoteAttachmentRemove>
        <add>NoteAttachmentSaveAsShortCut>>false</add>NoteAttachmentSaveAsShortCut>
        <!--Optional:-->
```

```
<add:ResolvedAttachmentBinaryFormat></add:ResolvedAttachmentBinaryFormat>
<!--Optional:-->
<add:ResolvedAttachmentFullPath></add:ResolvedAttachmentFullPath>
<!--Optional:-->
<add:ResolvedAttachmentBinary></add:ResolvedAttachmentBinary>
<add:ResolvedAttachmentRemove>>false</add:ResolvedAttachmentRemove>
<add:ResolvedAttachmentSaveAsShortCut>>false</add:ResolvedAttachmentSaveAsShortCut>
<!--Optional:-->
<add:CreatedBy>david_new</add:CreatedBy>
<add:CreateDate>2016-02-19</add:CreateDate>
<add:HeaderRecordUpdated>>false</add:HeaderRecordUpdated>
<add:SystemDefined>>false</add:SystemDefined>
<add:ErrorCode>0</add:ErrorCode>
</add:pCustomerLogHeader>

<!--Optional:-->
<add:pCustomerLogDetails>
  <!--Zero or more repetitions:-->
  <add:CustomerLogDetail>
    <add:SequenceNo>1</add:SequenceNo>
    <add:RepetitionNo>1</add:RepetitionNo>
    <!--Optional:-->
    <add:Label>regular field</add:Label>
    <add:DataTypeID>3</add:DataTypeID>
    <add:Length>0</add:Length>
    <!--Optional:-->
    <add:Minimum>0</add:Minimum>
    <!--Optional:-->
    <add:Maximum></add:Maximum>
    <add:Required>>false</add:Required>
    <add:ModifyBy>1</add:ModifyBy>
    <!--Optional:-->
    <add:FieldList></add:FieldList>
    <!--Optional:-->
    <add:FileAttachmentBinaryFormat></add:FileAttachmentBinaryFormat>
    <add:FileAttachmentRemove>>false</add:FileAttachmentRemove>
    <!--Optional:-->
    <add:FileAttachmentBinary></add:FileAttachmentBinary>
    <!--Optional:-->
    <add:FieldListDisplay></add:FieldListDisplay>
    <!--Optional:-->
    <add:OldValue></add:OldValue>
    <!--Optional:-->
    <add:NewValue>24</add:NewValue>
    <add:ErrorCode>0</add:ErrorCode>
  </add:CustomerLogDetail>
</add:pCustomerLogDetails>
<add:pErrorCount>0</add:pErrorCount>
```

```

    </add:UpdateCustomerLogEntry>
  </soap:Body>
</soap:Envelope>

```

VB.NET Example:

NOTE This is a simple example that assumes that ONE header record will be returned from the *FindCustomerLogEntries* and the only data we want to update is the 'Note' on the header record and the 3rd detail.

```

Dim customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
Dim myLogs As DCEnergyAPI.CustomerLog.CustomerLogHeader()
Dim selectionCrit As New EnergyAPI.CustomerLogSelectionCriteria
Dim errorCode As Integer = 0
Dim templates As Integer() = New Integer() {136}
Dim iCompanyID As Short = 2
Dim fullAccountNum As String = "1008"

Dim myDetails As DCEnergyAPI.CustomerLog.CustomerLogDetail()

'
With selectionCrit
    .FollowupStatusNotRequired = True
    .FollowupStatusResolved = True
    .TemplatesToSearch = templates
    .FullAccount = fullAccountNum
End With
myLogs = customerLog.FindCustomerLogEntries(iCompanyID, "WebUser", selectionCrit,
errorCode)

myDetails = customerLog.LoadCustomerLogDetails(iCompanyID, "WebUser",
myLogs(0).AccountNum, myLogs(0).LogHeaderID, myLogs(0).LogTemplateID, errorCode)

' one header and 4 details
' modify the note on the header record
myLogs(0).Note = "This note was entered using SmartConnect API"

' modify the 3rd detail label
myDetails(2).NewValue = "12345"

dim errorCount as Integer =0
dim retcode as Boolean = False
retcode = customerLog.UpdateCustomerLogEntry
(iCompanyID,"WebUser",myLogs,myDetails,errorCount)

```

```
IF errorCount > 0 then
    '
    Dim header as DCEnergyAPI.CustomerLog.CustomerLogHeader
    For Each header in myLogs
        IF header.ErrorCode <> 0 then
            ' retrieve and display error description using
utility.GetErrorDescription ()
            ' take action
        End If
    Next
    '

    Dim detail as DCEnergyAPI.CustomerLog.CustomerLogDetail
    For Each detail in myDetails
        IF detail.ErrorCode <> 0 then
            ' retrieve and display error description using
utility.GetErrorDescription ()
            ' take action
        End If
    Next

End if
```

cCustomerLog.LoadCustomerLogTemplates()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to retrieve list of valid customer log templates for display/search criteria.

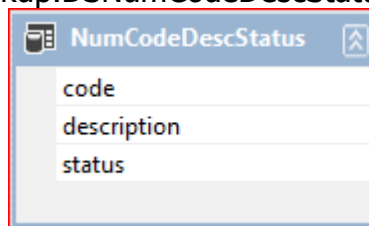
Public Function LoadCustomerLogTemplates(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As DCEnergyAPI.LookUp.DSNumCodeDescStatus

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pErrorCode	Integer (32bit Integer)	0: no error <>0: error# * <i>Utility.GetErrorDescription()</i> to retrieve the description.*

Return Values

DataSet of type DCEnergyAPI.Lookup.DSNumCodeDescStatus



code: what is saved to the database

description: what should be displayed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerLogTemplates>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCustomerLogTemplates>
  </soap:Body>
</soap:Envelope>
```

VB.NET Example

```
Private Sub btnLoadTemplates_Click(sender As Object, e As EventArgs) Handles
btnLoadTemplates.Click
```

```
    Using customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
        Dim errorCode As Integer = 0
        Dim iCompanyID as Short =2
        dgvLookups.DataSource = customerLog.LoadCustomerLogTemplates(iCompanyID, "WebUser",
errorCode).Tables(0)
    End Using
End Sub
```

```
dgvLookups = MS DataGridView
```

cCustomerLog.LoadCustomerLogFollowupResources()

- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Create Log Entries Business Edition

Description

This function can be invoked to retrieve list of valid follow-up resources for display/search criteria.

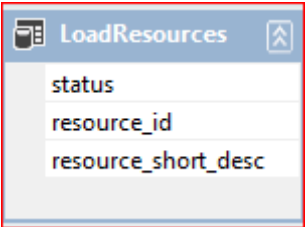
Public Function LoadCustomerLogFollowupResources(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As DCEnergyAPI.LookUp.DSLoadResources

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pErrorCode	Integer (32bit Integer)	0: no error <>0: error# * Utility.GetErrorDescription() to retrieve the description.*

Return Values

DataSet of type DCEnergyAPI.Lookup.DSLoadResources



resource_id: what is saved to the database

resource_short_desc: what should be displayed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerLogFollowupResources>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCustomerLogFollowupResources>
  </soap:Body>
</soap:Envelope>
```

VB.NET Example:

```
Private Sub btnResources_Click(sender As Object, e As EventArgs) Handles btnResources.Click
    Using customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
        Dim errorCode As Integer = 0
        Dim iCompanyID as Short = 2
        dgvLookups.DataSource = customerLog.LoadCustomerLogFollowupResources(iCompanyID,
"WebUser", errorCode).Tables(0)
    End Using
End Sub
```

dvgLookups = MS DataGridView

cCustomerLog.LoadCustomerLogDivisions()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

This function can be invoked to retrieve list of valid divisions for display/search criteria.

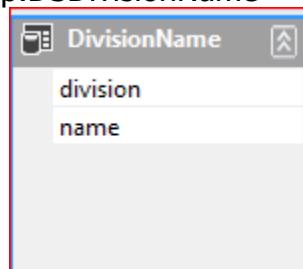
Public Function LoadCustomerLogDivisions(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As DCEnergyAPI.LookUp.DSDivisionName

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pErrorCode	Integer (32bit Integer)	0: no error <>0: error# * <i>Utility.GetErrorDescription()</i> to retrieve the description.*

Return Values

DataSet of type DCEnergyAPI.Lookup.DSDivisionName



division: what is saved to the database

name: what should be displayed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerLogDivisions>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCustomerLogDivisions>
  </soap:Body>
</soap:Envelope>
```

VB.NET Example:

```
Private Sub btnDivisions_Click(sender As Object, e As EventArgs) Handles btnDivisions.Click
    Using customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
        Dim errorCode As Integer = 0
        Dim iCompanyID as Short = 2
        dgvLookups.DataSource = customerLog.LoadCustomerLogDivisions(iCompanyID, "WebUser",
errorCode).Tables(0)
    End Using
End Sub

dvgLookups = MS DataGridView
```

cCustomerLog.LoadCustomerLogCollectors()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

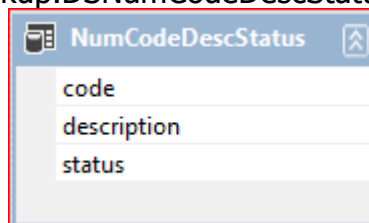
This function can be invoked to retrieve list of valid collectors for display/search criteria.
Public Function LoadCustomerLogCollectors(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As DCEnergyAPI.LookUp.DSNumCodeDescStatus

Input Parameters

pCompanyID	Short (16bit Integer)	Company ID
pSysUser	String	Calling program
pErrorCode	Integer (32bit Integer)	0: no error <>0: error * <i>Utility.GetErrorDescription()</i> to retrieve the description.*

Return Values

DataSet of type DCEnergyAPI.LookUp.DSNumCodeDescStatus



code: what is saved to the database
description: what should be displayed

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerLogCollectors>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCustomerLogCollectors>
  </soap:Body>
</soap:Envelope>
```

VB.NET Example:

```
Private Sub btnCollectors_Click(sender As Object, e As EventArgs) Handles btnCollectors.Click
    Using customerLog As EnergyAPI.CCustomerLog = New EnergyAPI.CCustomerLog()
        Dim errorCode As Integer = 0
        Dim iCompanyID as Short = 2
        dgvLookups.DataSource = customerLog.LoadCustomerLogCollectors(iCompanyID, "WebUser",
errorCode).Tables(0)
    End Using
End Sub

dvgLookups = MS DataGridView
```

CCustomerMaint class

Overview

This is the main interface class that must be used to create new customer records or modify existing records. It exposes all the needed methods to maintain all account record types (master, tank, service, wdms, meters, and ITS products) using an existing customer template. The programmer can use the following methods to create or modify a customer record:

- GetCustTemplates(): returns a list of existing templates
- GetCustMasterFields(): to return a list of customer master fields that can be displayed and modified by the user
- AddCustMaster(): to add a new customer master record
- UpdateCustMaster(): to modify an existing customer master record
- GetTankFields(): to return a list of tank fields that can be displayed and/or modified by the user
- AddTank(): to add a new customer delivery location (tank)
- UpdateTank(): to modify an existing delivery location
- GetServiceFields(): to return a list of service location fields that can be displayed and modified by the user
- AddService(): to add a new customer service location
- UpdateService(): to modify an existing service location
- GetMeterFields(): to return a list of meter record fields that can be displayed and modified by the user
- AddMeter(): to add a new meter record
- UpdateMeter(): to modify an existing meter record
- GetWdmsFields(): to return a list of wdms location fields that can be displayed and modified by the user
- AddWdms(): to add a new customer Wdms location
- UpdateWdms(): to modify an existing Wdms location
- GetITSFields(): to return a list of ITS product fields that can be displayed and modified by the user
- AddITSPRODUCT(): to add a new ITS product
- UpdateITSPRODUCT(): to modify an existing ITS product
- LoadDeliveryStopCodes() : used to lookup Delivery Stop Codes that are setup in the system
- LoadDeliveryStopInfo: used to load current Delivery Stop information
- SetDeliveryStop: used to set or update Delivery Stop information on a tank
- ClearDeliveryStop: used to clear Delivery Stop information on a tank

Note: For each of the cCustomerMaint.Update???() and cCustomerMaint.Add???() methods, you must include every customer field in the data set that is return by the cCustomerMaint.Get???Fields() method. Take everything starting from

<DSCustomerFields> until </DSCustomerFields> inclusive. The only thing to change is “new_value”.

cCustomerMaint.GetCustTemplates()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Returns a list of existing customer templates that can be used in ADD, Modify or both modes.

```
Public Function GetCustTemplates(ByVal p_iCompanyID As Short, _
    ByVal p_iRecordType As Short, _
    ByVal p_iMaintenanceMode As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_iRecordType	Short	0 = All, 1=Master, 2=Tanks, 3=Service, 4=Wdms, 5=Meters, 6=ITS
	p_iMaintenanceMode	Short	1 = ADD, 2=Modify, 3=Both
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

	p_lErrorCode (byRef)	Long integer	0 – if Successful <> 0 Error Number
Return Type : System.Data.DataSet			

- (0) Template ID ("template_id") - Long integer
- (1) Record Type ("group_id") – Short integer
- (2) Description ("description") – Char(50)
The description that will be presented to the end user
- (3) Tank template ID ("tank_template_id") – Long integer
- (4) Tank required("tank_required") – Bit
Must force the creation of a tank record if this field is true. Use the Tank template ID if not 0 otherwise allow user to select a template from the list of defined tank templates.
- (5) Service template ID ("service_template_id") – Long integer
- (6) Service required ("service_required") – Bit
Must force the creation of a service record if this field is true. Use the Service template ID if not 0 otherwise allow user to select a template from the list of defined service templates.
- (7) Wdms Template ID ("wdms_template_id") – Long integer
- (8) WDMS required ("wdms_required") – Bit
Must force the creation of a wdms record if this field is true. Use the wdms template ID if not 0 otherwise allow user to select a template from the list of defined wdms templates.
- (9) Meter template ID ("meter_template_id") – Long integer
- (10) Meter required ("meter_required") – Bit
Must force the creation of a meter record if this field is true. Use the Meter template ID if not 0 otherwise allow user to select a template from the list of defined meter templates.
- (11) ITS Product template ID ("its_product_template_id") – Long integer
- (12) ITS Product required ("its_product_required") – Bit
Must force the creation of a ITS record if this field is true. Use the ITS template ID if not 0 otherwise allow user to select a template from the list of defined ITS templates.
- (13) Maintenance mode ("maintenance_mode") – Short integer

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve a list of templates:

```

Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objTemplates As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objTemplates = objCustomerMaint.GetCustTemplates(iCompanyID, 1,2,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success

```

```
        For Each row In objTemplates.Tables(0).Rows
            Debug.Print "Template ID=" & Row (0) & "Type=" & _
                Row (1) & " Description=" & Row (2)
        Next
    End If
```

XML Example

The following example shows how to get a list of templates.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCustTemplates>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_iRecordType>0</add:p_iRecordType>
      <add:p_iMaintenanceMode>3</add:p_iMaintenanceMode>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetCustTemplates>
  </soap:Body>
</soap:Envelope>
```

cCustomerMaint.GetCustMasterFields()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Returns a list of customer master fields (as defined by the selected template) that can be displayed and modified by the user

```
Public Function GetCustMasterFields(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_lTemplateID As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	0= Add mode <>0 Modify Mode (Account number without the checkdigit
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number
Return Type : System.Data.DataSet		

(0)Field sequence number ("field_position") - Short integer

(1)Field name ("field_name") – Char(50)

The field name (column) as stored in the physical Sybase table.

(2)Long Description ("field_caption") – Char(100)

The field that should be presented to the end user.

(3)Data Type ("data_type") – Enumeration

2 – Short integer

3 – Long integer

4 – Single precision floating point

5 – Double precision floating point

6 – Currency

7 – Date

8 – String

11 – Boolean

99 – last maintenance date (internal use); handle as a string

98 – last maintenance user id (internal use); handle as a string

(4)Field Length("field_length") – Short integer

(5)LookUp Values("lookup_values") – Char(255)

A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).

(6)Stored Values ("stored_values") – Char(255)

A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).

(7)Display Field ("display_field") – Boolean

Do not display the field if the value is False.

(8)Primary key ("primary_key") – Boolean

(9)Read only ("read_only") – Boolean

The field cannot be modified.

(10)Default Value ("default_value") – Char(255)

When in add mode, use this value as the default value.

(11) Required field ("required") – Boolean

A value is mandatory if this field is True.

(12) Minimum value ("min_value") – Char(255)

Use this value to define the minimum value for the field.

(13) Maximum value ("max_value") – Char(255)

Use this value to define the maximum value for the field.

(14) Audit flag ("audit_flag") – Boolean Internal use.

(15) Log flag ("log_flag") – Boolean Internal use.

(16) Old Value ("old_value") – Char(255)

Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)

(17) New value ("new_value") – Char(255)

The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.

Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.

(18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields

(19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted, <> 0 → invalid data entered

(20) table_id – Short Integer – Internal Use.

- (21) other_flags – Short Integer - Internal Use.
- (22) special_field – Short Integer - Internal Use.

Special Consideration Fields

Field Name	Description
Telephone	The data will be received by the client application unformatted and therefore must be saved unformatted (9999999999 vs (999)999-9999). The received data will always include all possible digits.. Eg: if the phone has no area code then 0005551234, otherwise 9735551234
Postal_code	The data will be received by the client application unformatted and therefore must be saved unformatted (9999999999 vs 99999-9999). The received data will always include all possible digits. Eg: if zip+4 then 123453333 otherwise 123450000
Related_account	A valid account number that <> the current account number must be entered. The only available lookup for customers at this time will be by phone number.

Return code	Description
96	Unknown Error Occurred.
122	Invalid input parameters.

.NET Example

The following example shows how to retrieve a list of customer master fields to maintain:

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetCustMasterFields(iCompanyID, 0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & Row (0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of customer master fields to maintain:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
```

```
<soap:Header/>
<soap:Body>
  <add:GetCustMasterFields>
    <add:p_iCompanyID>2</add:p_iCompanyID>
    <add:p_lAccountNum>1</add:p_lAccountNum>
    <add:p_lTemplateID>-900</add:p_lTemplateID>
    <add:p_sSysUser>WebUser</add:p_sSysUser>
    <add:p_lErrorCode>0</add:p_lErrorCode>
  </add:GetCustMasterFields>
</soap:Body>
</soap:Envelope>
```

cCustomerMaint.AddCustMaster()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'AddCustMaster' method will be used to create a new customer (master record).

```
Public Function AddCustMaster(ByVal p_iCompanyID As Short, _
                             ByVal p_sSysUser As String, _
                             ByVal p_oRSMasterFields As System.Data.DataSet, _
                             ByRef p_lAccountNum As Integer, _
                             ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSMasterFields	System.Data.DataSet	The master record fields as selected by method GetCustMasterFields and updated by client.

Return Values

p_lAccountNum (ByRef)	Long integer	Account number without the checkdigit
p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	The master record fields as selected by method GetCustMasterFields and updated by client. If the master record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

Special Consideration Fields

Field Name	Description
Delivery	The data should be delimited with a '/' at about every 40
Instructions	characters. This will facilitate multi line DIN printing.

Return code	Description
96	Unknown Error Occurred.
122	Invalid input parameters.

128	The total length of the Name including the spaces and punctuation cannot exceed 40 characters
133	Unable to create the account. Please try again.
134	You have exceeded the maximum combined length of 87 characters for the fields Name, Street1, Street2, and/or Town, State. Try again.
135	Unable to update the general text record for the account. Please try again.
127	Nothing to save. Please try again

.NET Example

The following example shows how to create a new customer (master record).

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
' m_oRSMasterFields is dataset created by GetCustMasterFields () method.
m_oRSMasterFields.Tables(0).Rows(0).Item(17) = 100
```

```
...
'
```

```
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
```

```
'
m_oRSMasterFields = objCustomerMaint.AddCustMaster(iCompanyID, cSysUser,
m_oRSMasterFields ,lAcctNum,m_lRetCode)
```

```
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
End If
```

cCustomerMaint.UpdateCustMaster()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'UpdateCustMaster' method will be used to update an existing customer (master record).

NOTE: you must issue a Use cCustomerMaint.GetCustMasterFields to get the fields and values. Take everything starting from

```
<xs:schema id="DSCustomerFields"
Until
</diffgr:diffgram>
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSMasterFields	System.Data.DataSet	The master record fields as selected by method GetCustMasterFields and updated by client.

Return Values

p_lErrorCode (ByRef)	Long integer	0 – Successful, <> 0 – Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

The master record fields as selected by method GetCustMasterFields and updated by client. If the master record is Successfully updated the "New Value" will update the "Old Value". Basically, the passed dataset is updated (if applicable) and returned

Return	Description

code	
96	Unknown Error Occurred.
122	Invalid input parameters.
127	Nothing to save. Please try again
128	The total length of the Name including the spaces and punctuation cannot exceed 40 characters
132	The record has already been modified by another user. Please refresh the data and try again.
134	You have exceeded the maximum combined length of 87 characters for the fields Name, Street1, Street2, and/or Town, State. Try again.
136	Unable to update the delivery address record for the account. Please try again.

.NET Example

The following example shows how to modify an existing customer (master record).

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
' m_oRSMasterFields is dataset created by GetCustMasterFields () method.
m_oRSMasterFields.Tables(0).Rows(0).Item(17) = 100
...
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
m_oRSMasterFields = objCustomerMaint.UpdateCustMaster(iCompanyID,
1008,cSysUser,m_oRSMasterFields,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
End If
```

XML Example

The following example shows how to modify an existing customer (master record).

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:UpdateCustMaster>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_oRSMasterFields>
        <CustomerFields>
          <field_position>1</field_position>
```

```
<field_name>acquisition_date</field_name>
<field_caption>Acquisition date</field_caption>
<data_type>7</data_type>
<field_length>4</field_length>
<lookup_values></lookup_values>
<stored_values></stored_values>
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>12311960</min_value>
<max_value>06062079</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>9/13/1991 12:00:00 AM</old_value>
<new_value>9/13/1991 12:00:00 AM</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>3</field_position>
  <field_name>street1</field_name>
  <field_caption>Street 1</field_caption>
  <data_type>8</data_type>
  <field_length>40</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>3703 CAMINO HILLS DRIVE</old_value>
  <new_value>3703 CAMINO HILLS DRIVE</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
```

```
<CustomerFields>
  <field_position>4</field_position>
  <field_name>street2</field_name>
  <field_caption>Street 2</field_caption>
  <data_type>8</data_type>
  <field_length>40</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>2220 SEVEN OAKS CT.</old_value>
  <new_value>2220 SEVEN OAKS CT.</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>6</field_position>
  <field_name>postal_code</field_name>
  <field_caption>Zip code</field_caption>
  <data_type>8</data_type>
  <field_length>9</field_length>
  <lookup_values></lookup_values>
  <stored_values>999999999,99999</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>123450000</old_value>
  <new_value>123450000</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
```

```
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>7</field_position>
  <field_name>telephone</field_name>
  <field_caption>Telephone #</field_caption>
  <data_type>8</data_type>
  <field_length>10</field_length>
  <lookup_values></lookup_values>
  <stored_values>9999999999, 99999999,9999999</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>5551234567</old_value>
  <new_value>5551234567</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>8</field_position>
  <field_name>sort_code</field_name>
  <field_caption>Sort code</field_caption>
  <data_type>8</data_type>
  <field_length>6</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>*HAMIL</old_value>
  <new_value>*HAMIL</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
```

```
<table_id>900</table_id>
<other_flags>2</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>9</field_position>
  <field_name>general_text</field_name>
  <field_caption>General text</field_caption>
  <data_type>8</data_type>
  <field_length>250</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>INACTIVE ACCT./SOLD PROPERTY</old_value>
  <new_value>INACTIVE ACCT./SOLD PROPERTY</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>10</field_position>
  <field_name>division</field_name>
  <field_caption>Division</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values>EXEC load_divisions_$SP 3, VB_USER_ID, 1</lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>9998</old_value>
  <new_value>9998</new_value>
```

```

        <decimal_places>0</decimal_places>
        <error_code>0</error_code>
        <table_id>900</table_id>
        <other_flags>0</other_flags>
        <special_field>0</special_field>
    </CustomerFields>
    <CustomerFields>
        <field_position>11</field_position>
        <field_name>type</field_name>
        <field_caption>Customer type</field_caption>
        <data_type>2</data_type>
        <field_length>2</field_length>
        <lookup_values>,<Type #1,Type #2,Type #3,Type #4,Type #5,Type #6,Type
#7</lookup_values>
        <stored_values>0,1,2,3,4,5,6,7</stored_values>
        <display_field>true</display_field>
        <primary_key>>false</primary_key>
        <read_only>>false</read_only>
        <default_value></default_value>
        <required>true</required>
        <min_value>1</min_value>
        <max_value>7</max_value>
        <audit_flag>>false</audit_flag>
        <log_flag>true</log_flag>
        <old_value>2</old_value>
        <new_value>2</new_value>
        <decimal_places>0</decimal_places>
        <error_code>0</error_code>
        <table_id>900</table_id>
        <other_flags>0</other_flags>
        <special_field>0</special_field>
    </CustomerFields>
    <CustomerFields>
        <field_position>12</field_position>
        <field_name>category</field_name>
        <field_caption>category code</field_caption>
        <data_type>2</data_type>
        <field_length>2</field_length>
        <lookup_values>SELECT category, description, status FROM CATEGORY_INFO
ORDER BY category</lookup_values>
        <stored_values></stored_values>
        <display_field>true</display_field>
        <primary_key>>false</primary_key>
        <read_only>>false</read_only>
        <default_value></default_value>
        <required>>false</required>
        <min_value></min_value>
        <max_value></max_value>

```



```
<audit_flag>>false</audit_flag>
<log_flag>>true</log_flag>
<old_value>1</old_value>
<new_value>1</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>13</field_position>
  <field_name>priority_account</field_name>
  <field_caption>Priority account</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>14</field_position>
  <field_name>sales_person</field_name>
  <field_caption>Salesman</field_caption>
  <data_type>8</data_type>
  <field_length>3</field_length>
  <lookup_values>EXEC load_salesperson_lkp_$SP</lookup_values>
  <stored_values>ADD > , ,A<Ampersand></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
```

```
<min_value></min_value>
<max_value></max_value>
<audit_flag>>false</audit_flag>
<log_flag>>true</log_flag>
<old_value>GC</old_value>
<new_value>GC</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>15</field_position>
  <field_name>keyoff_account</field_name>
  <field_caption>Open Item</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value>N</default_value>
  <required>>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>17</field_position>
  <field_name>prem_bill_code</field_name>
  <field_caption>Premium billing code</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
```

```
<default_value></default_value>
<required>>false</required>
<min_value>0</min_value>
<max_value>7</max_value>
<audit_flag>>false</audit_flag>
<log_flag>>true</log_flag>
<old_value>0</old_value>
<new_value>0</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>18</field_position>
  <field_name>prem_bill_base</field_name>
  <field_caption>Premium billing base</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
  <required>>false</required>
  <min_value>0</min_value>
  <max_value>255</max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>19</field_position>
  <field_name>credit_line</field_name>
  <field_caption>Credit line dollars</field_caption>
  <data_type>3</data_type>
  <field_length>4</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>>true</display_field>
```

```

    <primary_key>>false</primary_key>
    <read_only>>false</read_only>
    <default_value></default_value>
    <required>>false</required>
    <min_value>0</min_value>
    <max_value>8350000</max_value>
    <audit_flag>>false</audit_flag>
    <log_flag>>true</log_flag>
    <old_value>100</old_value>
    <new_value>100</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>128</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>20</field_position>
    <field_name>credit_limit_code</field_name>
    <field_caption>Credit dollar-limit code</field_caption>
    <data_type>2</data_type>
    <field_length>1</field_length>
    <lookup_values>SELECT code, '$' + convert(varchar, convert(money,
float_value)), status FROM AR_PARAMETERS WHERE pneumonic = 'CRL' AND float_value
    <stored_values></stored_values>
    <display_field>>true</display_field>
    <primary_key>>false</primary_key>
    <read_only>>false</read_only>
    <default_value></default_value>
    <required>>false</required>
    <min_value></min_value>
    <max_value></max_value>
    <audit_flag>>false</audit_flag>
    <log_flag>>true</log_flag>
    <old_value>3</old_value>
    <new_value>3</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>21</field_position>
    <field_name>credit_time_code</field_name>
    <field_caption>Credit time-limit code</field_caption>
    <data_type>2</data_type>

```

```

        <field_length>1</field_length>
        <lookup_values>SELECT code, convert(varchar, int_value) + ' Days', status FROM
AR_PARAMETERS WHERE pneumatic = 'CRT' AND int_value <= 0 UNION SELECT
0,'No limit','A' order by code</lookup_values>
        <stored_values></stored_values>
        <display_field>true</display_field>
        <primary_key>false</primary_key>
        <read_only>false</read_only>
        <default_value></default_value>
        <required>true</required>
        <min_value>0</min_value>
        <max_value>9</max_value>
        <audit_flag>false</audit_flag>
        <log_flag>true</log_flag>
        <old_value>3</old_value>
        <new_value>3</new_value>
        <decimal_places>0</decimal_places>
        <error_code>0</error_code>
        <table_id>900</table_id>
        <other_flags>0</other_flags>
        <special_field>0</special_field>
    </CustomerFields>
    <CustomerFields>
        <field_position>22</field_position>
        <field_name>finance_charge</field_name>
        <field_caption>Finance charge</field_caption>
        <data_type>8</data_type>
        <field_length>1</field_length>
        <lookup_values>Yes,No</lookup_values>
        <stored_values>Y,N</stored_values>
        <display_field>true</display_field>
        <primary_key>false</primary_key>
        <read_only>false</read_only>
        <default_value>Y</default_value>
        <required>true</required>
        <min_value></min_value>
        <max_value></max_value>
        <audit_flag>false</audit_flag>
        <log_flag>true</log_flag>
        <old_value>Y</old_value>
        <new_value>Y</new_value>
        <decimal_places>0</decimal_places>
        <error_code>0</error_code>
        <table_id>900</table_id>
        <other_flags>4096</other_flags>
        <special_field>0</special_field>
    </CustomerFields>
    <CustomerFields>

```

```
<field_position>23</field_position>
<field_name>finance_charge_group</field_name>
<field_caption>Finance charge group</field_caption>
<data_type>2</data_type>
<field_length>1</field_length>
<lookup_values></lookup_values>
<stored_values></stored_values>
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>0</min_value>
<max_value>52</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>1</old_value>
<new_value>1</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>24</field_position>
  <field_name>budget_start</field_name>
  <field_caption>Budget start month</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>12</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
```

```
</CustomerFields>
<CustomerFields>
  <field_position>25</field_position>
  <field_name>num_budget_payments</field_name>
  <field_caption>Number of budget payments</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>12</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>26</field_position>
  <field_name>service_in_budget</field_name>
  <field_caption>Service contract in budget</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
```

```
<other_flags>4096</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>27</field_position>
  <field_name>budget_payment</field_name>
  <field_caption>Budget payment amount</field_caption>
  <data_type>6</data_type>
  <field_length>10</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0.00</min_value>
  <max_value>214700.00</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>28</field_position>
  <field_name>instant_invoice</field_name>
  <field_caption>Instant invoice</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value>N</default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>Y</old_value>
  <new_value>Y</new_value>
  <decimal_places>0</decimal_places>
```



```
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>4096</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>29</field_position>
  <field_name>ivo_account</field_name>
  <field_caption>Invoice flag</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>31</field_position>
  <field_name>instant_stat</field_name>
  <field_caption>Instant statement</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value>N</default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
```

```
<new_value>N</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>4096</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>32</field_position>
  <field_name>stat_day</field_name>
  <field_caption>Other statement days</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value>0</min_value>
  <max_value>160</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>32</old_value>
  <new_value>32</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>36</field_position>
  <field_name>tax_jurisdiction</field_name>
  <field_caption>Account tax county</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values>EXEC load_tax_jurisdiction_lkp_$SP 0</lookup_values>
  <stored_values>ADD>0,,A<&lt;</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>255</max_value>
  <audit_flag>false</audit_flag>
```

```
<log_flag>true</log_flag>
<old_value>1</old_value>
<new_value>1</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>37</field_position>
  <field_name>tax_1</field_name>
  <field_caption>Account tax 1</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>Y</old_value>
  <new_value>Y</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>38</field_position>
  <field_name>tax_2</field_name>
  <field_caption>Account tax 2</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
```

```
<max_value></max_value>
<audit_flag>>false</audit_flag>
<log_flag>>true</log_flag>
<old_value>Y</old_value>
<new_value>Y</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>4096</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>39</field_position>
  <field_name>tax_3</field_name>
  <field_caption>Account tax 3</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
  <required>>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>57</field_position>
  <field_name>canadian_postal_code</field_name>
  <field_caption>Canadian Postal Code</field_caption>
  <data_type>8</data_type>
  <field_length>6</field_length>
  <lookup_values></lookup_values>
  <stored_values>A9A9A9</stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
```

```
<required>false</required>
<min_value></min_value>
<max_value></max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value></old_value>
<new_value></new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>117</field_position>
  <field_name>prem_billing_terms</field_name>
  <field_caption>Premium billing terms</field_caption>
  <data_type>4</data_type>
  <field_length>4</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>99.99</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>118</field_position>
  <field_name>finance_charged_days</field_name>
  <field_caption>Finance charge days</field_caption>
  <data_type>2</data_type>
  <field_length>1</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
```

```
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>0</min_value>
<max_value>155</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>0</old_value>
<new_value>0</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>128</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>119</field_position>
  <field_name>cum_finance_charge</field_name>
  <field_caption>Cumulative finance charge</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0.00</min_value>
  <max_value>83500.00</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>128</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>120</field_position>
  <field_name>budget_paid</field_name>
  <field_caption>Budget Total year-to-date</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
```

```
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value></min_value>
<max_value></max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>0</old_value>
<new_value>0</new_value>
<decimal_places>2</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>128</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>133</field_position>
  <field_name>no_dunning</field_name>
  <field_caption>Exempt from dunning</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>4096</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>144</field_position>
  <field_name>next_year_volume</field_name>
  <field_caption>Next Years Volume</field_caption>
  <data_type>4</data_type>
  <field_length>8</field_length>
```

```

    <lookup_values></lookup_values>
    <stored_values></stored_values>
    <display_field>true</display_field>
    <primary_key>>false</primary_key>
    <read_only>>false</read_only>
    <default_value></default_value>
    <required>>false</required>
    <min_value></min_value>
    <max_value></max_value>
    <audit_flag>>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>0</old_value>
    <new_value>0</new_value>
    <decimal_places>2</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>128</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>146</field_position>
    <field_name>collector_num</field_name>
    <field_caption>Collector</field_caption>
    <data_type>2</data_type>
    <field_length>2</field_length>
    <lookup_values>SELECT collector_num, collector_name, status FROM
COLLECTOR WHERE collector_num > 0 ORDER BY(collector_num)</lookup_values>
    <stored_values>ADD >0, ,A<&lt;</stored_values>
    <display_field>true</display_field>
    <primary_key>>false</primary_key>
    <read_only>>false</read_only>
    <default_value></default_value>
    <required>>false</required>
    <min_value></min_value>
    <max_value></max_value>
    <audit_flag>>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>1</old_value>
    <new_value>1</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>149</field_position>
    <field_name>overage_dollar_amount</field_name>

```



```
<field_caption>OverAge Limit Dollars</field_caption>
<data_type>3</data_type>
<field_length>4</field_length>
<lookup_values></lookup_values>
<stored_values></stored_values>
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>0</min_value>
<max_value>8350000</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>0</old_value>
<new_value>0</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>128</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>152</field_position>
  <field_name>statement_message_code</field_name>
  <field_caption>Statement Message Code</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>9999</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
```

```
<field_position>153</field_position>
<field_name>company_form_code</field_name>
<field_caption>Company Code for Forms</field_caption>
<data_type>2</data_type>
<field_length>0</field_length>
<lookup_values></lookup_values>
<stored_values></stored_values>
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>0</min_value>
<max_value>255</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>10</old_value>
<new_value>10</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>157</field_position>
  <field_name>ytd_units</field_name>
  <field_caption>Year to date gallons</field_caption>
  <data_type>4</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>29282</old_value>
  <new_value>29282</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>128</other_flags>
  <special_field>0</special_field>
```

```
</CustomerFields>
<CustomerFields>
  <field_position>160</field_position>
  <field_name>balance</field_name>
  <field_caption>Account Balance</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>true</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>-99999999.99</min_value>
  <max_value>99999999.99</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>38006.28</old_value>
  <new_value>38006.28</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>129</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>161</field_position>
  <field_name>acquisition_balance</field_name>
  <field_caption>Current Acquisition Balance</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
```

```

    <other_flags>128</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>171</field_position>
    <field_name>acquisition_reason_code</field_name>
    <field_caption>Acquisition reason code</field_caption>
    <data_type>2</data_type>
    <field_length>0</field_length>
    <lookup_values>SELECT code = 0, description = "", status = "A" UNION
SELECT code,description,status FROM CODE_DESCRIPTION_DETAIL WHERE table_id=6
ORDER BY code</lookup_values>
    <stored_values></stored_values>
    <display_field>true</display_field>
    <primary_key>false</primary_key>
    <read_only>false</read_only>
    <default_value></default_value>
    <required>false</required>
    <min_value></min_value>
    <max_value></max_value>
    <audit_flag>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>0</old_value>
    <new_value>0</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>172</field_position>
    <field_name>acquisition_company</field_name>
    <field_caption>Acquisition Company</field_caption>
    <data_type>2</data_type>
    <field_length>0</field_length>
    <lookup_values></lookup_values>
    <stored_values></stored_values>
    <display_field>true</display_field>
    <primary_key>false</primary_key>
    <read_only>false</read_only>
    <default_value></default_value>
    <required>false</required>
    <min_value></min_value>
    <max_value></max_value>
    <audit_flag>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>0</old_value>

```

```
<new_value>0</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>173</field_position>
  <field_name>group_code_1</field_name>
  <field_caption>Group Code 1</field_caption>
  <data_type>3</data_type>
  <field_length>0</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>0</min_value>
  <max_value>999999</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>175</field_position>
  <field_name>budget_dollars_billed</field_name>
  <field_caption>Budget Dollars Billed</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>-99999.99</min_value>
  <max_value>4999999.99</max_value>
  <audit_flag>false</audit_flag>
```

```
<log_flag>true</log_flag>
<old_value>0</old_value>
<new_value>0</new_value>
<decimal_places>2</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>128</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>176</field_position>
  <field_name>budget_adjustment_dollars</field_name>
  <field_caption>Budget Adjustment Dollars</field_caption>
  <data_type>6</data_type>
  <field_length>8</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value>-99999.99</min_value>
  <max_value>99999.99</max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>2</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>128</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>177</field_position>
  <field_name>title</field_name>
  <field_caption>Title</field_caption>
  <data_type>8</data_type>
  <field_length>4</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
```

```
<max_value></max_value>
<audit_flag>>false</audit_flag>
<log_flag>>true</log_flag>
<old_value></old_value>
<new_value></new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>1</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>178</field_position>
  <field_name>first_name</field_name>
  <field_caption>First Name</field_caption>
  <data_type>8</data_type>
  <field_length>40</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
  <required>>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value></old_value>
  <new_value></new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>180</field_position>
  <field_name>last_name</field_name>
  <field_caption>Last name</field_caption>
  <data_type>8</data_type>
  <field_length>40</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
```

```
<required>true</required>
<min_value></min_value>
<max_value></max_value>
<audit_flag>>false</audit_flag>
<log_flag>true</log_flag>
<old_value>PHIL & DEBBIE HAMILTON</old_value>
<new_value>PHIL & DEBBIE HAMILTON</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>1</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>181</field_position>
  <field_name>name_suffix</field_name>
  <field_caption>Name Suffix</field_caption>
  <data_type>8</data_type>
  <field_length>3</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
  <required>>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value></old_value>
  <new_value></new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>183</field_position>
  <field_name>budget_start_year</field_name>
  <field_caption>Budget Start Year</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
```



```
<read_only>false</read_only>
<default_value></default_value>
<required>false</required>
<min_value>0</min_value>
<max_value>2079</max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value>2007</old_value>
<new_value>2007</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>188</field_position>
  <field_name>city</field_name>
  <field_caption>City</field_caption>
  <data_type>8</data_type>
  <field_length>37</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>true</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>EL DORADO HILLS, CA.</old_value>
  <new_value>EL DORADO HILLS, CA.</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>189</field_position>
  <field_name>state</field_name>
  <field_caption>State</field_caption>
  <data_type>8</data_type>
  <field_length>2</field_length>
  <lookup_values></lookup_values>
  <stored_values></stored_values>
```

```
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value></default_value>
<required>true</required>
<min_value></min_value>
<max_value></max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value></old_value>
<new_value></new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>191</field_position>
  <field_name>referral_source</field_name>
  <field_caption>Customer Source Referral</field_caption>
  <data_type>2</data_type>
  <field_length>4</field_length>
  <lookup_values>EXEC load_accounts_source_lkp_$SP 0,0</lookup_values>
  <stored_values>ADD >0, ,A<&lt;</stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value></default_value>
  <required>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>194</field_position>
  <field_name>term_id</field_name>
  <field_caption>Term code</field_caption>
  <data_type>2</data_type>
  <field_length>5</field_length>
```

```
<lookup_values>SELECT term_id, description, status FROM TERM_HEADER
WHERE term_id > 0 ORDER BY term_id</lookup_values>
<stored_values>ADD >0, ,A<&lt;</stored_values>
<display_field>true</display_field>
<primary_key>>false</primary_key>
<read_only>>false</read_only>
<default_value></default_value>
<required>>false</required>
<min_value></min_value>
<max_value></max_value>
<audit_flag>>false</audit_flag>
<log_flag>true</log_flag>
<old_value>1</old_value>
<new_value>1</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>201</field_position>
  <field_name>language</field_name>
  <field_caption>Language</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>,<English,French</lookup_values>
  <stored_values>,<E,F</stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value></default_value>
  <required>>false</required>
  <min_value></min_value>
  <max_value></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>F</old_value>
  <new_value>F</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>62</field_position>
  <field_name>last_maintenance_dt</field_name>
```

```
<field_caption>last maintenance date</field_caption>
<data_type>99</data_type>
<field_length>30</field_length>
<display_field>>false</display_field>
<read_only>true</read_only>
<required>>false</required>
<audit_flag>>false</audit_flag>
<log_flag>>false</log_flag>
<old_value>Apr 19 2012 3:05:48:103PM</old_value>
<new_value>Apr 19 2012 3:05:48:103PM</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>0</table_id>
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields>
  <field_position>63</field_position>
  <field_name>data_conversion</field_name>
  <field_caption>Data Conversion</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>Yes,No</lookup_values>
  <stored_values>Y,N</stored_values>
  <display_field>true</display_field>
  <read_only>>false</read_only>
  <default_value>N</default_value>
  <required>>false</required>
  <audit_flag>>false</audit_flag>
  <log_flag>>false</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>0</table_id>
  <other_flags>0</other_flags>
  <special_field>1</special_field>
</CustomerFields>>
</add:p_oRSMasterFields>
<add:p_lErrorCode>0</add:p_lErrorCode>
</add:UpdateCustMaster>
</soap:Body>
</soap:Envelope>
```

cCustomerMaint.GetTankFields()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Returns a list of tank fields (as defined by the selected template) that can be displayed and modified by the user.

```
Public Function GetTankFields(ByVal p_iCompanyID As Short, _  
    ByVal p_lAccountNum As Integer, _  
    ByVal p_lTankNum As Integer, _  
    ByVal p_lTemplateID As Integer, _  
    ByVal p_sSysUser As String, _  
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_lTankNum	Long integer	0 – Add mode <> 0 Modify Mode
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

- (0)Field sequence number ("field_position") - Short integer
- (1)Field name ("field_name") – Char(50)
The field name (column) as stored in the physical Sybase table.
- (2)Long Description ("field_caption") – Char(100)
The field that should be presented to the end user.
- (3)Data Type ("data_type") – Enumeration
2 – Short integer
3 – Long integer
4 – Single precision floating point
5 – Double precision floating point
6 – Currency
7 – Date
8 – String
11 – Boolean
99 – last maintenance date (internal use); handle as a string.
98 – last maintenance user id (internal use); handle as a string.
- (4)Field Length("field_length") – Short integer
- (5)LookUp Values("lookup_values") – Char(255)
A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
- (6)Stored Values ("stored_values") – Char(255)
A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
- (7)Display Field ("display_field") – Boolean
Do not display the field if the value is False.
- (8)Primary key ("primary_key") – Boolean
- (9)Read only ("read_only") – Boolean
The field cannot be modified.
- (10)Default Value ("default_value") – Char(255)
When in add mode, use this value as the default value.
- (11) Required field ("required") – Boolean
A value is mandatory if this field is True.
- (12) Minimum value ("min_value") – Char(255)
Use this value to define the minimum value for the field.
- (13) Maximum value ("max_value") – Char(255)
Use this value to define the maximum value for the field.
- (14) Audit flag ("audit_flag") – Boolean
Internal use.
- (15) Log flag ("log_flag") – Boolean Internal use.
- (16) Old Value ("old_value") – Char(255)
Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)
- (17) New value ("new_value") – Char(255)
The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.
Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.
- (18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields
- (19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted,
<> 0 → invalid data entered

- (20) table_id – Short Integer – Internal Use.
 (21) other_flags – Short Integer - Internal Use.
 (22) special_field – Short Integer - Internal Use.

Special Consideration Fields

Field Name	Description
Delivery_address	The delivery_address field has been rendered “read-only”. This field should not be programmatically manipulated and is meant only for display purposes. When adding/modifying delivery records the following fields should be used/manipulated and presented to the end user: dad_title, dad_first_name, dad_middle_initial, dad_last_name,dad_name_suffix,dad_street1,dad_street2,dad_city,dad_state and dad_postal_code Note: In the case where an existing delivery record that never had the address parsed is being loaded, the system will do its best to parse the unformatted text address based on the following assumption/rule; the address comprises of 4 lines of upto 40 characters/line where: Line 1 = full name – this value will be assigned to dad_last_name Line 2 = street 1 – this value will be assigned to dad_street1 Line 3 = street 2 – this value will be assigned to dad_street2 Line 4 = city, state zip code – these values will be assigned to dad_city, dad_state and dad_postal_code respectively

Return code	Description
96	Unknown Error Occurred.
122	Invalid input parameters.

.NET Example

The following example shows how to retrieve a list of tank fields to maintain:

```

Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetTankFields(iCompanyID, 1008,0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & Row (0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
End If

```

XML Example

The following example shows how to retrieve a list of tank fields to maintain:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetTankFields>
      <add:p_iCompanyID>2</add:p_iCompanyID>
      <add:p_lAccountNum>1</add:p_lAccountNum>
      <add:p_lTankNum>1</add:p_lTankNum>
      <add:p_lTemplateID>-901</add:p_lTemplateID>
      <add:p_sSysUser>WebUser</add:p_sSysUser>
      <add:p_lErrorCode>0</add:p_lErrorCode>
    </add:GetTankFields>
  </soap:Body>
</soap:Envelope>
```


cCustomerMaint.AddTank()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'AddTank()' method will be used to create a new tank record for an existing customer.

```
Public Function AddTank(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSTankFields As System.Data.DataSet, _
    ByRef p_lTankNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSTankFields	System.Data.DataSet	The tank record fields as selected by method GetTankFields() and updated by client.

Return Values

p_lTankNum (ByRef)	Long Integer	A zero is sent; a valid tank number is returned.
p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

The tank record fields as selected by method GetTankFields() and updated by client. If the tank record is Successfully created the "New Value" will update the "Old Value". Basically, the passed dataset is updated (if applicable) and returned

Return code	Description
96	Unknown Error Occurred.
122	Invalid input parameters.

127	Nothing to save. Please try again
136	Unable to update the delivery address record for the account. Please try again.
137	Unable to update the delivery instruction record for the account. Please try again.

.NET Example

The following example shows how to create a new tank record for an existing customer.

```
Dim objCustomerMaint as Object  
Dim lRetCode as Integer
```

```
‘ m_oRSTankFields is the dataset created by GetTankFields () method.  
m_oRSTankFields.Tables(0).Rows(0).Item(17) = 100
```

```
...  
;  
;
```

```
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")  
;
```

```
m_oRSTankFields = objCustomerMaint.AddTank(iCompanyID, 1008,cSysUser,  
m_oRSTankFields,lTankNum,lRetCode)  
;
```

```
If lRetCode <> 0 Then  
    ‘ Failure - Display error message  
Else  
End If
```

cCustomerMaint.UpdateTank()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The ‘*UpdateTank()*’ method will be used to update an existing tank record for an existing customer.

```
Public Function UpdateTank(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSTankFields As System.Data.DataSet, _
    ByVal p_lTankNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSTankFields	System.Data.DataSet	The tank record fields as selected by method GetTankFields() and updated by client.
	p_lTankNum	Long Integer	Selected tank number

Return Values

p_lErrorCode (ByRef)	Long Integer	0 – if Successful <> 0 Error Number
Return Type: System.Data.DataSet	System.Data.DataSet	The tank record fields as selected by method GetTankFields() and updated by client. If the tank record is Successfully updated the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

Return code	Description
96	Unknown Error Occurred.
122	Invalid input parameters.
127	Nothing to save. Please try again
132	The record has already been modified by another user. Please refresh the data and try again.
134	You have exceeded the maximum combined length of 87 characters for the fields Name, Street1, Street2, and/or Town, State. Try again.
136	Unable to update the delivery address record for the account. Please try again.

137	Unable to update the delivery instruction record for the account. Please try again.

.NET Example

The following example shows how to modify an existing tank record for an existing customer.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
'
' m_oRSTankFields is dataset created by GetTankFields () method.
m_oRSTankFields.Tables(0).Rows(0).Item(17) = 100

...
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
m_oRSTankFields = objCustomerMaint.UpdateTank(iCompanyID, 1008,cSysUser,
m_oRSTankFields ,1,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
End If
```

cCustomerMaint.GetMeterFields()**NOTE: For future use.**

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Returns a list of meter fields (as defined by the selected template) that can be displayed and modified by the user.

```
Public Function GetMeterFields(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum as Integer, _
    ByVal p_lTankNum As Integer, _
    ByVal p_lTemplateID As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_lTankNum	Long integer	0 – Add mode <> 0 Modify Mode; a valid tank number
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type : System.Data.DataSet

- (0)Field sequence number ("field_position") - Short integer
 - (1)Field name ("field_name") – Char(50)
The field name (column) as stored in the physical Sybase table.
 - (2)Long Description ("field_caption") – Char(100)
The field that should be presented to the end user.
 - (3)Data Type ("data_type") – Enumeration
 - 2 – Short integer
 - 3 – Long integer
 - 4 – Single precision floating point
 - 5 – Double precision floating point
 - 6 – Currency
 - 7 – Date
 - 8 – String
 - 11 – Boolean
 - 99 – last maintenance date (internal use)
 - 98 – last maintenance user id (internal use)
 - (4)Field Length("field_length") – Short integer
 - (5)Lookup Values("lookup_values") – Char(255)
A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (6)Stored Values ("stored_values") – Char(255)
A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (7)Display Field ("display_field") – Boolean
Do not display the field if the value is False.
 - (8)Primary key ("primary_key") – Boolean
 - (9)Read only ("read_only") – Boolean
The field cannot be modified.
 - (10)Default Value ("default_value") – Char(255)
When in add mode, use this value as the default value.
 - (11) Required field ("required") – Boolean
A value is mandatory if this field is True.
 - (12) Minimum value ("min_value") – Char(255)
Use this value to define the minimum value for the field.
 - (13) Maximum value ("max_value") – Char(255)
Use this value to define the maximum value for the field.
 - (14) Audit flag ("audit_flag") – Boolean Internal use.
 - (15) Log flag ("log_flag") – Boolean Internal use.
 - (16) Old Value ("old_value") – Char(255)
Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)
 - (17) New value ("new_value") – Char(255)
The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.
- Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.
- (18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields
 - (19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted, <> 0 → invalid data entered

- (20) table_id – Short Integer – Internal Use.
- (21) other_flags – Short Integer - Internal Use.
- (22) special_field – Short Integer - Internal Use.

.NET Example

The following example shows how to retrieve a list of meter fields to maintain:

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetMeterFields(iCompanyID, 1008,0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & Row (0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
End If
```

cCustomerMaint.AddMeter()**NOTE: For future use.**

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'AddMeter()' method will be used to create a new meter record for an existing customer tank record.

```
Public Function AddMeter(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSMeterFields As System.Data.DataSet, _
    ByVal p_lTankNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSMeterFields	System.Data.DataSet	The meter record fields as selected by method GetMeterFields() and updated by the client.

	p_lTankNum	Long Integer	Selected tank number

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

The meter record fields as selected by method GetMeterFields() and updated by the client. If the meter record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to create a new meter record for an existing customer tank record.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

‘ m_oRSMeterFields is dataset created by GetMeterFields () method.


```
m_oRSMeterFields.Tables(0).Rows(0).Item(17) = 100
...
'
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
m_oRSMeterFields = objCustomerMaint.AddMeter(iCompanyID, 1008,cSysUser,
m_oRSMeterFields,lTankNum,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
End If
```

cCustomerMaint.UpdateMeter()**NOTE: For future use.**

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*UpdateMeter()*' method will be used to update an existing meter record for an existing customer tank record.

```
Public Function UpdateMeter(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSMeterFields As System.Data.DataSet, _
    ByVal p_lTankNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSMeterFields	System.Data.DataSet	The meter record fields as selected by method GetMeterFields() and updated by the client.

	p_lTankNum	Long Integer	Selected tank number

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	The meter record fields as selected by method GetMeterFields() and updated by the client. If the meter record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to modify an existing meter record for an existing customer tank record.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
‘
‘ m_oRSMeterFields is dataset created by GetMeterFields () method.
```

```
m_oRSMeterFields.Tables(0).Rows(0).Item(17) = 100
...
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
m_oRSMeterFields = objCustomerMaint.UpdateMeter(iCompanyID, 1008,cSysUser,
m_oRSMeterFields,1,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
End If
```

cCustomerMaint.GetServiceFields()

- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Customer Add & Modify Bus Edition

Description

Returns a list of service fields (as defined by the selected template) that can be displayed and modified by the user.

```
Public Function GetServiceFields(ByVal p_iCompanyID As Short, _  
    ByVal p_lAccountNum As Integer, _  
    ByVal p_lServiceNum As Integer, _  
    ByVal p_lTemplateID As Integer, _  
    ByVal p_sSysUser As String, _  
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_lServiceNum	Long integer	0 – Add mode <> 0 Modify Mode
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number
Return Type: System.Data.DataSet		

- (0)Field sequence number ("field_position") - Short integer
 - (1)Field name ("field_name") – Char(50)
The field name (column) as stored in the physical Sybase table.
 - (2)Long Description ("field_caption") – Char(100)
The field that should be presented to the end user.
 - (3)Data Type ("data_type") – Enumeration
 - 2 – Short integer
 - 3 – Long integer
 - 4 – Single precision floating point
 - 5 – Double precision floating point
 - 6 – Currency
 - 7 – Date
 - 8 – String
 - 11 – Boolean
 - 99 – last maintenance date (internal use); handle as a string.
 - 98 – last maintenance user id (internal use); handle as a string.
 - (4)Field Length("field_length") – Short integer
 - (5)LookUp Values("lookup_values") – Char(255)
A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (6)Stored Values ("stored_values") – Char(255)
A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (7)Display Field ("display_field") – Boolean
Do not display the field if the value is False.
 - (8)Primary key ("primary_key") – Boolean
 - (9)Read only ("read_only") – Boolean
The field cannot be modified.
 - (10)Default Value ("default_value") – Char(255)
When in add mode, use this value as the default value.
 - (11) Required field ("required") – Boolean
A value is mandatory if this field is True.
 - (12) Minimum value ("min_value") – Char(255)
Use this value to define the minimum value for the field.
 - (13) Maximum value ("max_value") – Char(255)
Use this value to define the maximum value for the field.
 - (14) Audit flag ("audit_flag") – Boolean Internal use.
 - (15) Log flag ("log_flag") – Boolean Internal use.
 - (16) Old Value ("old_value") – Char(255)
Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)
 - (17) New value ("new_value") – Char(255)
The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.
- Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.
- (18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields
 - (19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted, <> 0 → invalid data entered

- (20) table_id – Short Integer – Internal Use.
- (21) other_flags – Short Integer - Internal Use.
- (22) special_field – Short Integer - Internal Use.

Special Consideration Fields

Field Name	Description
Service_address	The service_address field has been rendered “read-only”. This field should not be programmatically manipulated and is meant only for display purposes. When adding/modifying service records the following fields should be used/manipulated and presented to the end user: sad_title, sad_first_name, sad_middle_initial, sad_last_name, sad_name_suffix, sad_street1, sad_street2, sad_city, sad_state and sad_postal_code

Note: In the case where an existing service record that never had the address parsed is being loaded, the system will do its best to parse the unformatted text address based on the following assumption/rule; the address comprises of 4 lines of upto 40 characters/line where:

Line 1 = full name – this value will be assigned to sad_last_name

Line 2 = street 1 – this value will be assigned to sad_street1

Line 3 = street 2 – this value will be assigned to sad_street2

Line 4 = city, state zip code – these values will be assigned to sad_city, sad_state and sad_postal_code respectively

.NET Example

The following example shows how to retrieve a list of service fields to maintain:

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetServiceFields(iCompanyID, 1008,0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & Row (0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
End If
```

XML Example

The following example shows how to retrieve a list of service fields to maintain:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
```

```
<soap:Header/>
<soap:Body>
  <add:GetServiceFields>
    <add:p_iCompanyID>2</add:p_iCompanyID>
    <add:p_lAccountNum>1</add:p_lAccountNum>
    <add:p_lServiceNum>1</add:p_lServiceNum>
    <add:p_lTemplateID>-902</add:p_lTemplateID>
    <add:p_sSysUser>WebUser</add:p_sSysUser>
    <add:p_lErrorCode>0</add:p_lErrorCode>
  </add:GetServiceFields>
</soap:Body>
</soap:Envelope>
```

cCustomerMaint.AddService()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The ‘AddService()’ method will be used to create a new service record on an existing customer.

```
Public Function AddService(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSServiceFields As System.Data.DataSet, _
    ByRef p_lServiceNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSServiceFields	System.Data.DataSet	the service record fields as selected by method GetServiceFields() and updated by the client.

Return Values

	p_lServiceNum (ByRef)	Long Integer	A zero is sent; a valid service number is returned.
	p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: System.Data.DataSet		System.Data.DataSet	the service record fields as selected by method GetServiceFields() and updated by the client. If the service record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to create a new service record for an existing customer.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
‘ m_oRSServiceFields is dataset created by GetServiceFields () method.
m_oRSServiceFields.Tables(0).Rows(0).Item(17) = 100
```

```
...
;
```



```
‘  
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")  
‘  
m_oRSServiceFields = objCustomerMaint.AddService(iCompanyID, 1008,cSysUser,  
m_oRSServiceFields,lServiceNum,lRetCode)  
‘  
If lRetCode <> 0 Then  
    ‘ Failure - Display error message  
Else  
End If
```

cCustomerMaint.UpdateService()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*UpdateService()*' method will be used to update an existing service record for an existing customer.

```
Public Function UpdateService(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_oRSServiceFields As System.Data.DataSet, _
    ByVal p_lServiceNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet '
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSServiceFields	System.Data.DataSet	The service record fields as selected by method GetServiceFields() and updated by the client.
	p_lServiceNum	Long Integer	Selected service number

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type:	System.Data.DataSet	System.Data.DataSet	The service record fields as selected by method GetServiceFields() and updated by the client. If the service record is Successfully updated the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to modify an existing service record for an existing customer.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
'
' m_oRSServiceFields is dataset created by GetServiceFields () method.
m_oRSServiceFields.Tables(0).Rows(0).Item(17) = 100
...
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
```

```
m_oRSServiceFields = objCustomerMaint.UpdateService(iCompanyID, 1008,cSysUser,  
m_oRSServiceFields,1,lRetCode)  
,
```

```
If lRetCode <> 0 Then  
    ' Failure - Display error message  
EndIf
```

cCustomerMaint.GetWdmsFields()

- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Customer Add & Modify Bus Edition

Description

Returns a list of wdms fields (as defined by the selected template) that can be displayed and modified by the user.

```
Public Function GetWdmsFields(ByVal p_iCompanyID As Short, _  
    ByVal p_lAccountNum as Integer, _  
    ByVal p_lWdmsNum As Integer, _  
    ByVal p_lTemplateID As Integer, _  
    ByVal p_sSysUser As String, _  
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_lWdmsNum	Long integer	0 – Add mode <> 0 Modify Mode
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

- (0)Field sequence number ("field_position") - Short integer
 - (1)Field name ("field_name") – Char(50)
The field name (column) as stored in the physical Sybase table.
 - (2)Long Description ("field_caption") – Char(100)
The field that should be presented to the end user.
 - (3)Data Type ("data_type") – Enumeration
 - 2 – Short integer
 - 3 – Long integer
 - 4 – Single precision floating point
 - 5 – Double precision floating point
 - 6 – Currency
 - 7 – Date
 - 8 – String
 - 11 – Boolean
 - 99 – last maintenance date (internal use); handle as a string.
 - 98 – last maintenance user id (internal use); handle as a string.
 - (4)Field Length("field_length") – Short integer
 - (5)LookUp Values("lookup_values") – Char(255)
A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (6)Stored Values ("stored_values") – Char(255)
A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (7)Display Field ("display_field") – Boolean
Do not display the field if the value is False.
 - (8)Primary key ("primary_key") – Boolean
 - (9)Read only ("read_only") – Boolean
The field cannot be modified.
 - (10)Default Value ("default_value") – Char(255)
When in add mode, use this value as the default value.
 - (11) Required field ("required") – Boolean
A value is mandatory if this field is True.
 - (12) Minimum value ("min_value") – Char(255)
Use this value to define the minimum value for the field.
 - (13) Maximum value ("max_value") – Char(255)
Use this value to define the maximum value for the field.
 - (14) Audit flag ("audit_flag") – Boolean Internal use.
 - (15) Log flag ("log_flag") – Boolean Internal use.
 - (16) Old Value ("old_value") – Char(255)
Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)
 - (17) New value ("new_value") – Char(255)
The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.
- Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.
- (18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields
 - (19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted, <> 0 → invalid data entered

- (20) table_id – Short Integer – Internal Use.
- (21) other_flags – Short Integer - Internal Use.
- (22) special_field – Short Integer - Internal Use.

Special Consideration Fields

Field Name	Description
Phone_number	The data will be received by the client application unformatted and therefore must be saved unformatted (9999999999 vs (999)999-9999). The received data will always include all possible digits.. Eg: if the phone has no area code then 0005551234, otherwise 9735551234
Zip	The data will be received by the client application unformatted and therefore must be saved unformatted (9999999999 vs 99999-9999). The received data will always include all possible digits.. Eg: if zip+4 then 123453333 otherwise 123450000
Location_text	A forward-slash (/) delimited string of text. Defined as 9 lines of maximum 70 chars for each line. When saving, the order must be kept and a forward-slash (/) must be present regardless if data is present or not. Eg: Line1//Line3//Line5//Line6//Line8//Line9

.NET Example

The following example shows how to retrieve a list of wdms fields to maintain:

```

Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetWDMSFields(iCompanyID, 1008,0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & objFields.Tables(0).Rows(0).Item(0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
End If

```

cCustomerMaint.AddWdms()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The ‘AddWdms()’ method will be used to create a new wdms record on an existing customer.

```
Public Function AddWdms(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSWdmsFields As System.Data.DataSet, _
    ByRef p_lLocationNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSWDMSFields	System.Data.DataSet	the wdms record fields as selected by method GetWdmsFields() and updated by the client.

Return Values

	p_lLocationNum (ByRef)	Long Integer	A zero is sent; a valid wdms number is returned.
	p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: System.Data.DataSet		System.Data.DataSet	the wdms record fields as selected by method GetWdmsFields() and updated by the client. If the wdms record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to create a new wdms record for an existing customer.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
‘ m_oRSWDMSFields is dataset created by GetWDMSFields () method.
m_oRSWDMSFields.Tables(0).Rows(0).Item(17) = 100
...
‘
‘
```

```
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
m_oRSWDMSFields = objCustomerMaint.AddWdms(iCompanyID, 1008,cSysUser,
m_oRSWDMSFields,lWdmsNum,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
End If
```


cCustomerMaint.UpdateWdms()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'UpdateWdms()' method will be used to update an existing wdms record for an existing customer.

```
Public Function UpdateWdms(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSWdmsFields As System.Data.DataSet, _
    ByVal p_lWdmsNum As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
    ,
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSWdmsFields	System.Data.DataSet	the wdms record fields as selected by method GetWdmsFields() and updated by the client.
	p_lWdmsNum	Long Integer	Selected wdms location number

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number
Return Type:	System.Data.DataSet	System.Data.DataSet	the wdms record fields as selected by method GetWdmsFields() and updated by the client. If the Wdms record is Successfully updated the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to modify an existing wdms record for an existing customer.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
'
' m_oRSWdmsFields is dataset created by GetWdmsFields () method.
m_oRSWdmsFields.Tables(0).Rows(0).Item(17) = 100
...
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
```

```
m_oRSWdmsFields = objCustomerMaint.UpdateWdms(iCompanyID, 1008,cSysUser,  
m_oRSWdmsFields ,1,lRetCode)  
,
```

```
If lRetCode <> 0 Then  
    ' Failure - Display error message  
EndIf
```

cCustomerMaint.GetItsFields()**NOTE: For future use.**

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

Returns a list of ITS fields (as defined by the selected template) that can be displayed and modified by the user

```
Public Function GetItsFields(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum as Integer, _
    ByVal p_lWdmsNum As Integer, _
    ByVal p_lItsProduct as Integer, _
    ByVal p_lTemplateID As Integer, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_lWdmsNum	Long integer	Wdms location number
	p_lItsProduct	Long Integer	0 – Add mode <> 0 Modify Mode
	p_lTemplateID	Long integer	
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type : System.Data.DataSet

- (0)Field sequence number ("field_position") - Short integer
 - (1)Field name ("field_name") – Char(50)
The field name (column) as stored in the physical Sybase table.
 - (2)Long Description ("field_caption") – Char(100)
The field that should be presented to the end user.
 - (3)Data Type ("data_type") – Enumeration
 - 2 – Short integer
 - 3 – Long integer
 - 4 – Single precision floating point
 - 5 – Double precision floating point
 - 6 – Currency
 - 7 – Date
 - 8 – String
 - 11 – Boolean
 - 99 – last maintenance date (internal use); handle as a string.
 - 98 – last maintenance user id (internal use); handle as a string.
 - (4)Field Length("field_length") – Short integer
 - (5)LookUp Values("lookup_values") – Char(255)
A list of choices that are to be presented to the end user. The end user will be allowed to make only one selection. To be used in conjunction with “stored_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (6)Stored Values ("stored_values") – Char(255)
A list of values that correspond to the lookup values. The data contained in this list represents the data that will be saved in place of the “lookup value”. To be used in conjunction with “lookup_values” and must be passed as a parameter when calling cLookup.GetLookupStoredValues (refer to GetLookupStoredValues method definition).
 - (7)Display Field ("display_field") – Boolean
Do not display the field if the value is False.
 - (8)Primary key ("primary_key") – Boolean
 - (9)Read only ("read_only") – Boolean
The field cannot be modified.
 - (10)Default Value ("default_value") – Char(255)
When in add mode, use this value as the default value.
 - (11) Required field ("required") – Boolean
A value is mandatory if this field is True.
 - (12) Minimum value ("min_value") – Char(255)
Use this value to define the minimum value for the field.
 - (13) Maximum value ("max_value") – Char(255)
Use this value to define the maximum value for the field.
 - (14) Audit flag ("audit_flag") – Boolean Internal use.
 - (15) Log flag ("log_flag") – Boolean Internal use.
 - (16) Old Value ("old_value") – Char(255)
Blank when in add mode; otherwise data from the database. The data comes in as a string but the client application should work w/the data using the correct data type definition as described by the “data_type” (see item #3)
 - (17) New value ("new_value") – Char(255)
The value is equal to the Old Value when existing data is first loaded. Ultimately it represents the end user’s input. Same rules as “Old Value” apply here.
- Note: When dealing with a field that has lookup values, this value must contain the “Stored value” for saving to the database.
- (18) Decimal places (“decimal_places”) – Short integer – Contains the number of decimal places for numeric fields
 - (19) Error code (“error_code”) – Long integer – validation error code: 0=valid data inputted, <> 0 → invalid data entered

- (20) table_id – Short Integer – Internal Use.
- (21) other_flags – Short Integer - Internal Use.
- (22) special_field – Short Integer - Internal Use.

.NET Example

The following example shows how to retrieve a list of its fields to maintain:

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim objFields As System.Data.DataSet
'
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
'
objFields = objCustomerMaint.GetItsFields(iCompanyID, 1008,1,0,123,cSysUser,lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    ' Success
    For Each row In objFields.Tables(0).Rows
        Debug.Print "ID=" & Row (0) & "Name=" & _
            Row (1) & " Description=" & Row (2)
    Next
EndIf
```

cCustomerMaint.AddItsProduct()

NOTE: For future use.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*AddItsProduct()*' method will be used to create a new its record on an existing customer wdms record.

```
Public Function AddItsProduct(ByVal p_iCompanyID As Short, _
                                ByVal p_lAccountNum As Integer, _
                                ByVal p_sSysUser As String, _
                                ByVal p_oRSItsFields As System.Data.DataSet, _
                                ByVal p_lWdmsNum As Integer, _
                                ByVal p_lItsProduct As Integer, _
                                ByRef p_lErrorCode As Integer) As System.Data.DataSet '
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSItsFields	System.Data.DataSet	The its record fields as selected by method GetItsFields() and updated by the client.
	p_lWdmsNum	Long Integer	Selected wdms location number
	p_lItsProduct	Long Integer	Selected ITS product
Return Values			
	p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type:	System.Data.DataSet	System.Data.DataSet	The its record fields as selected by method GetItsFields() and updated by the client. If the ITS record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to create a new its record for an existing customer wdms location record.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
' m_oRSItsFields is dataset created by GetItsFields () method.
m_oRSItsFields.Tables(0).Rows(0).Item(17) = 100
```

```
...  
,  
,  
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")  
,  
m_oRSItsFields = objCustomerMaint.AddItsProduct(iCompanyID, 1008,cSysUser,  
m_oRSItsFields,lWdmsNum,lItsProduct,lRetCode)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
EndIf
```

cCustomerMaint.UpdateItsProduct()

NOTE: For future use.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*UpdateItsProduct()*' method will be used to update an its record on an existing customer wdms record.

```
Public Function UpdateItsProduct(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_sSysUser As String, _
    ByVal p_oRSItsFields As System.Data.DataSet, _
    ByVal p_lWdmsNum As Integer, _
    ByVal p_lItsProduct As Integer, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short	The Energy Company ID
	p_lAccountNum	Long integer	Account number without the checkdigit
	p_sSysUser	String	Calling program (IVRUser/WebUser)
	p_oRSItsFields	System.Data.DataSet	The its record fields as selected by method GetItsFields() and updated by the client.
	p_lWdmsNum	Long Integer	Selected wdms location number
	p_lItsProduct	Long Integer	Selected ITS product

Return Values

	p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: System.Data.DataSet		System.Data.DataSet	The its record fields as selected by method GetItsFields() and updated by the client. If the ITS record is Successfully created the “New Value” will update the “Old Value”. Basically, the passed dataset is updated (if applicable) and returned

.NET Example

The following example shows how to update an ITS record for an existing customer wdms location record.

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
```

```
' m_oRSItsFields is dataset created by GetItsFields () method.
m_oRSItsFields.Tables(0).Rows(0).Item(17) = 100
```



```
...  
,  
,  
Set objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")  
,  
m_oRSItsFields = objCustomerMaint.UpdateItsProduct(iCompanyID, 1008,cSysUser,  
m_oRSItsFields,1,120,lRetCode)  
,  
If lRetCode <> 0 Then  
    ' Failure - Display error message  
EndIf
```

cCustomerMaint.GetContactTypes()

This class is used with V9.01 and lower level E3 databases. See the cContactInfo class for databases at E3 V10.01 and higher.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'GetContactTypes()' method will be used to get the different customer contact types from the ENERGY_MASTER database

```
Public Function GetContactTypes(ByVal p_sSysUser As String, _  
                                ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

Return Values	Data type	Description
p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: System.Data.DataSet	System.Data.DataSet	(0) Contact Type ID ("contact_type_id") -- Short integer (0-32767) (1) Description ("description") -- String - maximum 50 characters (2) Row status ("status") -- 1-character field (3) Contact type ("type") -- 1=Telephone, 2=Fax, 3=Email address

Return code	Description
96	Unknown Error Occurred.

.NET Example

The following example shows how to retrieve the contact type information

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim oContactTypes As System.Data.DataSet

objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")

oContactTypes = objCustomerMaint.GetContactTypes(cSysUser, lRetCode)

If lRetCode <> 0 Then
```

```
        ' Failure - Display error message
Else
    'Display The Returned Data
EndIf
```

cCustomerMaint.LoadCustContactInfo()

This class is used with V9.01 and lower level E3 databases. See the cContactInfo class for databases at E3 V10.01 and higher.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*LoadCustContactInfo()*' method will return a list of AR Contact types entered for 1 customer and/or location

```
Public Function LoadCustContactInfo(ByVal p_iCompanyID As Short, _
    ByVal p_lAccountNum As Integer, _
    ByVal p_iLocationType As Short, _
    ByVal p_iLocationNum As Short, _
    ByVal p_sSysUser As String, _
    ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short (16 bit integer)	The Energy Company ID
	p_lAccountNum	Integer (32 bit integer)	The account number ithout check digit
	p_iLocationType	Short (16 bit integer)	0=Account Level, 1=Delivery, 2=Service, 3=WDMS, 9=All
	p_iLocationNum	Short (16 bit integer)	0=All Locations, <>0=specific location
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

(0) Location Type ("location_type") -- Tinyint 0=Account Level, 1=Delivey, 2=Service, 3=WDMS
 (1) Location number ("location_num") -- Short integer (0-32767)
 (2) Row status ("status") -- char(1)
 (3) Contact Type ID ("contact_type_id") - Short integer (0-32767)
 (4) Contact value ("contact_value") - varchar(60)
 (5) Primary flag ("primary_flag") -- char(1) Y/N
 (6) Notes ("notes") -- varchar(255)
 (7) Notification flag ("notification") -- Tinyint 0=none, 1=delivery, 2=service, 3=both
 (8) Privacy flag ("privacy") -- Tinyint 0=none, 1=Opt In, 2=Opt Out
 (9) Contact Type ("type") -- Short integer 1=Telephone, 2=Fax, 3=email
 (10) Last Maintenance TimeStamp ("last_maintenance_dt") -- Varchar(30) -- Read-only field
 (11) Update error code ("error_no") -- Long Integer 0=Success, <>0 Failure to insert/update the row in the table

Return code	Description
96	Unknown Error Occurred.
211	This functionality is not available at this release.

.NET Example

The following example shows how to retrieve the customer contacts

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim oContactInfo As System.Data.DataSet
```

```
objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
```

```
oContactInfo = objCustomerMaint.LoadCustContactInfo(iCompany, lAccountNum,
iLocationType, iLocationNum, cSysUser, lRetCode)
'
```

```
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cCustomerMaint.SaveCustContactInfo()

This class is used with V9.01 and lower level E3 databases. See the cContactInfo class for databases at E3 V10.01 and higher.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The 'SaveCustContactInfo()' method will save the contact information

```
Public Function SaveCustContactInfo(ByVal p_iCompanyID As Short, _
                                   ByVal p_lAccountNum As Integer, _
                                   ByVal p_dsContactInfo As System.Data.DataSet, _
                                   ByVal p_sSysUser As String, _
                                   ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short (16 bit integer)	The Energy Company ID
	p_lAccountNum	Integer (32 bit integer)	The account number ithout check digit
	p_dsContactInfo	System.Data.DataSet	DataSet containing the customer contact info rows to save
	p_sSysUser	String	Calling program (IVRUser/WebUser)

Return Values

p_lErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

(0) Location Type ("location_type") -- Tinyint 0=Account Level, 1=Delivey, 2=Service, 3=WDMS
 (1) Location number ("location_num") -- Short integer (0-32767)
 (2) Row status ("status") -- char(1)
 (3) Contact Type ID ("contact_type_id") - Short integer (0-32767)
 (4) Contact value ("contact_value") - varchar(60)
 (5) Primary flag ("primary_flag") -- char(1) Y/N
 (6) Notes ("notes") -- varchar(255)
 (7) Notification flag ("notification") -- Tinyint 0=none, 1=delivery, 2=service, 3=both
 (8) Privacy flag ("privacy") -- Tinyint 0=none, 1=Opt In, 2=Opt Out
 (9) Contact Type ("type") -- Short integer 1=Telephone, 2=Fax, 3=email
 (10) Last Maintenance TimeStamp ("last_maintenance_dt") -- Varchar(30)
 -- Read-only field
 (11) Update error code ("error_no") -- Long Integer 0=Success, <>0 Failure to insert/update the row in the table

Return code	Description
96	Unknown Error Occurred.
211	This functionality is not available at this release.
337	Another user has already modified the selected information. Press OK to refresh this information then redo the action you just performed.
565	This data has been changed by another user, please refresh the data and try again.

.NET Example

The following example shows how to retrieve the customer contacts

```
Dim objCustomerMaint as Object
Dim lRetCode as Integer
Dim oContactInfo As System.Data.DataSet

objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")

oContactInfo = objCustomerMaint.SaveCustContactInfo(iCompany, lAccountNum,
oModifiedContactInfo, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
```

‘Display The Returned Data
End If

cCustomerMaint.DeleteCustContactInfo()

This class is used with V9.01 and lower level E3 databases. See the cContactInfo class for databases at E3 V10.01 and higher.

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The '*DeleteCustContactInfo()*' method will delete a contact for 1 account and location

```
Public Function DeleteCustContactInfo(ByVal p_iCompanyID As Short, _
                                       ByVal p_lAccountNum As Integer, _
                                       ByVal p_iLocationType As Short, _
                                       ByVal p_iLocationNum As Short, _
                                       ByVal p_iContactInfoID As Short, _
                                       ByVal p_sSysUser As String, _
                                       ByRef p_lErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	p_iCompanyID	Short (16 bit integer)	The Energy Company ID
	p_lAccountNum	Integer (32 bit integer)	The account number without check digit
	p_iLocationType	Short (16 bit integer)	0=Account Level, 1=Delivery, 2=Service, 3=WDMS
	p_iLocationNum	Short (16 bit integer)	0=All Locations, <>0=specific location
	p_iContactInfoID	Short (16 bit integer)	Short integer (0-32767) ID that determines the contact info type we should delete for this customer/location
	p_sSysUser	String	Calling program (IVRUser/WebUser)
Return Type: Long Integer (32 bit)		Long integer	0 – if Successful <> 0 Error Number

Return code	Description
96	Unknown Error Occurred.
211	This functionality is not available at this release.
337	Another user has already modified the selected information. Press OK to refresh this information then redo the action you just performed.

565	This data has been changed by another user, please refresh the data and try again.
899	For each contact type, only one entry can be designated as the primary contact at the customer or location level. Please try again.

.NET Example

The following example shows how to retrieve the customer contacts

```
Dim objCustomerMaint as Object
```

```
Dim lRetCode as Integer
```

```
objCustomerMaint = CreateObject("EnergyAPI.cCustomerMaint")
```

```
lRetCode = objCustomerMaint.DeleteCustContactInfo(iCompany, lAccountNum,  
iLocationType, iLocationNum, iContactInfoID, cSysUser)
```

```
,
```

```
If lRetCode <> 0 Then
```

```
    ' Failure - Display error message
```

```
Else
```

```
    'Display The Returned Data
```

```
End If
```

cCustomerMaint.LoadDeliveryStopCodes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method can be used to load existing Delivery Stop Codes currently setup in the system.

Method Signature:

```
Friend Function LoadDeliveryStopCodes(ByVal pCompanyId As Short, ByVal pSysUser As String) As DCEnergyAPI.CustomerMaint.DeliveryStopCode()
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: array of DCEnergyAPI.CustomerMaint.DeliveryStopCode			
StopCode		Byte	Stop code identifier
Description		String	Description of the stop code
Remarks		String	Remarks
ClearOnPosting		Boolean	Clears the stop code on posting
Status		string	Stop code status, Active or Inactive

Return code	Description
122	Invalid Input Parameters

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDeliveryStopCodes>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadDeliveryStopCodes>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
Using oCustomerMaint As EnergyAPI.CCustomerMaint = New EnergyAPI.CCustomerMaint
    Return oCustomerMaint.LoadDeliveryStopCodes(2, "WebUser")
End Using
```

cCustomerMaint.LoadDeliveryStopInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method can be used to load existing Delivery Stop information for a tank.

Method Signature:

```
Friend Function LoadDeliveryStopInfo(ByVal pCompanyId As Short, ByVal pSysUser As String, ByVal pAccountNum As Integer, ByVal pTankNum As Short) As DCEnergyAPI.CustomerMaint.DeliveryStopInfo
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	E3 account number without checkdigit
	pTankNum	Short	Tank number on the account
Return Type: DCEnergyAPI.CustomerMaint.DeliveryStopInfo			
	AccountNum	Integer	E3 account number without check digit
	TankNum	Short	Tank number for the account
	WarningCode	Integer	A warning code if available
	ErrorCode	Integer	An error code if available
	AllowDeliveryStop	Boolean	If false the user is not allowed to place a delivery stop
	StopCode	Byte	Numeric valid for the stop code, see LoadDeliveryStopCodes for description
	ReinstateDate	Datetime	The date time when deliveries will be reinstated
	Comment	String	Any free form text saved to customer log
	PhoneOrder	String	Y/N flag
	TicketNum	Integer	Non-zero value if an existing ticket is out for the tank
	TicketStatus	String	Status of the current ticket P = Pending U/A = Unprinted
	Last MaintenanceDT	String	Date & time stamp on the tank

Return code	Description
122	Invalid Input Parameters
985	There is currently a ticket out. Do you want to cancel it?
1341	Tank number not found for this customer.
1478	This tank has unprinted selected order(s); the entrie(s) will be deleted from the selection table when the tank is put on delivery stop.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDeliveryStopInfo>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pTankNum>1</add:pTankNum>
    </add:LoadDeliveryStopInfo>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
Using oCustomerMaint As EnergyAPI.CCustomerMaint = New EnergyAPI.CCustomerMaint
    Return oCustomerMaint.LoadDeliveryStopInfo(2, "WebUser", 25, 1)
End Using
```

cCustomerMaint.SetDeliveryStop()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method can be used to set or update a delivery stop for a tank.

Method Signature:

```
Public Function SetDeliveryStop(ByVal pCompanyId As Short, ByVal pSysUser As String
, ByVal pAccountNum As Integer, ByVal pTankNum As Short, ByVal pStopCode As Byte, B
yVal pReinstateDate As DateTime, ByVal pComment As String, ByVal pCurrentLmd As Str
ing) As Integer
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	E3 account number without checkdigit
	pTankNum	Short	Tank number on the account
	pStopCode	Byte	Non-zero Numeric stop code, see LoadDeliveryStopCodes for description
	pReinstateDate	Datetime	The date time when deliveries will be reinstated for the tank
	pComment	String	Free form comments to save to customer log
	pCurrentLmd	String	Current LMD as retrieved from LoadDeliveryStopInfo used to perform multi-user error checking
Return Type: Error code is any else 0			

Return code	Description
96	Unknown error.
122	Invalid Input Parameters
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
984	The date cannot be earlier than today.
1479	This user is not allowed to perform a delivery stop action.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SetDeliveryStop>
```

```
<add:pCompanyId>2</add:pCompanyId>
<add:pSysUser>WebUser</add:pSysUser>
<add:pAccountNum>25</add:pAccountNum>
<add:pTankNum>1</add:pTankNum>
<add:pStopCode>4</add:pStopCode>
<add:pReinstateDate>2015-12-01</add:pReinstateDate>
<add:pComment> SomeComment </add:pComment>
<add:pCurrentLmd>Jun 26 2015 5:21:25:893PM</add:pCurrentLmd>
</add:SetDeliveryStop>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
Using oCustomerMaint As EnergyAPI.CCustomerMaint = New EnergyAPI.CCustomerMaint
Return oCustomerMaint.SetDeliveryStop(2, "WebUser", 25, 1, 4, "2016-01-
01", "SomeComment", "Jun 26 2015 5:21:25:893PM")
End Using
```

cCustomerMaint.ClearDeliveryStop()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method can be used to clear delivery stop on a tank so the tank can be selected for future deliveries.

Method Signature:

```
Public Function ClearDeliveryStop(ByVal pCompanyId As Short, ByVal pSysUser As String,
ByVal pAccountNum As Integer, ByVal pTankNum As Short, ByVal pComment As String
, ByVal pCurrentLmd As String) As Integer
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	E3 account number without checkdigit
	pTankNum	Short	Tank number on the account
	pComment	String	Free form comments to save to customer log
	pCurrentLmd	String	Current LMD as retrieved from LoadDeliveryStopInfo used to perform multi-user error checking
Return Type: Error code is any else 0			

Return code	Description
96	Unknown error.
122	Invalid Input Parameters
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
984	The date cannot be earlier than today.
1479	This user is not allowed to perform a delivery stop action.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ClearDeliveryStop>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
```



```
<add:pTankNum>1</add:pTankNum>
<add:pComment>SomeComment</add:pComment>
<add:pCurrentLmd>Nov 12 2015 3:31:39:723PM</add:pCurrentLmd>
</add:ClearDeliveryStop>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
Using oCustomerMaint As EnergyAPI.CCustomerMaint = New EnergyAPI.CCustomerMaint
    Return oCustomerMaint.ClearDeliveryStop(2, "WebUser", 25, 1, "SomeComment", "Nov
12 2015 3:31:39:723PM")
End Using
```

cCustomerMaint.GetTemplates()

- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Customer Add & Modify Bus Edition

Description

This method can be used to retrieve the list of available templates for either adding or modifying records.

NOTE the Smartconnect user id (eg: WebUser) must be granted “Can Use” privileges to templates via E3 in order for the template(s) to be available in smartconnect)

Method Signature:

```
Public Function GetTemplates(ByVal pCompanyID As Short,
ByVal pRecordType As DCEnergyAPI.CustomerMaint.CustomerMaintenanceRecordType,
ByVal pMaintenanceMode As Short, ByVal pSysUser As String,
ByRef pErrorCode As Integer) As DCEnergyAPI.CustomerMaint.CustomerMaintenanceTemplate()
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pRecordType	<i>CustomerMaintenanceRecordType</i>	Record type
	pMaintenanceMode	<i>CustomerMaintenanceMode</i>	1:Add, 2:Modify
	pErrorCode	Integer	0: success < >0: error code
Return Type		<i>CustomerMaintenanceTemplate()</i>	Array of type DCEnergyAPI.CustomerMaint.CustomerMaintenanceTemplate

DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.*CustomerMaintenanceRecordType*

```
public enum CustomerMaintenanceRecordType
{
    [XmlEnum("1")]
    Customer = 1,
    [XmlEnum("2")]
    Delivery,
    [XmlEnum("3")]
    Service,
    [XmlEnum("4")]
    WDMS,
    [XmlEnum("5")]
    Meter,
    [XmlEnum("6")]
    ITS
}
```

```

    }
    DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.CustomerMaintenanceMode

```

```

public enum CustomerMaintenanceMode
{
    [XmlEnum("1")]
    Add = 1,
    [XmlEnum("2")]
    Modify
}

```

For use with XML use the numerical representation, not the name.

DCEnergyAPI.CustomerMaint.*CustomerMaintenanceTemplate*

Field name	description
TemplateID	Template id. To be used when calling: AddOrUpdateCustomerRecord()
TemplateDescription	Description. To be shown on client UI If necessary.

Examples:

Retrieve Customer Master templates (for adding a record)

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetTemplates>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pRecordType>1</add:pRecordType>
      <add:pMaintenanceMode>1</add:pMaintenanceMode>
      <add:pErrorCode>0</add:pErrorCode>
    </add:GetTemplates>
  </soap:Body>
</soap:Envelope>

```

Retrieve Customer Master templates (for modifying a record)

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetTemplates>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->

```

```
<add:pSysUser>WebUser</add:pSysUser>
<add:pRecordType>1</add:pRecordType>
<add:pMaintenanceMode>2</add:pMaintenanceMode>
<add:pErrorCode>0</add:pErrorCode>
</add:GetTemplates>
</soap:Body>
</soap:Envelope>
```

cCustomerMaint.AddOrUpdateCustomerRecord()

- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Customer Add & Modify Bus Edition

Description

This method can be used to add/update customer records (customer master, delivery, meter, service, wdms & its).

Method Signature:

```
Public Function AddOrUpdateCustomerRecord(ByVal pCompanyID As Short,
ByVal pSysUser As String,
ByVal pMaintenanceMode As DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.CustomerMaintenanceMode,
ByVal pTemplateID As Integer,
ByVal pRecordType As DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.CustomerMaintenanceRecordType,
ByVal pCustomerData As DCEnergyAPI.CustomerMaint.CustomerMaintenanceData,
ByRef pErrorCode As Integer) As DCEnergyAPI.CustomerMaint.CustomerMaintenanceData
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pTemplateID	Integer	Template id as retrieved by call to GetTemplates()
	pMaintenanceMode	<i>CustomerMaintenanceMode</i>	1:Add, 2:Modify
	pRecordType	<i>CustomerMaintenanceRecordType</i>	Record type to add
	pCustomerData	<i>CustomerMaintenanceData</i>	Customer information.
	pErrorCode	Integer	0: success < >0: error count***
Return Type		<i>CustomerMaintenanceData</i>	Updated Customer information.

pErrorCode: is a count of errors. Iterate through the pCustomerData.Fields collection and interrogate the “error_code” dataset field for value < > 0. Any field having error_code < > 0 must be corrected.

DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.CustomerMaintenanceRecordType

```
public enum CustomerMaintenanceRecordType
{
    [XmlEnum("1")]
    Customer = 1,
    [XmlEnum("2")]
    Delivery,
    [XmlEnum("3")]
    Service,
    [XmlEnum("4")]
    WDMS,
    [XmlEnum("5")]
    Meter,
    [XmlEnum("6")]
    ITS
}
```

DCEnergyAPI.CustomerMaint.CustomerMaintenanceData.CustomerMaintenanceMode

```
public enum CustomerMaintenanceMode
{
    [XmlEnum("1")]
    Add = 1,
    [XmlEnum("2")]
    Modify
}
```

For use with XML use the numerical representation, not the name.

DCEnergyAPI.CustomerMaint.CustomerMaintenanceData

Field name	description
AccountNum	Account # without checkdigit or zero when adding a customer.
LocationNum	Location#: Should be a valid non zero value when: 1) Modifying a delivery record, adding or modifying meter record for an existing delivery location 2) Modifying a service record 3) Modifying a wdms record, adding or modifying an ITS record for an existing WDMS location Should be zero when: 1) Adding or modifying a customer master record 2) Adding a delivery, service or WDMS location

ITSPProduct	ITS product: Should be a valid non zero value when: 1) Adding an ITS product to an existing WDMS record Should be zero when: 1) Adding or modifying a customer master, delivery, service, wdms, meter record.
Fields	Dataset of fields as retrieved by: GetCustMasterFields() GetTanksFields() GetMeterFields() GetServiceFields() GetWDMSFields() GetITSFields()

Examples:

Adding a Customer Master record

assumption call to *GetCustMasterFields()* was executed to retrieve fields linked to the template in question.

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add: AddOrUpdateCustomerRecord>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pTemplateID>323</add:pTemplateID>
      <add:pMaintenanceMode>1</add:pMaintenanceMode>
      <add:pRecordType>1</add:pRecordType>
      <!--Optional:-->
      <add:pCustomerData>
        <add:AccountNum>0</add:AccountNum>
        <add:LocationNumber>0</add:LocationNumber>
        <add:ITSPProduct>0</add:ITSPProduct>
        <!--Optional:-->
        <add:Fields>
          <DSCustomerFields>
            <CustomerFields >
              <field_position>8</field_position>
              <field_name>sort_code</field_name>
              <field_caption>Sort code</field_caption>
              <data_type>8</data_type>
              <field_length>6</field_length>
              <lookup_values xml:space="preserve"></lookup_values>
              <stored_values xml:space="preserve"></stored_values>
            </CustomerFields>
          </DSCustomerFields>
        </add:Fields>
      </add:pCustomerData>
    </add: AddOrUpdateCustomerRecord>
  </soap:Body>
</soap:Envelope>

```

```
<display_field>true</display_field>
<primary_key>false</primary_key>
<read_only>false</read_only>
<default_value xml:space="preserve"></default_value>
<required>true</required>
<min_value xml:space="preserve"></min_value>
<max_value xml:space="preserve"></max_value>
<audit_flag>false</audit_flag>
<log_flag>true</log_flag>
<old_value/>
<new_value>RUDY</new_value>
<decimal_places>0</decimal_places>
<error_code>0</error_code>
<table_id>900</table_id>
<other_flags>2</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>10</field_position>
  <field_name>division</field_name>
  <field_caption>Division</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values>EXEC load_divisions_$SP 3, VB_USER_ID, 1</lookup_values>
  <stored_values xml:space="preserve"></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value xml:space="preserve"></default_value>
  <required>true</required>
  <min_value xml:space="preserve"></min_value>
  <max_value xml:space="preserve"></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value/>
  <new_value>61</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>11</field_position>
  <field_name>type</field_name>
  <field_caption>Customer type</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
```


<lookup_values>SELECT distinct type, 'Type# ' + Convert(varchar,type),status = (select MIN(status) from TYPE_INFO B where B.type = A.type and B.division > 0) from TYPE_INFO A where type between 1 and 7 and division > 0</lookup_values>

<stored_values xml:space="preserve"></stored_values>

<display_field>true</display_field>

<primary_key>>false</primary_key>

<read_only>>false</read_only>

<default_value xml:space="preserve"></default_value>

<required>true</required>

<min_value xml:space="preserve"></min_value>

<max_value xml:space="preserve"></max_value>

<audit_flag>>false</audit_flag>

<log_flag>true</log_flag>

<old_value/>

<new_value>1</new_value>

<decimal_places>0</decimal_places>

<error_code>0</error_code>

<table_id>900</table_id>

<other_flags>0</other_flags>

<special_field>0</special_field>

</CustomerFields>

<CustomerFields >

<field_position>180</field_position>

<field_name>last_name</field_name>

<field_caption>Last Name</field_caption>

<data_type>8</data_type>

<field_length>40</field_length>

<lookup_values xml:space="preserve"></lookup_values>

<stored_values xml:space="preserve"></stored_values>

<display_field>true</display_field>

<primary_key>>false</primary_key>

<read_only>>false</read_only>

<default_value xml:space="preserve"></default_value>

<required>true</required>

<min_value xml:space="preserve"></min_value>

<max_value xml:space="preserve"></max_value>

<audit_flag>>false</audit_flag>

<log_flag>true</log_flag>

<old_value/>

<new_value>BANNER</new_value>

<decimal_places>0</decimal_places>

<error_code>0</error_code>

<table_id>900</table_id>

<other_flags>1</other_flags>

<special_field>0</special_field>

</CustomerFields>

<CustomerFields >

<field_position>5</field_position>

```

    <field_name>last_maintenance_dt</field_name>
    <field_caption>last maintenance date</field_caption>
    <data_type>99</data_type>
    <field_length>30</field_length>
    <display_field>false</display_field>
    <read_only>true</read_only>
    <required>false</required>
    <audit_flag>false</audit_flag>
    <log_flag>false</log_flag>
    <old_value/>
    <new_value/>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>0</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields >
    <field_position>6</field_position>
    <field_name>data_conversion</field_name>
    <field_caption>Data Conversion</field_caption>
    <data_type>8</data_type>
    <field_length>1</field_length>
    <lookup_values>Yes,No</lookup_values>
    <stored_values>Y,N</stored_values>
    <display_field>true</display_field>
    <read_only>false</read_only>
    <default_value>N</default_value>
    <required>false</required>
    <audit_flag>false</audit_flag>
    <log_flag>false</log_flag>
    <old_value/>
    <new_value>N</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>0</table_id>
    <other_flags>0</other_flags>
    <special_field>1</special_field>
  </CustomerFields>
</DSCustomerFields>
</add:Fields>
</add:pCustomerData>
<add:pErrorCode>0</add:pErrorCode>
</add: AddOrUpdateCustomerRecord >
</soap:Body>
</soap:Envelope>

```

Modifying a Customer Master record

assumption call to GetCustMasterFields() was executed to retrieve fields/values linked to the template in question.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add: AddOrUpdateCustomerRecord>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pTemplateID>325</add:pTemplateID>
      <add:pMaintenanceMode>2</add:pMaintenanceMode>
      <add:pRecordType>1</add:pRecordType>
      <!--Optional:-->
      <add:pCustomerData>
        <add:AccountNum>100732</add:AccountNum>
        <add:LocationNumber>0</add:LocationNumber>
        <add:ITSPProduct>0</add:ITSPProduct>
        <!--Optional:-->
        <add:Fields>
          <DSCustomerFields>
            <CustomerFields >
              <field_position>8</field_position>
              <field_name>sort_code</field_name>
              <field_caption>Sort code</field_caption>
              <data_type>8</data_type>
              <field_length>6</field_length>
              <lookup_values xml:space="preserve"></lookup_values>
              <stored_values xml:space="preserve"></stored_values>
              <display_field>true</display_field>
              <primary_key>false</primary_key>
              <read_only>false</read_only>
              <default_value xml:space="preserve"></default_value>
              <required>true</required>
              <min_value xml:space="preserve"></min_value>
              <max_value xml:space="preserve"></max_value>
              <audit_flag>false</audit_flag>
              <log_flag>true</log_flag>
              <old_value>RUDY</old_value>
              <new_value>RUDY</new_value>
              <decimal_places>0</decimal_places>
              <error_code>0</error_code>
              <table_id>900</table_id>
              <other_flags>2</other_flags>
              <special_field>0</special_field>
            </CustomerFields>
          <CustomerFields >
```

```

    <field_position>10</field_position>
    <field_name>division</field_name>
    <field_caption>Division</field_caption>
    <data_type>2</data_type>
    <field_length>2</field_length>
    <lookup_values>EXEC load_divisions_$SP 3, VB_USER_ID, 1</lookup_values>
    <stored_values xml:space="preserve"></stored_values>
    <display_field>true</display_field>
    <primary_key>false</primary_key>
    <read_only>false</read_only>
    <default_value xml:space="preserve"></default_value>
    <required>true</required>
    <min_value xml:space="preserve"></min_value>
    <max_value xml:space="preserve"></max_value>
    <audit_flag>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>61</old_value>
    <new_value>61</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields >
    <field_position>11</field_position>
    <field_name>type</field_name>
    <field_caption>Customer type</field_caption>
    <data_type>2</data_type>
    <field_length>2</field_length>
    <lookup_values>SELECT distinct type, 'Type# ' + Convert(varchar,type),status = (select
MIN(status) from TYPE_INFO B where B.type = A.type and B.division > 0) from TYPE_INFO A
where type between 1 and 7 and division > 0</lookup_values>
    <stored_values xml:space="preserve"></stored_values>
    <display_field>true</display_field>
    <primary_key>false</primary_key>
    <read_only>false</read_only>
    <default_value xml:space="preserve"></default_value>
    <required>true</required>
    <min_value xml:space="preserve"></min_value>
    <max_value xml:space="preserve"></max_value>
    <audit_flag>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>1</old_value>
    <new_value>1</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>900</table_id>

```

```
<other_flags>0</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>180</field_position>
  <field_name>last_name</field_name>
  <field_caption>Last Name</field_caption>
  <data_type>8</data_type>
  <field_length>40</field_length>
  <lookup_values xml:space="preserve"></lookup_values>
  <stored_values xml:space="preserve"></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value xml:space="preserve"></default_value>
  <required>true</required>
  <min_value xml:space="preserve"></min_value>
  <max_value xml:space="preserve"></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>BANNER</old_value>
  <new_value>BANNER Senior</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>900</table_id>
  <other_flags>1</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>5</field_position>
  <field_name>last_maintenance_dt</field_name>
  <field_caption>last maintenance date</field_caption>
  <data_type>99</data_type>
  <field_length>30</field_length>
  <display_field>false</display_field>
  <read_only>true</read_only>
  <required>false</required>
  <audit_flag>false</audit_flag>
  <log_flag>false</log_flag>
  <old_value>Apr 2 2018 7:59:36:890AM</old_value>
  <new_value>Apr 2 2018 7:59:36:890AM</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>0</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields >
```

```

    <field_position>6</field_position>
    <field_name>data_conversion</field_name>
    <field_caption>Data Conversion</field_caption>
    <data_type>8</data_type>
    <field_length>1</field_length>
    <lookup_values>Yes,No</lookup_values>
    <stored_values>Y,N</stored_values>
    <display_field>true</display_field>
    <read_only>false</read_only>
    <default_value>N</default_value>
    <required>false</required>
    <audit_flag>false</audit_flag>
    <log_flag>false</log_flag>
    <old_value>N</old_value>
    <new_value>N</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>0</table_id>
    <other_flags>0</other_flags>
    <special_field>1</special_field>
  </CustomerFields>
</DSCustomerFields>
</add:Fields>
</add:pCustomerData>
<add:pErrorCode>0</add:pErrorCode>
</add: AddOrUpdateCustomerRecord >
</soap:Body>
</soap:Envelope>

```

Adding a Delivery record

assumption call to GetTankFields() was executed to retrieve fields linked to the template in question

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add: AddOrUpdateCustomerRecord>
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pTemplateID>116</add:pTemplateID>
      <add:pMaintenanceMode>1</add:pMaintenanceMode>
      <add:pRecordType>2</add:pRecordType>
      <!--Optional:-->
      <add:pCustomerData>
        <add:AccountNum>100732</add:AccountNum>
        <add:LocationNumber>0</add:LocationNumber>
        <add:ITSPProduct>0</add:ITSPProduct>
      </add:pCustomerData>
    </add: AddOrUpdateCustomerRecord>
  </soap:Body>
</soap:Envelope>

```

```

<!--Optional:-->
<add:Fields>
  <DSCustomerFields >
    <CustomerFields>
      <field_position>8</field_position>
      <field_name>deviation</field_name>
      <field_caption>Deviation</field_caption>
      <data_type>4</data_type>
      <field_length>4</field_length>
      <lookup_values xml:space="preserve"></lookup_values>
      <stored_values xml:space="preserve"></stored_values>
      <display_field>true</display_field>
      <primary_key>false</primary_key>
      <read_only>false</read_only>
      <default_value xml:space="preserve"></default_value>
      <required>false</required>
      <min_value>-9999.9999</min_value>
      <max_value>9999.9999</max_value>
      <audit_flag>false</audit_flag>
      <log_flag>true</log_flag>
      <old_value>166.5</old_value>
      <new_value>166.5</new_value>
      <decimal_places>4</decimal_places>
      <error_code>0</error_code>
      <table_id>901</table_id>
      <other_flags>128</other_flags>
      <special_field>0</special_field>
    </CustomerFields>
    <CustomerFields>
      <field_position>29</field_position>
      <field_name>tax_jurisdiction</field_name>
      <field_caption>Tank county code</field_caption>
      <data_type>2</data_type>
      <field_length>2</field_length>
      <lookup_values>SELECT jurisdiction,jurisdiction_description,status FROM
TAX_JUR_DESC</lookup_values>
      <stored_values xml:space="preserve"></stored_values>
      <display_field>true</display_field>
      <primary_key>false</primary_key>
      <read_only>false</read_only>
      <default_value xml:space="preserve"></default_value>
      <required>true</required>
      <min_value xml:space="preserve"></min_value>
      <max_value xml:space="preserve"></max_value>
      <audit_flag>false</audit_flag>
      <log_flag>true</log_flag>
      <old_value>2</old_value>
      <new_value>2</new_value>

```

```

    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>901</table_id>
    <other_flags>0</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>117</field_position>
    <field_name>print_bal_on_remote_invoice</field_name>
    <field_caption>Print Balance on Remote Invoice</field_caption>
    <data_type>8</data_type>
    <field_length>1</field_length>
    <lookup_values>EXEC sys_get_alpha_code_desc_sp
1,'A',0,0,VB_USER_ID</lookup_values>
    <stored_values xml:space="preserve"></stored_values>
    <display_field>true</display_field>
    <primary_key>false</primary_key>
    <read_only>false</read_only>
    <default_value>N</default_value>
    <required>false</required>
    <min_value xml:space="preserve"></min_value>
    <max_value xml:space="preserve"></max_value>
    <audit_flag>false</audit_flag>
    <log_flag>true</log_flag>
    <old_value>N</old_value>
    <new_value>Y</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>901</table_id>
    <other_flags>4096</other_flags>
    <special_field>0</special_field>
  </CustomerFields>
  <CustomerFields>
    <field_position>4</field_position>
    <field_name>last_maintenance_dt</field_name>
    <field_caption>last maintenance date</field_caption>
    <data_type>99</data_type>
    <field_length>30</field_length>
    <display_field>false</display_field>
    <read_only>true</read_only>
    <required>false</required>
    <audit_flag>false</audit_flag>
    <log_flag>false</log_flag>
    <old_value>Feb 2 2018 10:22:38:993AM</old_value>
    <new_value>Feb 2 2018 10:22:38:993AM</new_value>
    <decimal_places>0</decimal_places>
    <error_code>0</error_code>
    <table_id>0</table_id>

```



```
        <other_flags>0</other_flags>
        <special_field>0</special_field>
    </CustomerFields>
</DSCustomerFields>
</add:Fields>
</add:pCustomerData>
<add:pErrorCode>0</add:pErrorCode>
</add: AddOrUpdateCustomerRecord >
</soap:Body>
</soap:Envelope>
```

Updating a Delivery record

assumption call to GetTankFields() was executed to retrieve fields/values linked to the template in question.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add: AddOrUpdateCustomerRecord >
      <add:pCompanyID>1600</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pTemplateID>293</add:pTemplateID>
      <add:pMaintenanceMode>2</add:pMaintenanceMode>
      <add:pRecordType>2</add:pRecordType>
      <!--Optional:-->
      <add:pCustomerData>
        <add:AccountNum>100732</add:AccountNum>
        <add:LocationNumber>1</add:LocationNumber>
        <add:ITSPProduct>0</add:ITSPProduct>
        <!--Optional:-->
        <add:Fields>
          <DSCustomerFields >
            <CustomerFields >
              <field_position>2</field_position>
              <field_name>delivery_instruction</field_name>
              <field_caption>Delivery instructions</field_caption>
              <data_type>8</data_type>
              <field_length>250</field_length>
              <lookup_values xml:space="preserve"></lookup_values>
              <stored_values xml:space="preserve"></stored_values>
              <display_field>true</display_field>
              <primary_key>false</primary_key>
              <read_only>false</read_only>
              <default_value xml:space="preserve"></default_value>
              <required>false</required>
              <min_value xml:space="preserve"></min_value>
              <max_value xml:space="preserve"></max_value>
              <audit_flag>false</audit_flag>
              <log_flag>true</log_flag>
              <old_value xml:space="preserve"></old_value>
              <new_value xml:space="preserve">del instructions</new_value>
              <decimal_places>0</decimal_places>
              <error_code>0</error_code>
              <table_id>901</table_id>
              <other_flags>1</other_flags>
              <special_field>0</special_field>
            </CustomerFields>
          </DSCustomerFields >
        </add:Fields>
      </add:pCustomerData>
    </add: AddOrUpdateCustomerRecord >
  </soap:Body>
</soap:Envelope>
```

```

<CustomerFields >
  <field_position>5</field_position>
  <field_name>product</field_name>
  <field_caption>Product code</field_caption>
  <data_type>2</data_type>
  <field_length>2</field_length>
  <lookup_values>EXEC load_products_$SP 1,"</lookup_values>
  <stored_values>ADD >0, ,A<&lt;</stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value xml:space="preserve"></default_value>
  <required>>false</required>
  <min_value xml:space="preserve"></min_value>
  <max_value xml:space="preserve"></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>>true</log_flag>
  <old_value>0</old_value>
  <new_value>2</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>901</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>

```

```

</CustomerFields>

```

```

<CustomerFields >
  <field_position>118</field_position>
  <field_name>bill_in_gross</field_name>
  <field_caption>Always Bill Gross on Remote Invoice</field_caption>
  <data_type>8</data_type>
  <field_length>1</field_length>
  <lookup_values>EXEC sys_get_alpha_code_desc_sp
1,'A',0,0,VB_USER_ID</lookup_values>
  <stored_values xml:space="preserve"></stored_values>
  <display_field>true</display_field>
  <primary_key>>false</primary_key>
  <read_only>>false</read_only>
  <default_value xml:space="preserve"></default_value>
  <required>>false</required>
  <min_value xml:space="preserve"></min_value>
  <max_value xml:space="preserve"></max_value>
  <audit_flag>>false</audit_flag>
  <log_flag>>true</log_flag>
  <old_value>N</old_value>
  <new_value>N</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>901</table_id>

```

```
<other_flags>4096</other_flags>
<special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>142</field_position>
  <field_name>acquisition_reason</field_name>
  <field_caption>Acquisition Reason</field_caption>
  <data_type>2</data_type>
  <field_length>5</field_length>
  <lookup_values>EXEC sys_get_code_description_sp 6,'
,1,0,VB_USER_ID</lookup_values>
  <stored_values xml:space="preserve"></stored_values>
  <display_field>true</display_field>
  <primary_key>false</primary_key>
  <read_only>false</read_only>
  <default_value xml:space="preserve"></default_value>
  <required>false</required>
  <min_value xml:space="preserve"></min_value>
  <max_value xml:space="preserve"></max_value>
  <audit_flag>false</audit_flag>
  <log_flag>true</log_flag>
  <old_value>0</old_value>
  <new_value>0</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>901</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
<CustomerFields >
  <field_position>5</field_position>
  <field_name>last_maintenance_dt</field_name>
  <field_caption>last maintenance date</field_caption>
  <data_type>99</data_type>
  <field_length>30</field_length>
  <display_field>false</display_field>
  <read_only>true</read_only>
  <required>false</required>
  <audit_flag>false</audit_flag>
  <log_flag>false</log_flag>
  <old_value>Apr 2 2018 8:14:16:220AM</old_value>
  <new_value>Apr 2 2018 8:14:16:220AM</new_value>
  <decimal_places>0</decimal_places>
  <error_code>0</error_code>
  <table_id>0</table_id>
  <other_flags>0</other_flags>
  <special_field>0</special_field>
</CustomerFields>
```

```
</DSCustomerFields>
</add:Fields>
</add:pCustomerData>
<add:pErrorCode>0</add:pErrorCode>
</add: AddOrUpdateCustomerRecord >
</soap:Body>
</soap:Envelope>
```

The same approach can be taken for service, wdms, meters and ITS.

CPlanPrice class

Overview

This is the main interface class that must be used to create special liquid product price records -- at this time the class only exposes methods that will allow the programmer to create records --. The programmer can use the following methods to create a slpp record:

- LoadStatus(): returns a list of status codes (Active, Inactive)
- LoadPlans(): returns list of plans associated to customer
- LoadTanks(): returns list of tanks associated to customer
- LoadProducts(): returns list of products prepended with "All"
- LoadAlternateEligibleProducts (): returns list of alternate products prepended with "All" and "None"
- LoadAlternateEligiblePricing (): returns list of alternate pricing
- LoadPlanCodesWithValues(): returns list of all valid plans and pertinent data related to each plan. This is needed to set the rules for plan creation.
- LoadPlanTypes(): returns list of all valid plan types (dollar, gallon, energy assistance, etc).
- LoadSalesTypes(): returns list of all valid sales types
- LoadCostTypes(): returns list of all valid cost types
- LoadDepartmentCodes(): returns list of all valid department codes
- GetProjectedUsage(): returns 12 months of projected usage for customer/tank
- Save(): Used to create the SLPP record for customer.

cPlanPrice.LoadStatus()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The '*LoadStatus()*' method will return a list of valid status codes.

```
Public Function LoadStatus(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef
pErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet
 (0) Id ("id") – String "A","I"
 (1) description ("description") - String
 ; Active, Inactive

.NET Example

The following example shows how to retrieve the list of statuses.

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")

oRet = oPlanPrice.LoadStatus(iCompany, cSysUser, lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice.LoadPlans()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadPlans()' method will return a list of plans associated (linked) to the customer.

Public Function LoadPlans(ByVal pCompanyID As Short, ByVal pAccountNum As Integer, ByVal pTankNum As Integer, ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	PCompanyID	Short (16 bit integer)	The Energy Company ID
	PCustomerNum	Long Integer	The account number without check digit
	PTankNum	Short Integer	The tank number
	PSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful <> 0 Error Number

Return Type: System.Data.DataSet

- (1) account_num - integer
- (2) tank_num - short integer
- (3) product_code - Short integer
- (4) plan_code - char(4)
- (5) transaction_key - integer
- (6) plan_type - short integer
- (7) priority - short integer
- (8) master_account – char(20)
- (9) status_code - char(1)
- (10) starting_date - date
- (11) ending_date - date
- (12) starting_gallons - float
- (13) remaining_gallons - float
- (14) starting_dollars - float
- (15) remaining_dollars - float
- (16) price_per_gallon - currency
- (17) price_differential - currency
- (18) follow_price_down - char(1)
- (19) agency_code - char(4)
- (20) agency_reference – char(15)
- (21) number_of_deliveries - short
- (22) alternate_prod_code - short integer
- (23) alternate_prod_pricing – tiny int.
- (24) starting_num_deliveries – short
- (25) remaining_num_deliveries - short
- (26) plan_committed_dt - date
- (27) plan_committed_by - char(20)
- (28) pre_buy_plan – short integer
- (29) last_maintenance_dt=Date (string)
- (30) card_site_num
- (31) sales_type
- (32) cost_type

(33) department_code
 (34) assigned_cost
 (35) pre_buy_expected_dollars
 (36) full_delivery
 (37) pre_buy_cents_gallon
 (38) remaining_pre_buy_dollars
 (39) open_ended_plan
 (40) use_offset_pc
 (41) po_number
 (42) variable_price_plan
 (43) price_increase_ratio
 (44) price_decrease_ratio
 (45) annual_increase
 (46) plan_years
 (47) min_annual_increase
 (48) minimum_price
 (49) maximum_price
 (50) use_dol_incl_taxes

The following example shows how to retrieve the list of plans for one customer:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet
Dim lAccountNum as Integer

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
lAccountNum=1234
oRet = oPlanPrice.LoadPlans(iCompany, lAccountNum, iTank,cSysUser, lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

XML Example

The following example shows how to retrieve the list of plans for one customer:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadPlans>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pCustomerNumber>1</add:pCustomerNumber>
      <add:pTankNum>0</add:pTankNum>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
```

```
</add:LoadPlans>  
</soap:Body>  
</soap:Envelope>
```

cPlanPrice.LoadTanks()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The '*LoadTanks()*' method will return a list of tanks associated (linked) to the customer.

Public Function LoadTanks(ByVal pCompanyID As Short, ByVal pAccountNum As Integer, ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	PCompanyID	Short (16 bit integer)	The Energy Company ID
	PAccountNum	Long Integer	The account number without check digit
	PSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	(0)Tank number ("tank_number") - Short integer (1)Delivery address ("delivery_address") - Char(250) (2)Product code ("product") -- Short integer (3)Product description ("product_desc") -- Char(30)

.NET Example

The following example shows how to retrieve the list of tanks for one customer:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet
Dim lAccountNum as Integer

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
lAccountNum=1234
oRet = oPlanPrice.LoadTanks(iCompany, lAccountNum, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice.LoadProducts()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The '*LoadProducts()*' method will return a list of liquid products and one extra entry for "All" products. The intended use is for the Product selection for the Plan.

Public Function LoadProducts(ByVal pCompanyID As Short,ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	(0)Product code Short Integer
	(1)Product description
	("product_desc") -- Char(30)
	(2) Status (Char(1) – "A"ctive,
	"I"nactive

.NET Example

The following example shows how to retrieve the list of products:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice.LoadProducts(iCompany, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadAlternateEligibleProducts()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadAlternateEligibleProducts()' method will return a list of liquid products and two extra entries – one for “All” products and one for “None” no products.

The intended use is for the Alternate Eligible Product selection for the Plan.

Public Function LoadAlternateEligibleProducts(ByVal pCompanyID As Short, BByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Long Integer	The account number without check digit
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet
 (0)Product code Short Integer
 (1)Product description
 ("product_desc") -- Char(30)
 (2) Status (Char(1) – “A”ctive,
 “I”nactive

.NET Example

The following example shows how to retrieve the list of Alternate Eligible Products

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadAlternateEligibleProducts (iCompany, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadAlternateEligiblePricing()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadAlternateEligiblePricing()' method will return a list of alternate pricing.

Public Function LoadAlternateEligiblePricing(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

- (0) ID ("id") – Short Integer
- (1) Description ("description") – char(50) ["SLPP", "Tank"]

.NET Example

The following example shows how to retrieve the list of alternate eligible pricing values:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadAlternateEligiblePricing (iCompany, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadPlanCodeValidationInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadPlanCodeValidationInfo()' method will return pertinent data for the specific plan that may/can be used when saving the slpp record for the customer.

```
Public Function LoadPlanCodeValidationInfo(ByVal pCompanyID As Short, ByVal pPlanCode As String, ByVal pSysUser As String, ByRef pErrorCode As Integer) As PlanCodeValidationInfo
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pPlanCode	String	Plan code
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values			
	pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: Class Object PlanCodeValidationInfo

.NET Example

The following example shows how to retrieve the list of plan codes and system values

```
Dim oPlanPrice as New EnergyAPI.cPlanPrice ()
Dim lRetCode as Integer
Dim planCodeInfo As New PlanCodeValidationInfo()

Dim companyID as Short =1
Dim planCode as String ="PLAN1"
planCodeInfo = oPlanPrice. LoadPlanCodesWithValues (companyID, planCode,cSysUser, lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

XML Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadPlanCodeValidationInfo>
```

```

<add:pCompanyID>1</add:pCompanyID>
<!--Optional:-->
<add:pPlanCode>PLAN1</add:pPlanCode>
<!--Optional:-->
<add:pSysUser>WebUser</add:pSysUser>
<add:pErrorCode>0</add:pErrorCode>
</add:LoadPlanCodeValidationInfo>
</soap:Body>
</soap:Envelope>

```

EnergyAPI.PlanCodeValidationInfo

field name	Description
string PlanCode	Plan code
bool GallonType	True if plan is gallon type
bool DollarType	True if plan is dollar type
bool NumDeliveriesType	True if plan is number of deliveries type
bool EnergyAssGallonsType	True if plan is energy assistance gallons type
bool EnergyAssDollarsType	True if plan is energy assistance dollars type
bool MasterType	True if plan is master type
decimal MinimumAssignedCost	minimum assigned cost to a price protection plan
decimal MaximumAssignedCost	maximum assigned cost to a price protection plan
bool HidassignedCost	When true, the assigned cost field is hidden
bool PreBuy	True if plan is prebuy
bool Renewable	
bool UseLowestPrice	When true, the system uses the lowest available price for the transaction: either the SLPP price or the standard price on the tank, i.e., as derived from the base price code, base price sublevel code, deviation and possibly even account discount percent. *NOTE* UseLowestPrice & FollowPriceDown are one and the same. The difference is the data type.
bool UseOffsetPostingCode	When true, the system uses the offset posting code specified for the price protection plan transaction
date EarliestStartDate	The earliest possible start date for the price protection plan
date LatestEndDate	The latest possible end date for the price protection plan
date LatestSystemDate	The latest system date. Plans with system dates after this date will not be allowed
decimal MinimumPrice	the minimum price for unit
decimal MaximumPrice	The maximum price for unit,
short MinimumNumDeliveries	The minimum number of deliveries for number of delivery type price protection plans
short MaximumNumDeliviries	The maximum number of deliveries for number of

	delivery type price protection plans
bool LogAdd	When true creates a log entry in a system template each time an individual price protection plan is added.
bool LogModify	When true, creates a log entry in a system template each time an individual price protection plan is modified
bool LogDelete	When true, creates a log entry in a system template each time an individual price protection plan is deleted.
bool VariablePricePlan	When true, the plan is a Variable Price Plan

cPlanPrice. LoadPlanCodesWithValues()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadPlanCodesWithValues()' method will return a list of valid (defined) plans including pertinent data specific to each plan that may/can be used when saving the slpp record for the customer.

Public Function LoadPlanCodesWithValues(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

- (0) Plan Code ("plan_code") char(4),
- (1) Description ("description") char(50),
- (2) Status ("status") char(1),
- (3) Gallon Type ("gallon_type") bit,
- (4) Dollar Type ("dollar_type") bit,
- (5) Number of Deliveries ("num_deliveries_type") bit,
- (6) Energy Assistance Gallons Type("energy_ass_gallons_type") bit,
- (7) Energy Assistance Dollars Type("energy_ass_dollars_type") bit,
- (8) Master Plan ("master_type") bit,
- (9) Default ASSigned Cost("default_assigned_cost") float,
- (10) Min Assigned Cost("minimum_assigned_cost") float,
- (11) Max Assigned Cost("maximum_assigned_cost") float,
- (12) Hide Assigned Cost("hide_assigned_cost") bit,
- (13) Pre Buy("pre_buy") bit,
- (14) Renewable("renewable") bit,
- (15) Use Lowest Price("use_lowest_price") bit,
- (16) Use Offsetting Posting Code("use_offset_posting_code") bit,
- (17) Earliest Start Date("earliest_start_date") date,
- (18) Latest End Date("latest_end_date") date,
- (19) Latest System Date("latest_system_date") date,
- (20) Default Start Date("default_start_date") date,

```

(21) Default End
Date("default_end_date") date,
(22) Default Price("default_price") float,
(23) Min Price("minimum_price") float,
(24) Max Price("maximum_price")
float,
(25) Default # of
Deliveries("default_num_deliveries")
short integer,
(26) Min # of
Deliveries("minimum_num_deliveries")
short integer,
(27) Max # of
Deliveries("maximum_num_deliveries")
short integer,
(28) Default Agency
Code("default_agency_code") char(4),
(29) Variable Plan Price
Indicator("variable_price_plan") short
integer,
(30) Price Increase
Ratio("price_increase_ratio") short
integer,
(31) Price Decrease
Ratio("price_decrease_ratio") short
integer,
(32) Annual
Increase("annual_increase") char(1),
(33) Plan Years("plan_years") short
integer,
(34) Min Annual
Increase("min_annual_increase") short
integer
(35) In Use Indicator("in_use short")
integer,

```

.NET Example

The following example shows how to retrieve the list of plan codes and system values

```

Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadPlanCodesWithValues (0, cSysUser, lRetCode)
,

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If

```

cPlanPrice. LoadPlanTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadPlanTypes()' method will return a list of valid (defined) plan types.

```
Public Function LoadPlanTypes(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef
pErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

- (0) Code ("code") – short integer
- (1) Description ("description") – char(50)

.NET Example

The following example shows how to retrieve the list of plan types:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadPlanTypes (iCompanyID, cSysUser, lRetCode)
'
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadSalesTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadSalesTypes()' method will return a list of valid (defined) sales types.

```
Public Function LoadSalesTypes(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef
pErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet

- (0) Code ("sales_type_id") – short integer
- (1) Description ("description") – char(50)
- (2) Status ("status") – char(1) ['A' or 'I'] ***

*** Only active 'lookup' records should be presented to the end user when used for creating new records.

.NET Example

The following example shows how to retrieve the list of sales types:

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadSalesTypes (iCompanyID, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadCostTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadCostTypes()' method will return a list of valid (defined) cost types.

```
Public Function LoadCostTypes(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef
pErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	(0) Code ("code") – short integer
	(1) Description ("description") – char(50)
	(2) Status ("status") – char(1)
	['A' or 'I'] ***

*** Only active 'lookup' records should be presented to the end user when used for creating new records.

.NET Example

The following example shows how to retrieve the list of cost types

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadCostTypes (iCompanyID, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. LoadDepartmentCodes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'LoadDepartmentCodes()' method will return a list of valid (defined) department codes.

Public Function LoadDepartmentCodes(ByVal pCompanyID As Short, ByVal pSysUser As String, ByRef pErrorCode As Integer) As System.Data.DataSet

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	(0) Code ("code") – short integer
	(1) Description ("description") – char(50)
	(2) Status ("status") – char(1)
	['A' or 'I'] ***

*** Only active 'lookup' records should be presented to the end user when used for creating new records.

.NET Example

The following example shows how to retrieve the list of department codes

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice. LoadDepartmentCodes(iCompanyID, cSysUser, lRetCode)
'

If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```


cPlanPrice. GetProjectedUsage()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'GetProjectedUsage()' method will return projected usage for the 12 month period.

```
Public Function GetProjectedUsage(ByVal pCompanyID As Short, _
    ByVal pAccountNum As Integer, _
    ByVal pTankNum As Integer, _
    ByVal pProduct As Short, _
    ByVal pDate As Date, _
    ByVal pSysUser As String, _
    ByRef pErrorCode As Integer) As System.Data.DataSet
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Long Integer	Customer # without checkdigit
	pTankNum	Short	Customer tank #
	pProduct	Short	Product #
	pDate	Date	Current date
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	(0) Month ("month") – short integer
	(1) Month Name ("month_name") – char(50)
	(2) Units ("units") – float

.NET Example

The following example shows how to retrieve the projected usage

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As System.Data.DataSet
Dim iTankNum as integer
Dim lAccountNum as long
Dim iProduct as integer
Dim dteDate as Date

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
lAccountNum =1234
iTankNum=1
```

```
iProduct=2
dteDate=""09/05/2006""
oRet = oPlanPrice. GetProjectedUsage (iCompanyID,
lAccountNum,iTankNum,iProduct,dteDate,cSysUser, lRetCode)
,
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

cPlanPrice. Save()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Plan Price SLPP Business Edition**

Description

The 'Save()' method can be used to create a new slpp record or update an existing slpp record for the customer.

NOTE A call to *LoadPlans()* must be made to retrieve the current data before an attempt is made to update an **existing** SLPP record. All data (including data that is not being modified must be sent to the Save).

Public Function Save(ByVal pCompanyID As Short, ByVal pSLPPInfo As EnergyAPI.SLPPData, ByVal pSysUser As String, ByRef pErrorCode As Integer) As EnergyAPI.SLPPData

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSLPPInfo	Object	Custom Object (refer to the SLPPData definition)
	pSysUser	String	Calling program (IVRUser/WebUser)

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

EnergyAPI.SLPPData	Custom Object (refer to the SLPPData definition)

.NET Example

The following example shows how to save the new SLPP record

```
Dim oPlanPrice as Object
Dim lRetCode as Integer
Dim oRet As SLPPData

oPlanPrice = CreateObject("EnergyAPI.cPlanPrice")
oRet = oPlanPrice.Save(pCompanyID, myNewSLPPInfo, sSysUser, lRetCode) '
If lRetCode <> 0 Then
    ' Failure - Display error message
Else
    'Display The Returned Data
End If
```

XML Example

The following example shows how to create/modify SLPP records

Creating a new SLPP record:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:Save>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSLPPInfo>
        <add:AccountNum>100</add:AccountNum>
        <add:TankNum>1</add:TankNum>
        <add:ProductCode>1</add:ProductCode>
        <add:Priority>10</add:Priority>
        <add:MasterAccount></add:MasterAccount>
        <add:StartingDate>1961-01-01</add:StartingDate>
        <add:EndingDate>2049-12-31</add:EndingDate>
        <add:StartingGallons>0</add:StartingGallons>
        <add:RemainingGallons>0</add:RemainingGallons>
        <add:StartingDollars>250</add:StartingDollars>
        <add:RemainingDollars>0</add:RemainingDollars>
        <add:PricePerGallon>0</add:PricePerGallon>
        <add:PriceDifferential>0</add:PriceDifferential>
        <add:FollowPriceDown>N</add:FollowPriceDown>
        <add:AgencyCode></add:AgencyCode>
        <add:AgencyReference></add:AgencyReference>
        <add:Usage1>0</add:Usage1>
        <add:Usage2>0</add:Usage2>
        <add:Usage3>0</add:Usage3>
        <add:Usage4>0</add:Usage4>
        <add:Usage5>0</add:Usage5>
        <add:Usage6>0</add:Usage6>
        <add:Usage7>0</add:Usage7>
        <add:Usage8>0</add:Usage8>
        <add:Usage9>0</add:Usage9>
        <add:Usage10>0</add:Usage10>
        <add:Usage11>0</add:Usage11>
        <add:Usage12>0</add:Usage12>
        <add:PlanCode>1</add:PlanCode>
        <add:StatusCode>A</add:StatusCode>
        <add:PlanType>2</add:PlanType>
        <add:CardSiteNum>0</add:CardSiteNum>
        <add:AlternateProdCode>0</add:AlternateProdCode>
        <add:AlternateProdPricing>0</add:AlternateProdPricing>
        <add:SalesType>0</add:SalesType>
      </add:pSLPPInfo>
    </add:Save>
  </soap:Body>
</soap:Envelope>
```

```

<add:CostType>0</add:CostType>
<add:DepartmentCode>0</add:DepartmentCode>
<add:AssignedCost>0</add:AssignedCost>
<add:PreBuyDollars>0</add:PreBuyDollars>
<add:FullDelivery>0</add:FullDelivery>
<add:PreBuyCentsGallon>0</add:PreBuyCentsGallon>
<add:RemainingPreBuyDollars>0</add:RemainingPreBuyDollars>
<add:OpenEndedPlan>0</add:OpenEndedPlan>
<add:StartingNumDeliveries>0</add:StartingNumDeliveries>
<add:UseOffsetPC>0</add:UseOffsetPC>
<add:RemainingNumDeliveries>0</add:RemainingNumDeliveries>
<add:PoNumber>0</add:PoNumber>
<add:VariablePricePlan>0</add:VariablePricePlan>
<add:PriceIncreaseRatio>0</add:PriceIncreaseRatio>
<add:PriceDecreaseRatio>0</add:PriceDecreaseRatio>
<add:AnnualIncrease>N</add:AnnualIncrease>
<add:PlanYears>50</add:PlanYears>
<add:MinAnnualIncrease>0</add:MinAnnualIncrease>
<add:MinimumPrice>0</add:MinimumPrice>
<add:MaximumPrice>0</add:MaximumPrice>
<add:PlanCommittedDT>2013-03-01</add:PlanCommittedDT>
<add:PlanCommittedBy></add:PlanCommittedBy>
<add:PreBuyPlan>1</add:PreBuyPlan>
<add:TransactionKey>0</add:TransactionKey>
<add>LastMaintenanceDate</add>LastMaintenanceDate>
</add:pSLPPInfo>
<add:pSysUser>WebUser</add:pSysUser>
<add:pErrorCode>0</add:pErrorCode>
</add:Save>
</soap:Body>
</soap:Envelope>

```

Updating an existing SLPP record:

```

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:Save>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSLPPInfo>
        <add:AccountNum>100</add:AccountNum>
        <add:TankNum>1</add:TankNum>
        <add:ProductCode>1</add:ProductCode>
        <add:Priority>10</add:Priority>
        <add:MasterAccount></add:MasterAccount>
        <add:StartingDate>1961-01-01</add:StartingDate>
        <add:EndingDate>2049-12-31</add:EndingDate>
        <add:StartingGallons>0</add:StartingGallons>

```

<add:RemainingGallons>0</add:RemainingGallons>
<add:StartingDollars>250</add:StartingDollars>
<add:RemainingDollars>0</add:RemainingDollars>
<add:PricePerGallon>0</add:PricePerGallon>
<add:PriceDifferential>0</add:PriceDifferential>
<add:FollowPriceDown>N</add:FollowPriceDown>
<add:AgencyCode></add:AgencyCode>
<add:AgencyReference></add:AgencyReference>
<add:Usage1>0</add:Usage1>
<add:Usage2>0</add:Usage2>
<add:Usage3>0</add:Usage3>
<add:Usage4>0</add:Usage4>
<add:Usage5>0</add:Usage5>
<add:Usage6>0</add:Usage6>
<add:Usage7>0</add:Usage7>
<add:Usage8>0</add:Usage8>
<add:Usage9>0</add:Usage9>
<add:Usage10>0</add:Usage10>
<add:Usage11>0</add:Usage11>
<add:Usage12>0</add:Usage12>
<add:PlanCode>1</add:PlanCode>
<add:StatusCode>A</add:StatusCode>
<add:PlanType>2</add:PlanType>
<add:CardSiteNum>0</add:CardSiteNum>
<add:AlternateProdCode>0</add:AlternateProdCode>
<add:AlternateProdPricing>0</add:AlternateProdPricing>
<add:SalesType>0</add:SalesType>
<add:CostType>0</add:CostType>
<add:DepartmentCode>0</add:DepartmentCode>
<add:AssignedCost>0</add:AssignedCost>
<add:PreBuyDollars>0</add:PreBuyDollars>
<add:FullDelivery>0</add:FullDelivery>
<add:PreBuyCentsGallon>0</add:PreBuyCentsGallon>
<add:RemainingPreBuyDollars>0</add:RemainingPreBuyDollars>
<add:OpenEndedPlan>0</add:OpenEndedPlan>
<add:StartingNumDeliveries>0</add:StartingNumDeliveries>
<add:UseOffsetPC>0</add:UseOffsetPC>
<add:RemainingNumDeliveries>0</add:RemainingNumDeliveries>
<add:PoNumber>0</add:PoNumber>
<add:VariablePricePlan>0</add:VariablePricePlan>
<add:PriceIncreaseRatio>0</add:PriceIncreaseRatio>
<add:PriceDecreaseRatio>0</add:PriceDecreaseRatio>
<add:AnnualIncrease>N</add:AnnualIncrease>
<add:PlanYears>50</add:PlanYears>
<add:MinAnnualIncrease>0</add:MinAnnualIncrease>
<add:MinimumPrice>0</add:MinimumPrice>
<add:MaximumPrice>0</add:MaximumPrice>
<add:PlanCommittedDT>2013-03-01</add:PlanCommittedDT>

```
<add:PlanCommittedBy></add:PlanCommittedBy>
<add:PreBuyPlan>1</add:PreBuyPlan>
<add:TransactionKey>12345</add:TransactionKey>
<add>LastMaintenanceDate>Nov 17 2017 4:01:42:526PM</add>LastMaintenanceDate>
</add:pSLPPInfo>
<add:pSysUser>WebUser</add:pSysUser>
<add:pErrorCode>0</add:pErrorCode>
</add:Save>
</soap:Body>
</soap:Envelope>
```

SLPPData**Description**

This is the custom object that must be instantiated and populated before the save can take place.

Definition

Field	DataType	Description
AccountNum	Integer	Customer # without checkdigit
TankNum	Short Integer	Associated Tank # or 0 for All
ProductCode	Short Integer	Product # or 0 for All
Priority	Short Integer	
MasterAccount	String	Full account #; cannot be same as AccountNum. Only valid when the selected plan is a “master” plan.
StartingDate	Date	Plan Start date
EndingDate	Date	Plan End Date
StartingGallons	Float	Starting gallons (Only use when plan type is gallon or unit, send 0 otherwise)
RemainingGallons	Float	Remaining Gallons (Only use when plan type is gallon or unit, send 0 otherwise)
StartingDollars	Float	Starting Dollars (Only use when plan type is dollar, send 0 otherwise)
RemainingDollars	Float	Remaining Dollars (Only use when plan type is dollar, send 0 otherwise)
PricePerGallon	Float	Price/Gallon (cents)
PriceDifferential	Float	Price Differential (cents)
FollowPriceDown	String	Follow Price Down
AgencyCode	String	Agency Code
AgencyReference	String	Agency Reference
Usage1	Integer	Projected Usage month 1
Usage2	Integer	Projected Usage month 2
Usage3	Integer	Projected Usage month 3
Usage4	Integer	Projected Usage month 4
Usage5	Integer	Projected Usage month 5
Usage6	Integer	Projected Usage month 6
Usage7	Integer	Projected Usage month 7
Usage8	Integer	Projected Usage month 8
Usage9	Integer	Projected Usage month 9
Usage10	Integer	Projected Usage month 10
Usage11	Integer	Projected Usage month 11
Usage12	Integer	Projected Usage month 12
PlanCode	String	Plan Code
StatusCode	String	“A”ctive or “I”nactive
PlanType	Short Integer	Plan Type
CardSiteNum	Short Integer	Card Site Number
AlternateProdCode	Short Integer	Alternate Product Code *NOTE*

		-1 : None 0: All >0: specific product code
AlternateProdPricing	Short Integer	Alternate Pricing 0: SLPP Plan 1: Tank pricing
SalesType	Short Integer	Sales Type (optional)
CostType	Short Integer	Cost Type (optional)
DepartmentCode	Short Integer	Department Code (optional)
AssignedCost	Float	Assigned Cost (cents)
PreBuyDollars	Float	Expected Pre buy dollars
FullDelivery	Short Integer	Full Delivery
PreBuyCentsGallon	Float	Pre Buy Gallons (cents)
RemainingPreBuyDollars	Float	Remaining Prebuy Dollars
OpenEndedPlan	Short Integer	Indicates an open ended plan
StartingNumDeliveries	Short Integer	Starting number of deliveries (Only use when plan type is number of deliveries, send 0 otherwise)
UseOffsetPC	Short Integer	Indicates whether to use offsetting posting codes
RemainingNumDeliveries	Short Integer	Remaining Number of deliveries (Only use when plan type is number of deliveries, send 0 otherwise)
PoNumber	String	PO# (optional)
VariablePricePlan	Short Integer	Indicates a variable plan ***
PriceIncreaseRatio	Short Integer	Price Increase Ratio ***
PriceDecreaseRatio	Short Integer	Price Decrease Ratio ***
AnnualIncrease	String	Annual Increase ***
PlanYears	Short Integer	Plan Years ***
MinAnnualIncrease	Short Integer	Minimum Annual Increase ***
MinimumPrice	Float	Minimum Price ***
MaximumPrice	Float	Maximum Price ***
PlanCommittedDT	Date	Plan opened date
PlanCommittedBy	String	Plan opened by
PreBuyPlan	Short Integer	Pre buy Plan flag (0 not prebuy, 1 prebuy)
TransactionKey	Long Integer	Transaction Unique key as retrieved by call to LoadPlans() 0: when creating a new slpp record <>0: transaction_key as returned by LoadPlans() when modifying an existing record
LastMaintenanceDate	Date (string)	Last maintenance date of existing record as retrieved by call to LoadPlans() Blank: when creating a new slpp record Date: last_maintenance_dt as returned by LoadPlans() when modifying an existing record.

*** Data that is retrieved from call to the *GetPlanCodesWithValues()* method.

cCustBatch class

Overview

This is the main interface class that must be used to create/update customer **master** records in batch mode.

CCustBatch.CustBatchUpdate()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The method cCustBatch.CustBatchUpdate() will allow for the updating of customer **master** records in batch mode.

```
public DataSet CustBatchUpdate(short pCompanyID, string pSysUser, DataSet
pSourceDS, int pTemplateID, ref int pErrorCode)
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pSourceDS***	System.Data.DataSet	
	pTemplateID	Int	Smartconnect master template to use.

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	The source dataset is returned. The columns error_no/err_desc will indicate success or failure of update.

pSourceDS: The source Dataset must comprise of columns whose name matches a field in the base table being updated. These column names will be mapped to the fields defined in the template and as long as there is a corresponding match in the template, the value in the dataset column will be written out to the field in the base table.

Mandatory columns:

“account_num”, int (account# without checkdigit)
 “error_code”, int (0 or error #)
 “error_desc”, string (blank or description of error_code)

.NET Example

Example: Update division to 25 for account 1234 and account 5678.

```
DataSet workingDS = new DataSet();
DataColumn col = null;
DataRow newRow = null;
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int32");
col.DefaultValue = 0;
col.ColumnName = "account_num";
workingDS.Tables[0].Columns.Add(col);
//
```

```
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int32");
col.DefaultValue = 0;
col.ColumnName = "error_code";
workingDS.Tables[0].Columns.Add(col);
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.String");
col.DefaultValue = "";
col.ColumnName = "error_desc";
workingDS.Tables[0].Columns.Add(col);
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int16");
col.DefaultValue = 0;
col.ColumnName = "division";
workingDS.Tables[0].Columns.Add(col);
//
newRow = workingDS.Tables[0].NewRow();
newRow["account_num"]=1234;
newRow["division"]=25;
//
workingDS.Tables[0].Rows.Add(newRow);
//
newRow = workingDS.Tables[0].NewRow();
newRow["account_num"]=5678;
newRow["division"]=25;
//
workingDS.Tables[0].Rows.Add(newRow);
//
int pErrorCode =0;
workingDS=cCustBatch.CustBatchUpdate (2,"WebUser",workingDS,-900,pErrorCode);
// if <>0 then not all updates were successful.
if (pErrorCode!=0)
{
    foreach (DataRow row in updCustomerDS.Tables[0].Rows)
    {
        if (row["error_code"]!=0)
        {
            // handle & show row["error_desc"]
        }
    }
}
```

CCustBatch.CustBatchAdd()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The method cCustBatch.CustBatchAdd() will allow for the inserting of customer **master** records in batch mode.

```
public DataSet CustBatchAdd(short pCompanyID, string pSysUser, DataSet pSourceDS, int
pTemplateID, ref int pErrorCode)
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pSourceDS***	System.Data.DataSet	
	pTemplateID	Int	Smartconnect master template to use.

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet	The source dataset is returned. The columns error_no/err_desc will indicate success or failure of update.

pSourceDS: The source Dataset must comprise of columns whose name matches a field in the base table being updated. These column names will be mapped to the fields defined in the template and as long as there is a corresponding match in the template, the value in the dataset column will be written out to the field in the base table.

Mandatory columns:

“account_num”, int (must be 0)
 “last_name”,string
 “street1”,string
 “city”,string
 “state”,string
 “postal_code”,string
 “sort_code”,string
 “division”,short
 “type”,short
 “credit_time_code”,short
 “finance_charge”,string
 “instant_invoice”,string
 “stat_day”,short
 “error_code”,int (0 or error #)
 “error_desc”,string (blank or description of error_code)

.NET Example

```
DataSet workingDS = new DataSet();
DataColumn col = null;
DataRow newRow = null;
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int32");
col.DefaultValue = 0;
col.ColumnName = "account_num";
workingDS.Tables[0].Columns.Add(col);
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int32");
col.DefaultValue = 0;
col.ColumnName = "error_code";
workingDS.Tables[0].Columns.Add(col);
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.String");
col.DefaultValue = "";
col.ColumnName = "error_desc";
workingDS.Tables[0].Columns.Add(col);
//
col = new DataColumn();
col.DataType = System.Type.GetType("System.String");
col.DefaultValue = "";
col.ColumnName = "last_name";
workingDS.Tables[0].Columns.Add(col);
...
col = new DataColumn();
col.DataType = System.Type.GetType("System.Int16");
col.DefaultValue = 0;
col.ColumnName = "stat_day";
workingDS.Tables[0].Columns.Add(col);
//
newRow = workingDS.Tables[0].NewRow();
newRow["account_num"]=0;
newRow["last_name"]="Smith";
/* set up all other fields...*/
...
newRow["stat_day"]=1;
workingDS.Tables[0].Rows.Add(newRow);
//
newRow = workingDS.Tables[0].NewRow();
newRow["account_num"]=0;
newRow["last_name"]="Jones";
/* set up all other fields...*/
...
newRow["stat_day"]=2;
workingDS.Tables[0].Rows.Add(newRow);
//
int pErrorCode =0;
workingDS=cCustBatch.CustBatchAdd (2,"WebUser",workingDS,-900,pErrorCode);
```

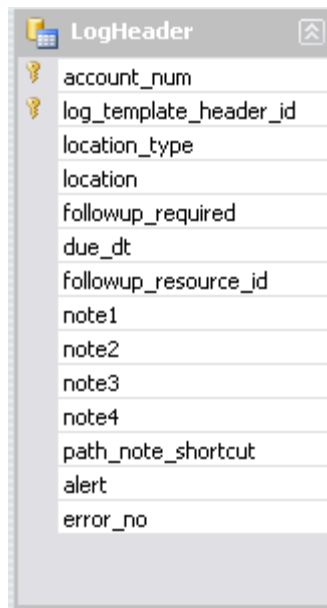
```
int accountNum =0;
foreach (DataRow row in updCustomerDS.Tables[0].Rows)
{
    if (row["error_code"]!=0)
    {
        // handle & show row["error_desc"]
    }
    else {
        // retrieve account #
        accountNum = row["account_num"];
        // do something...
    }
}
```

CCustBatch.GetLogHeaderDS()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

The method cCustBatch.GetLogHeaderDS() will return a DataSet of type DSLogHeader.



The screenshot shows a table named 'LogHeader' with the following fields:

account_num
log_template_header_id
location_type
location
followup_required
due_dt
followup_resource_id
note1
note2
note3
note4
path_note_shortcut
alert
error_no

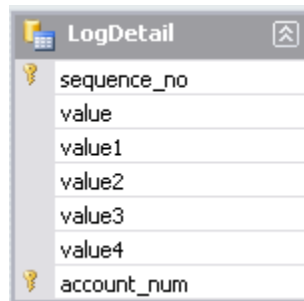
(Refer to method [cCustomerLog.CreateLogEntry](#) parameters for the datatype definition of the above listed fields).

CCustBatch.GetLogDetailDS()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

The method cCustBatch.GetLogDetailDS() will return a DataSet of type DSLogDetail.



LogDetail	
sequence_no	
value	
value1	
value2	
value3	
value4	
account_num	

(DSLogDetail can be handled much in the same way p_vDetails is for method [cCustomerLog.CreateLogEntry](#).)

CCustBatch.CustBatchLogEntry()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Create Log Entries Business Edition**

Description

The method cCustBatch.CustBatchLogEntry() will allow for the creation of logs in batch mode.

```
public DataSet CustBatchLogEntry(short pCompanyID, string pSysUser, DataSet
pLogHdrDS, DataSet pLogDtlDS, ref int pErrorCode)
```

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pLogHdrDS	System.Data.DataSet	
	pLogDtlDS	System.Data.DataSet	

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number

Return Type: System.Data.DataSet

System.Data.DataSet An updated pLogHdrDS is returned

pLogHdrDS is initialized by calling method GetLogHeaderDS

pLogDtlDS is initialized by calling method GetLogDetailDS

.NET Example

Example: create logs for account 1234 and 5678.

```
DataSet logHdrDS = cCustBatch.GetLogHeaderDS();
DataSet logDtlDS = cCustBatch.GetLogDetailDS();
DataRow newRow = null;
//
newRow = logHdrDS.Tables[0].NewRow();
newRow["account_num"]=1234;
newRow["log_template_header_id"]=90;
...
newRow["note1"]="this will be created via SmartConnect batch mode";
...
logHdrDS.Tables[0].Rows.Add(newRow);
//
newRow = logHdrDS.Tables[0].NewRow();
newRow["account_num"]=5678;
newRow["log_template_header_id"]=90;
...
newRow["note1"]="2nd log:this will be created via SmartConnect batch mode";
...
logHdrDS.Tables[0].Rows.Add(newRow);
//
```

```
newRow = logDt1DS.Tables[0].NewRow();
newRow["account_num"]=1234;
newRow["sequence_no"]=1;
newRow["value"]=<old value>;
newRow["value1"]=<new value>;
logDt1DS.Tables[0].Rows.Add(newRow);
//
newRow = logDt1DS.Tables[0].NewRow();
newRow["account_num"]=1234;
newRow["sequence_no"]=2;
newRow["value"]=<old value>;
newRow["value1"]=<new value>;
logDt1DS.Tables[0].Rows.Add(newRow);
//
newRow = logDt1DS.Tables[0].NewRow();
newRow["account_num"]=5678;
newRow["sequence_no"]=1;
newRow["value"]=<old value>;
newRow["value1"]=<new value>;
logDt1DS.Tables[0].Rows.Add(newRow);
//
newRow = logDt1DS.Tables[0].NewRow();
newRow["account_num"]=5678;
newRow["sequence_no"]=2;
newRow["value"]=<old value>;
newRow["value1"]=<new value>;
logDt1DS.Tables[0].Rows.Add(newRow);
//
int pErrorCode =0;
logHdrDS = cCustBatch.CustBatchLogEntry (2,"WebUser",logHdrDS,logDt1DS,pErrorCode)
//
foreach (DataRow row in logHdrDS.Tables[0].Rows)
{
    if (row["error_no"]!=0)
    {
        // handle
    }
    else {
        // do something...
    }
}
```

CCustBatch.BatchUpdateMaster()

(not available as web service)

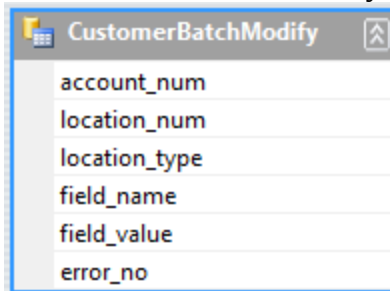
- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The method cCustBatch.BatchUpdateMaster() will allow for the mass update of account records in “batch mode”. The method will allow for one or more fields to be updated per account.

```
public DCEnergyAPI.CustomerMaint.DSCustomerBatchModify BatchUpdateMaster(short pCompanyID,
string pSysUser,DCEnergyAPI.CustomerMaint.DSCustomerBatchModify pDSBatchUpdate, bool pCreateLog,
ref int pErrorCount)
```

DSCustomerBatchModify



The screenshot shows a form titled "CustomerBatchModify" with a list of input fields: account_num, location_num, location_type, field_name, field_value, and error_no. Each field has a corresponding input box.

account_num: account # without the checkdigit

location_num: 0 for master account, location # for delivery, wdms, service etc.

location_type: 0 – Account

field_name*:** the field name as it exists in the Sybase table. Refer to the Appendix

field_value: regardless of data type must be inputted as string and it represents the NEW VALUE.

error_no: return value of 0 – success, <> 0 – value from Resource. Description can be retrieved using cUtility.GetErrorDescription ()

*** field_name: only fields that are currently available via the smart connect template for the given record type will be available. As well, only fields that exist in the base table will be available. Any field like “delivery_address” that comprises a series of fields in an extension table will not be available.

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSBatchUpdate	System.Data.DataSet	
	pCreateLog***	Boolean	True – Create log entry, False – Do not create log entry

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error count

Return Type: System.Data.DataSet

System.Data.DataSet

DSCustomerBatchModify

*** **pCreateLog** : It should be understood that creating logs will impact the update execution time.

.NET Example (ACCOUNTS)

(C#)

```
DataSet oDS = null;
```

```
// Get list of 1000 accounts.. "GetAccountList" is not part of SmartConnect but would be some  
// method created by developer; meant only as example.
```

```
oDS = GetAccountList(1000);
```

```
DataRow oRow = null;
```

```
EnergyAPI.cCustBatch custBatchUpdate = new EnergyAPI.cCustBatch();
```

```
DCEnergyAPI.CustomerMaint.DSCustomerBatchModify CustFields = new  
DCEnergyAPI.CustomerMaint.DSCustomerBatchModify();
```

```
// Change two fields in ACCOUNTS with the exact value FOR ALL 1000 accounts.
```

```
int ErrorCode = 0;
```

```
short locationNum = 0;
```

```
short locationType=0;
```

```
foreach (DataRow oRow in oDS.Tables["ACCOUNTS"].Rows) {
```

```
    CustFields.Tables(0).Rows.Add(oRow["account_num"], locationNum, locationType,  
    "prem_bill_code", "1");
```

```
    CustFields.Tables(0).Rows.Add(oRow["account_num"], locationNum, locationType,  
    "prem_bill_base", "112");
```

```
}
```

```
bool createLog = true;
```

```
short companyID =2;
```

```
CustFields= custBatchUpdate.BatchUpdateMaster(companyID, "WebUser", CustFields, createLog,  
ErrorCode);
```

```
If (ErrorCode !=0)
```

```
{
```

```
    // flip through CustFields and examine ["error_no"]
```

```
}
```

(VB)

```
Dim oDS As DataSet = Nothing
```

“GetAccountList” is not part of SmartConnect but would be some
‘ method created by developer; meant only as example.

```
oDS = utility.GetAccountList(1000)
```

```
Dim oRow As DataRow = Nothing
```

```
Dim custBatchUpdate As New EnergyAPI.cCustBatch()
```

```
Dim CustFields As New DCEnergyAPI.CustomerMaint.DSCustomerBatchModify()
```

```
Dim ErrorCode As Integer=0
```

```
Dim locationNum as short =0
```

```
Dim locationType as Short =0
```

```
For Each oRow In oDS.Tables("ACCOUNTS").Rows
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), locationNum, locationType,  
    "prem_bill_code", "1")
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), locationNum, locationType,  
    "prem_bill_base", "112")
```

```
Next
```

```
Dim createLog as Boolean =true
```

```
Dim companyID as Short =2
```

```
CustFields=custBatchUpdate.BatchUpdateMaster(companyID, "WebUser", CustFields, createLog,  
ErrorCode)
```

```
If (ErrorCode <> 0) then
```

```
    ‘flip through CustFields and examine [“error_no”]
```

```
End if
```

CCustBatch.BatchUpdateTank()

(not available as web service)

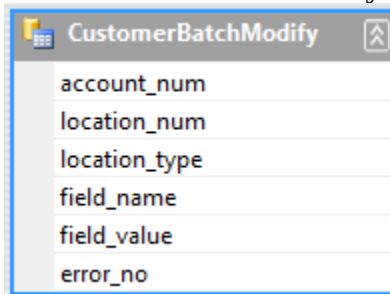
- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The method cCustBatch.BatchUpdateTank() will allow for the mass update of account tank records in “batch mode”. The method will allow for one or more fields to be updated per account/tank.

```
public DCEnergyAPI.CustomerMaint.DSCustomerBatchModify BatchUpdateTank(short pCompanyID, string
pSysUser,DCEnergyAPI.CustomerMaint.DSCustomerBatchModify pDSBatchUpdate, bool pCreateLog, ref int
pErrorCount)
```

DSCustomerBatchModify



Field Name
account_num
location_num
location_type
field_name
field_value
error_no

account_num: account # without the checkdigit

location_num: 0 for master account, location # for delivery, wdms, service etc.

location_type: 1 – Tank

field_name*:** the field name as it exists in the Sybase table. Refer to the Appendix

field_value: regardless of data type must be inputted as string and it represents the NEW VALUE.

error_no: return value of 0 – success, <> 0 – value from Resource. Description can be retrieved using cUtility.GetErrorDescription ()

*** field_name: only fields that are currently available via the smart connect template for the given record type will be available. As well, only fields that exist in the base table will be available. Any field like “delivery_address” that comprises a series of fields in an extension table will not be available.

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSBatchUpdate	System.Data.DataSet	
	pCreateLog***	Boolean	True – Create log entry, False – Do not create log entry

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error count

Return Type: System.Data.DataSet

System.Data.DataSet

DSCustomerBatchModify

*** **pCreateLog** : It should be understood that creating logs will impact the update execution time.

.NET Example (TANKS)

(C#)

```
DataSet oDS = null;
```

```
// "GetTankList" is not part of SmartConnect but would be some
```

```
// method created by developer; meant only as example.
```

```
oDS = GetTankList(1000);
```

```
int ErrorCode =0;
```

```
short locationType = 1;
```

```
// Update the same 2 fields/values for 1000 tank records.
```

```
foreach (DataSet oRow in oDS.Tables("TANKS").Rows) {
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("tank_num"), locationType,  
    "product", "6");
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("tank_num"), locationType,  
    "tax_1", "N");
```

```
}
```

```
short companyID = 2;
```

```
bool createLog = false;
```

```
CustFields = custBatchUpdate.BatchUpdateTank(companyID, "WebUser", CustFields, createLog,  
ErrorCode);
```

```
if (ErrorCode !=0)
```

```
{
```

```
    // flip through the CustFields and verify ["error_no"]
```

```
}
```


(VB)

Dim oDS as DataSet

oDS = GetTankList(1000)

Dim custBatchUpdate As New EnergyAPI.cCustBatch()

Dim CustFields As New DCEnergyAPI.CustomerMaint.DSCustomerBatchModify()

Dim locationType As Short = 1

For Each oRow In oDS.Tables("TANKS").Rows

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("tank_num"), locationType,
 "product", "6")

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("tank_num"), locationType,
 "tax_1", "N")

Next

Dim createLog as Boolean = True

Dim companyID as Short = 2

CustFields=custBatchUpdate.BatchUpdateTank(companyID, "WebUser", CustFields, createLog,
ErrorCode)

If (ErrorCode <> 0) then

 'flip through CustFields and examine ["error_no"]

End if

CCustBatch.BatchUpdateService()

(not available as web service)

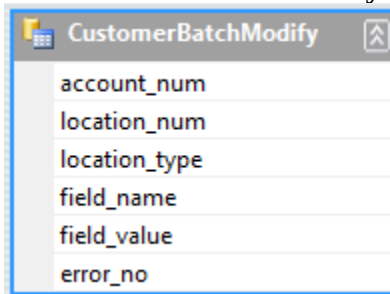
- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

The method cCustBatch.BatchUpdateService() will allow for the mass update of account tank records in “batch mode”. The method will allow for one or more fields to be updated per account/tank.

```
public DCEnergyAPI.CustomerMaint.DSCustomerBatchModify BatchUpdateService(short pCompanyID,
string pSysUser,DCEnergyAPI.CustomerMaint.DSCustomerBatchModify pDSBatchUpdate, bool pCreateLog,
ref int pErrorCount)
```

DSCustomerBatchModify



Field Name
account_num
location_num
location_type
field_name
field_value
error_no

account_num: account # without the checkdigit

location_num: 0 for master account, location # for delivery, wdms, service etc.

location_type: 1 – Tank

field_name*:** the field name as it exists in the Sybase table. Refer to the Appendix

field_value: regardless of data type must be inputted as string and it represents the NEW VALUE.

error_no: return value of 0 – success, <> 0 – value from Resource. Description can be retrieved using cUtility.GetErrorDescription ()

*** field_name: only fields that are currently available via the smart connect template for the given record type will be available. As well, only fields that exist in the base table will be available. Any field like “delivery_address” that comprises a series of fields in an extension table will not be available.

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSBatchUpdate	System.Data.DataSet	
	pCreateLog***	Boolean	True – Create log entry, False – Do not create log entry

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error count

Return Type: System.Data.DataSet

System.Data.DataSet

DSCustomerBatchModify

*** **pCreateLog** : It should be understood that creating logs will impact the update execution time.

.NET Example (SERVICE)

(C#)

```
DataSet oDS = null;
```

```
// "GetServiceList" is not part of SmartConnect but would be some
```

```
// method created by developer; meant only as example.
```

```
oDS = GetServiceList(1000);
```

```
int ErrorCode =0;
```

```
short locationType = 2;
```

```
// Update the same 2 fields/values for 1000 service records.
```

```
foreach (DataSet oRow in oDS.Tables("SERVICE").Rows) {
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("service_num"), locationType,  
    "product", "6");
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("service_num"), locationType,  
    "tax_1", "N");
```

```
}
```

```
short companyID = 2;
```

```
bool createLog = false;
```

```
CustFields = custBatchUpdate.BatchUpdateService(companyID, "WebUser", CustFields, createLog,  
ErrorCode);
```

```
if (ErrorCode !=0)
```

```
{
```

```
    // flip through the CustFields and verify ["error_no"]
```

```
}
```

(VB)

Dim oDS as DataSet

oDS = GetServiceList(1000)

Dim custBatchUpdate As New EnergyAPI.cCustBatch()

Dim CustFields As New DCEnergyAPI.CustomerMaint.DSCustomerBatchModify()

Dim locationType As Short = 2

For Each oRow In oDS.Tables("SERVICE").Rows

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("service_num"), locationType,
 "product", "6")

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("service_num"), locationType,
 "tax_1", "N")

Next

Dim createLog as Boolean = True

Dim companyID as Short = 2

CustFields=custBatchUpdate.BatchUpdateService(companyID, "WebUser", CustFields, createLog,
ErrorCode)

If (ErrorCode <> 0) then

 ‘flip through CustFields and examine [“error_no”]

End if

CCustBatch.BatchUpdateWDMS()

(not available as web service)

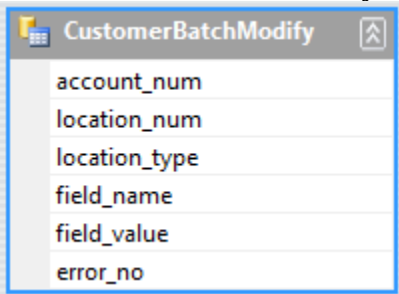
- License Required: SmartConnect API Customer Look-up Business Edition
- License Required: SmartConnect API Customer Add & Modify Bus Edition

Description

The method cCustBatch.BatchUpdateWDMS() will allow for the mass update of account tank records in “batch mode”. The method will allow for one or more fields to be updated per account/tank.

```
public DCEnergyAPI.CustomerMaint.DSCustomerBatchModify BatchUpdateWDMS(short pCompanyID,
string pSysUser,DCEnergyAPI.CustomerMaint.DSCustomerBatchModify pDSBatchUpdate, bool pCreateLog,
ref int pErrorCount)
```

DSCustomerBatchModify



account_num: account # without the checkdigit
location_num: 0 for master account, location # for delivery, wdms, service etc.
location_type: 1 – Tank
field_name***: the field name as it exists in the Sybase table. Refer to the Appendix
field_value: regardless of data type must be inputted as string and it represents the NEW VALUE.
error_no: return value of 0 – success, <> 0 – value from Resource. Description can be retrieved using cUtility.GetErrorDescription ()

*** field_name: only fields that are currently available via the smart connect template for the given record type will be available. As well, only fields that exist in the base table will be available. Any field like “delivery_address” that comprises a series of fields in an extension table will not be available.

INPUT PARAMETERS

Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSBatchUpdate	System.Data.DataSet	
	pCreateLog***	Boolean	True – Create log entry, False – Do not create log entry

Return Values

pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error count

Return Type: System.Data.DataSet

System.Data.DataSet	DSCustomerBatchModify

*** **pCreateLog** : It should be understood that creating logs will impact the update execution time.

.NET Example (WDMS)

(C#)

```
DataSet oDS = null;
```

```
// "GetWDMSList" is not part of SmartConnect but would be some
```

```
// method created by developer; meant only as example.
```

```
oDS = GetWDMSList(1000);
```

```
int ErrorCode =0;
```

```
short locationType = 3;
```

```
// Update the same 2 fields/values for 1000 WDMS records.
```

```
foreach (DataSet oRow in oDS.Tables("WDMS").Rows) {
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("location_num"), locationType,  
    "product", "6");
```

```
    CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("location_num"), locationType,  
    "tax_1", "N");
```

```
}
```

```
short companyID = 2;
```

```
bool createLog = false;
```

```
CustFields = custBatchUpdate.BatchUpdateWDMS(companyID, "WebUser", CustFields, createLog,  
ErrorCode);
```

```
if (ErrorCode !=0)
```

```
{
```

```
    // flip through the CustFields and verify ["error_no"]
```

```
}
```

(VB)

Dim oDS as DataSet

oDS = GetWDMSList(1000)

Dim custBatchUpdate As New EnergyAPI.cCustBatch()

Dim CustFields As New DCEnergyAPI.CustomerMaint.DSCustomerBatchModify()

Dim locationType As Short = 3

For Each oRow In oDS.Tables("WDMS").Rows

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("location_num"), locationType,
 "product", "6")

 CustFields.Tables(0).Rows.Add(oRow("account_num"), oRow("location_num"), locationType,
 "tax_1", "N")

Next

Dim createLog as Boolean = True

Dim companyID as Short = 2

CustFields=custBatchUpdate.BatchUpdateWDMS(companyID, "WebUser", CustFields, createLog,
ErrorCode)

If (ErrorCode <> 0) then

 ‘flip through CustFields and examine [“error_no”]

End if

cCreditCard class

Overview

This section describes the methods that will enable the programmer to retrieve/modify customer credit card information. There are also methods that in conjunction with the cDelivery branch will allow the execution of the pre-authorization routine when logging phone orders.

NOTE: With Version 12.01.01+ If using a “CC Vault” provider you must make use of the customer vault id parameter(s) when required. The “GetCCMerchantIdAndVaultModeForCustomer/GetCCVaultModeForCCMerchant” method(s) can be used to determine if the customer’s CC information needs to be stored in the “vault” before saving it to the ADDs database via SmartConnect

- LoadCreditCardTypes(): Retrieves all valid credit card type IDs & descriptions.
- LoadSourcesOfCharge(): Retrieves all valid sources of charge IDs & their descriptions.
- VerifyCardNumber(): Performs inline credit card validation without interaction with the clearing house.
- ValidateCreditCard(): Performs credit card validation by interacting with the clearing house.
- LoadCCCCustomerInfo(): Retrieves credit card information for customer.
- LoadCCCCustomerSourcesOfCharge(): Retrieves all valid sources of charge for a customer’s credit card on file
- SaveCCCCustomerInfo(): Saves credit card information for customer.
- GetTransactionChargeAmount(): Retrieves additional card information and preauthorized amount required by PreAuthorizeCharge method.
- GetCCTransactionEligibility(): Will retrieve the default card for the customer/location/product and an indicator if a charge must be authorized prior to creating orders.
- GetCCMerchantIDForCustomer (): Returns the merchant ID that is associated with the customers division and optionally the credit card type.
- GetCCMerchantIDForCC(): retrieves the merchant id for a customer’s credit card
- PreAuthorizeCharge(): Preauthorize a charge to a credit card with the WSCCVerify web service and records it as a CC transaction.
- ChargeCreditCard(): Charges customer’s credit card for the amount specified and creates the necessary E3 transactions.
- EncryptCardNumber(): Encrypts a given unencrypted credit card number.
- GetCCMerchantIdAndVaultModeForCustomer(): Returns the CCMerchant ID that is associated with the customers division & the CC type. Also returns a flag indicating the CC merchant is setup to use a PCI vault
- GetCCVaultModeForCCMerchant(): returns a flag indicating if the CC merchant is setup to use a PCI vault

Vault

Vault is a method for storing credit card numbers implement by Network Merchant (NMI). You store the billing info and credit card number with them and they will return a token (vault id) representing the credit card for our mutual client.

You will need to program the getting of the vault id using the NMI api. More information can found at <https://secure.networkmerchants.com> you will need a username and password to access the site.

Using the SmartConnect API, you store the credit card info and vault id (not the credit card number) on our system. All transactions will then utilize this vault id. We will charge the customer's credit card using the vault id.

NOTE: if the E3 database is using the credit card module without the vault feature implemented, then you will have to use the encrypted card number field.

Please refer the next section Class CreditCardProcessData Definition for more information on these fields.

General steps to Adding/Modifying a credit card record when using the vault

- For an existing Credit Card the "customer_vault_id" is available in the "DSCCCustomerHeader" DataSet that is returned by calling the "LoadCCCustomerInfo" method
- Directly interact with the 3rd party vault provider to add/update the end user's credit card information
- Call SaveCCCustomerInfo & make sure to populate the "customer_vault_id" in the "DSCCCustomerHeader" DataSet if a new record is being added
- The "encrypted_cc_num" should not be used
- When using the "vault" the end user's credit card number should never be sent to SmartConnect under any circumstances.

When Authorizing/Charging/Voiding a credit card using the vault the only difference is that the "CustomerVaultId" needs to be populated instead of the "EncryptedCardNumber" in the "CreditCardProcessData" structure the rest of the functionality is identical. The "CustomerVaultId" is available in the same DataSet (LoadCCCustomerInfo) that contains all the other information for the customer's credit card

Data Classes and Datasets

Class CreditCardProcessData Definition:

Datatype/column name	Description
int _CustomerHeaderID	From DSCCCustomerHeaderInfo. cc_customer_header_id
int _CCTrxSourceCode	Value as returned by call to cCreditCard.LoadSourcesOfCharge. Must be 2 for phone orders
int _ARBatchNumber	0
int _ReferenceNumber	0
int _AdjustTrxHeaderID	0
int _TrxHeaderID	0
int _AccountNum	From DSCCCustomerHeaderInfo. account_num
short _CCTypeID	From DSCCCustomerHeaderInfo.cc_type_id
String _CardNumber	No longer supported as of 11.01.01+
String _EncryptedCardNumber	This is the encrypted credit card number version 11.01.01+ no longer supports use of the unencrypted credit card number
short _ExpirationMonth	From DSCCCustomerHeaderInfo. expiration_month
short _ExpirationYear	From DSCCCustomerHeaderInfo. expiration_year
Decimal _ChargeAmount	0
Decimal _OriginalChargeAmount	0
byte _WSCCVSuccess	1 indicates success
string _CardValidationID	This is the unique transaction id for the original transaction that needs to be Voided/Credited. This is an output parameter & should be left blank when making a call to the API.
string _PreAuthAuthorizationNumber	Will get updated after successful call to cCreditCard.PreAuthorizeCreditCard
int _PreAuthOldCCTVerified	Will get updated after successful call to cCreditCard.PreAuthorizeCreditCard
string _AuthorizationNumber	Unique transaction id returned by the gateway after a call to Validate, Pre-Authorize, Charge, Credit, Void.
int _OldCCTVerified	Will get updated after successful call to cCreditCard.PreAuthorizeCreditCard/ChargeCreditCard
string _HoldComments	Blank
short _HoldReason	Blank
string _WSCCVMessage1	UNUSED
string _WSCCVMessage2	UNUSED
string _WSCCVMessage3	UNUSED
string _WSCCVMessage4	UNUSED
string _WSCCVMessage5	UNUSED
string _WSCCVMessage6	UNUSED
string _WSErrorMsgText	The message that is returned from the processing gateway
string _TrxLMD	Blank

short _CurrentCCStatus	The status that is returned after making a call to PreAuthorizeCreditCard/ChargeCreditCard, Can be ignored on input parameter. 1=Pre-Authorized; 2=PaymentPending (Card has been charged but A/R has not been updated); 3=A/R has been updated; 4=Voided; 5=Held in E3; 6=Released a previously Held transaction in E3; 7=Credited; 8=Rejected; 9=CC Expired
string _CVVSecCode	Card Security Value or blank
byte _HistoryTransaction	0
int _CCCompanyID	Make a call to GetCCMerchantIDForCustomer/GetCCMerchantIDForCC
LocationAddress AddressOnCard	Class structure see below
string _CustomerVaultId	Unique identifier that links to the CC information stored in a CC Vault (Version 12.01.01+) When performing CC transactions this value should be populated with the value returned from the LoadCCCustomerInfo() "customer_value_id". NOTE: due to PCI compliance requirements, addition & modification of card data is not done via SmartConnect, it is the developers responsibility to directly interact with the vault provider to add/update information & then call SaveCCCustomerInfo() with the new/updated information to update the corresponding E3 record

Class AddressOnCard Definition:

Datatype/column name	Description
private string _title	Title or blank
private string _nameSuffix	Suffix or blank
private string _firstName	First name
private string _middleInitial	Middle initial or blank
private string _lastName	Last name
private string _street1	Street1
private string _street2	Street2 or blank
private string _street3	blank
private string _city	City
private string _state	State
private string _postalCode	Postal code
private short _country	0- USA, 1-Canada
private string _unparsedAddress	Blank
private decimal _latitude	Latitude or 0
private decimal _longitude	Longitude or 0

XML document:

The following is the xml document representation of the class object as defined above.

```
<?xml version="1.0" encoding="utf-8"?>
<CreditCardProcessData xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
<AddressOnCard>
  <FirstName>John</FirstName>
  <Title>Mr.</Title>
  <NameSuffix>phd</NameSuffix>
  <MiddleInitial> </MiddleInitial>
  <LastName>doe</LastName>
  <Street1>6 laurel drive</Street1>
  <Street2> </Street2>
  <Street3> </Street3>
  <City>Flanders</City>
  <State>NJ</State>
  <PostalCode>07836</PostalCode>
  <Country>0</Country>
  <UnparsedAddress>John Doe/6 Laurel Drive/</UnparsedAddress>
  <Latitude>0.0</Latitude>
  <Longitude>0.0</Longitude>
</AddressOnCard>
<CCCCompanyID>0</CCCCompanyID>
<CustomerHeaderID>0</CustomerHeaderID>
<CCTrxSourceCode>0</CCTrxSourceCode>
<ARBatchNumber>0</ARBatchNumber>
<ReferenceNumber>0</ReferenceNumber>
<AdjustTrxHeaderID>456</AdjustTrxHeaderID>
<TrxHeaderID>0</TrxHeaderID>
<AccountNum>2931</AccountNum>
<CCTypeID>0</CCTypeID>
<EncryptedCardNumber>EncryptedString</EncryptedCardNumber>
<ExpirationMonth>0</ExpirationMonth>
<ExpirationYear>0</ExpirationYear>
<ChargeAmount>0</ChargeAmount>
<WSCCVSuccess>0</WSCCVSuccess>
<HistoryTransaction>0</HistoryTransaction>
<CardValidationID></CardValidationID>
<PreAuthAuthorizationNumber> </PreAuthAuthorizationNumber>
<PreAuthOldCCTVerified>0</PreAuthOldCCTVerified>
<AuthorizationNumber>0</AuthorizationNumber>
<OldCCTVerified>0</OldCCTVerified>
<WSCCVMessage1> </WSCCVMessage1>
<WSCCVMessage2> </WSCCVMessage2>
<WSCCVMessage3> </WSCCVMessage3>
<WSCCVMessage4> </WSCCVMessage4>
<WSCCVMessage5> </WSCCVMessage5>
<WSCCVMessage6> </WSCCVMessage6>
<HoldReason>0</HoldReason>
<HoldComments> </HoldComments>
<WSErrorMsgText> </WSErrorMsgText>
<TrxLMD> </TrxLMD>
<CurrentCCStatus>0</CurrentCCStatus>
<CVVSecCode>0</CVVSecCode>
```

```
<CustomerVaultId>CCVaultUniqueld</CustomerVaultId>
</CreditCardProcessData>
```

DataSet DSCCCustomerHeader Definition:

Datatype/column name	Description
int cc_customer_header_id,	Unique id that represents the Credit Card record. Must be 0 when adding a credit card record
string cc_status,	'A'ctive, 'I'nactive, 'R'ejected
int account_num,	Valid account # (without the checkdigit)
short cc_type_id,	Valid credit card type as returned by cCreditCard.CreditCardTypes
int cc_company_id,	Unique id to identify merchant as returned by cCreditCard.GetCCMerchantIDForCustomer
string description,	Description of Credit Card
string first_name,	First name
string middle_initial,	Middle initial or blank
string last_name,	Last name
string street1,	Street1
string street2,	Street2 or blank
string city,	City
string state,	State
string postal_code,	Postal code
byte country,	0 for USA, 1 for Canada
string cc_number,	Encrypted credit card; will be returned with only the last 4 digits and rest showing as "*".
short cc_number_short,	Last 4 digits of the credit card
short length,	Full length of credit card
string decry_cc_num,	Decrypted Credit card <i>No longer supported as of 11.01.01+</i>
string encrypted_cc_num,	This is the encrypted credit card number version 11.01.01+ no longer supports use of the unencrypted credit card number. Use this field instead of cc_number when passing the encrypted CC number to the SaveCCCustomerInfo method
byte expiration_month,	Month credit card expires
short expiration_year,	Year credit card expires
decimal max_amount,	Maximum allowable amount to charge
string cc_validation_id,	Unique authorization number as returned by the call to cCreditCard.ValidateCard (CreditCardProcessData.AuthorizationNumber)
byte pc_selection,	Selection : 0 – posting group, 1- posting code
byte default_card,	<>0 indicates that card is the primary (default) card
string last_maintenance_dt,	Used for multi user validation. Blank when adding a new credit card.
string status	"A"ctive, "I"nactive
string customer_vault_id	Unique identifier that links to the CC information stored

	in a CC Vault (Version 12.01.01+)
boolean vault_change_made	This will always be false. The developer consuming the API is expected to perform all interaction with the Vault for adding/updating credit card information.

XML document:

The following is the xml document representation of type DSCCCustomerHeader. The example below shows 2 credit card records belonging to account 5 (without the checkdigit).

```
<DSCCCustomerHeader xmlns="http://www.addsys.com/DSCCCustomerHeader.xsd">
  <CCCustomerHeader>
    <cc_customer_header_id>596</cc_customer_header_id>
    <cc_status>A</cc_status>
    <account_num>5</account_num>
    <cc_type_id>1007</cc_type_id>
    <cc_company_id>74</cc_company_id>
    <description>Calvin1 - visa</description>
    <first_name>LORI</first_name>
    <middle_initial></middle_initial>
    <last_name>SMREKAR</last_name>
    <street1>P.O. BOX 9</street1>
    <street2></street2>
    <city>MT AUKUM</city>
    <state>CA</state>
    <postal_code>956840000</postal_code>
    <country>0</country>
    <cc_number>*****8291</cc_number>
    <cc_number_short>8291</cc_number_short>
    <length>16</length>
    <encrypted_cc_num>IB13nQTYKj5.....1akXvfAg</encrypted_cc_num>
    <expiration_month>3</expiration_month>
    <expiration_year>2016</expiration_year>
    <max_amount>0</max_amount>
    <cc_validation_id>2185926290</cc_validation_id>
    <pc_selection>1</pc_selection>
    <default_card>0</default_card>
    <last_maintenance_dt>Mar 19 2013 12:05:17:940PM</last_maintenance_dt>
    <status>A</status>
    <term_id>0</term_id>
    <budget_flag>0</budget_flag>
    <last_budget_charge_month>0</last_budget_charge_month>
    <last_budget_charge_year>0</last_budget_charge_year>
    <budget_charge_day>0</budget_charge_day>
    <statement_term_days>0</statement_term_days>
    <customer_vault_id"></customer_vault_id>
    <vault_change_made>>false</vault_change_made>
  </CCCustomerHeader>
</CCCustomerHeader>
  <cc_customer_header_id>94</cc_customer_header_id>
```

```

<cc_status>I</cc_status>
<account_num>5</account_num>
<cc_type_id>1007</cc_type_id>
<cc_company_id>0</cc_company_id>
<description>VISA</description>
<first_name>LORI</first_name>
<middle_initial></middle_initial>
<last_name>SMREKAR</last_name>
<street1>P.O. BOX 9</street1>
<street2></street2>
<city>MT AUKUM</city>
<state>CA</state>
<postal_code>956840000</postal_code>
<country>0</country>
<cc_number>*****0027</cc_number>
<cc_number_short>27</cc_number_short>
<length>13</length>
<encrypted_cc_num>zJilWb+.....+lFaQ=</encrypted_cc_num>
<expiration_month>8</expiration_month>
<expiration_year>2017</expiration_year>
<max_amount>0</max_amount>
<cc_validation_id>507513108</cc_validation_id>
<pc_selection>1</pc_selection>
<default_card>0</default_card>
<last_maintenance_dt>Mar 11 2013 3:48:30:923PM</last_maintenance_dt>
<status>A</status>
<term_id>0</term_id>
<budget_flag>0</budget_flag>
<last_budget_charge_month>0</last_budget_charge_month>
<last_budget_charge_year>0</last_budget_charge_year>
<budget_charge_day>0</budget_charge_day>
<statement_term_days>0</statement_term_days>
<customer_vault_id></customer_vault_id>
<vault_change_made>false</vault_change_made>
</CCCustomerHeader>
</DSCCCustomerHeader>

```

Please note: when adding a new card '**encrypted_cc_num**' must contain the encrypted credit card number unless you are using vault.

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

XML document:

The following is the xml document representation of the Dataset object as defined above. The example below shows 2 records for sources of charges.

```
<DSNumCodeDescription xmlns="http://www.addsys.com/DSNumCodeDescription.xsd">
  <NumCodeDescription>
    <code>1</code>
    <description>Description 1</description>
    <status>A</status>
    <selected>>false</selected>
  </NumCodeDescription>
  <NumCodeDescription>
    <code>2</code>
    <description>Description 2</description>
    <status>A</status>
    <selected>>false</selected>
  </NumCodeDescription>
</DSNumCodeDescription>
```


cCreditCard.LoadCreditCardTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve all valid credit card type IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCE3Common.DSCCVCreditCardTypes LoadCreditCardTypes(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCE3Common.DSCCVCreditCardTypes			
		Typed dataset	See below for details

DataSet DSCCVCreditCardTypes Defn:

Datatype/column name	Description
int code	Unique id to identify record
string description	Description attached to the card
string status	'A'ctive, 'I'nactive
bool selected	N/A – leave as false
string prefix	Comma delimited string of valid card prefixes
string length	Comma delimited string on valid card length

Return code	Description
96	Unknown Error Occurred.

XML document:

The following is the xml document representation of the object DSCCVCreditCardTypes. Below is an example of 2 credit card type records defined in the database

```
<DSCCVCreditCardTypes xmlns="http://www.addsys.com/DSCCVCreditCardTypes.xsd">
  <DSCCVCreditCardTypes>
    <code>2</code>
    <description>AMEX</description>
    <status>A</status>
    <selected>>false</selected>
    <prefix>34,37</prefix>
    <length>15</length>
  </DSCCVCreditCardTypes>
  <DSCCVCreditCardTypes>
    <code>3</code>
    <description>Visa</description>
    <status>A</status>
    <selected>>false</selected>
    <prefix>4</prefix>
    <length>13,16</length>
  </DSCCVCreditCardTypes>
</DSCCVCreditCardTypes>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int errorCode =0;
//
System.Data.DataSet
ccTypes=cCreditCard.LoadCreditCardTypes(companyID, sysUser, ref errorCode);
//
If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.LoadSourcesOfCharge()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be used to load all valid Sources of Charges & descriptions defined for the database.

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadSourcesOfCharge(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		System.Data.DataSet	See definition above for details

Return code	Description
96	Unknown Error Occurred.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadSourcesOfCharge>
      <add:pCompanyID>1101</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadSourcesOfCharge>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int errorCode =0;
//
System.Data.DataSet
sourcesOfCharge=cCreditCard.LoadSourcesOfCharge(companyID,sysUser);
```

cCreditCard.VerifyCardNumber()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to verify the credit card. It should always be called before ***ValidateCreditCard***. This method will perform some offline validation without interaction with merchant (clearing house). Should not be called when using “The Vault” to store CC information as all validation is performed by the developer consuming the SmartConnect API

Method Signature:

public int VerifyCardNumber(**short** pCompanyID, **string** pSysUser, **short** pCardType, **string** pEncryptedCardNumber)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCardType	Short Integer	Credit card type; DSCCCustomerHeader.cc_type_id
	pEncryptedCardNumber	string	Credit card number for customer (encrypted)
Return Type: int			
		Long integer	0 = success, <>0 = error #

Return code	Description
96	Unknown Error Occurred.
865	You may not enter a date prior to current date for the Credit Card expiration date. Please reenter.

Please note

pCardType is from the LoadCCCustomerInfo method.

XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:VerifyCardNumber>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCardType>1007</add:pCardType>
```

```
<add:pEncryptedCardNumber>IB13nQT.....akXvfAg</add:pEncryptedCardNumber>
</add:VerifyCardNumber>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
short cardType =1; // from DSCCCustomerHeader.cc_type_id
string cardNumber="encrypted string"; // from DSCCCustomerHeader.encrypted_cc_num
//
int errorCode
=custCreditCard.VerifyCardNumber(companyID,sysUser,cardType,pEncryptedCardNumber);

if (errorCode==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.ValidateCreditCard()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to validate the credit card against the merchant (clearing house). It should only be called after a successful call to **VerifyCardNumber**. When using “The Vault” to store CC information the card needs to be saved to the vault before using this method as SmartConnect/E3 does not have access to the card number & will using the “CustomerVaultID” to perform any transactions

Enterprise Signature:

```
public CreditCardProcessData ValidateCreditCard(short pCompanyID, string pSysUser,
CreditCardProcessData pCCData, out int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCData	Object of type CreditCardProcessData	See below for details
Return Values			
	pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: DCE3Common.CreditCardProcessData			
		Custom data structure	See definition above

Return code	Description
96	Unknown Error Occurred.
565	This data has been changed by another user, please refresh the data and try again.
788	There is a problem inserting data into a table. The process did not complete. Call your System Administrator.

Please note

CCTypeID is from the LoadCCCustomerInfo method.

CardValidationID is blank.

Authorization number is the unique number returned by the gateway.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
```

```
<soap:Header/>
<soap:Body>
  <add:ValidateCreditCard>
    <add:pCompanyID>1121</add:pCompanyID>
    <add:pSysUser>WebUser</add:pSysUser>
    <add:pCCData>
      <add:CustomerVaultId></add:CustomerVaultId>
      <add:EncryptedCreditCardInfo></add:EncryptedCreditCardInfo>
      <add:DisplayedCardNum></add:DisplayedCardNum>
      <add:CardNumShort>8291</add:CardNumShort>
      <add:CardNumLength>16</add:CardNumLength>
      <add:AddressOnCard>
        <add:FirstName>LORI</add:FirstName>
        <add>Title></add>Title>
        <add:NameSuffix></add:NameSuffix>
        <add:MiddleInitial></add:MiddleInitial>
        <add:LastName>SMREKAR</add:LastName>
        <add:Street1>P.O. BOX 9</add:Street1>
        <add:Street2></add:Street2>
        <add:Street3></add:Street3>
        <add:City>MT AUKUM</add:City>
        <add:State>CA</add:State>
        <add:PostalCode>956840000</add:PostalCode>
        <add:Country>0</add:Country>
        <add:UnparsedAddress></add:UnparsedAddress>
        <add:Latitude>0</add:Latitude>
        <add:Longitude>0</add:Longitude>
        <add:Telephone></add:Telephone>
        <add:FormattedAddress></add:FormattedAddress>
      </add:AddressOnCard>
      <add:CCCCompanyID>74</add:CCCCompanyID>
      <add:CustomerHeaderID>596</add:CustomerHeaderID>
      <add:CCTrxSourceCode>0</add:CCTrxSourceCode>
      <add:ARBatchNumber>0</add:ARBatchNumber>
      <add:ReferenceNumber>1234567890</add:ReferenceNumber>
      <add:AdjustTrxHeaderID>0</add:AdjustTrxHeaderID>
      <add:TrxHeaderID>0</add:TrxHeaderID>
      <add:AccountNum>5</add:AccountNum>
      <add:CCTypeID>1007</add:CCTypeID>
      <add:EncryptedCardNumber>0j1K.....U1fAg</add:EncryptedCardNumber>
      <add:ExpirationMonth>3</add:ExpirationMonth>
      <add:ExpirationYear>2016</add:ExpirationYear>
      <add:ChargeAmount>1.03</add:ChargeAmount>
      <add:WSCCVSuccess>0</add:WSCCVSuccess>
      <add:HistoryTransaction>0</add:HistoryTransaction>
      <add:CardValidationID></add:CardValidationID>
      <add:PreAuthAuthorizationNumber></add:PreAuthAuthorizationNumber>
      <add:PreAuthOldCCTVerified>0</add:PreAuthOldCCTVerified>
```

```

    <add:AuthorizationNumber>0</add:AuthorizationNumber>
    <add:OldCCTVerified>0</add:OldCCTVerified>
    <add:WSCCVMessage1></add:WSCCVMessage1>
    <add:WSCCVMessage2></add:WSCCVMessage2>
    <add:WSCCVMessage3></add:WSCCVMessage3>
    <add:WSCCVMessage4></add:WSCCVMessage4>
    <add:WSCCVMessage5></add:WSCCVMessage5>
    <add:WSCCVMessage6></add:WSCCVMessage6>
    <add:HoldReason>0</add:HoldReason>
    <add:HoldComments></add:HoldComments>
    <add:WSErrorMsgText></add:WSErrorMsgText>
    <add:TrxLMD></add:TrxLMD>
    <add:CurrentCCStatus>1</add:CurrentCCStatus>
    <add:CVVSecCode></add:CVVSecCode>
  </add:pCCData>
  <add:pErrorCode>0</add:pErrorCode>
</add:ValidateCreditCard>
</soap:Body>
</soap:Envelope>

```

.NET Example

```

EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int errorCode =0;
//
CreditCardProcessData creditCardData = new CreditCardProcessData();
//
// populate xml document
...
//
creditCardData = cCreditCard.ValidateCreditCard
(companyID,sysUser,creditCardData,ref errorCode);
//
If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}

```


cCreditCard.LoadCCCustomerInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve all Credit Card Related Account Information. The method will return all records (eg: if customer has 2 credit cards on record, 2 rows of data will be returned).

Method Signature:

public DSCCCustomerHeader LoadCCCustomerInfo(**short** pCompanyID, **string** pSysUser, **int** pAccountNum)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Long Integer	Account # without check digit
Return Type: DCElectronic.DSCCCustomerHeader			
		System.Data.DataSet	See definition above

Return code	Description
96	Unknown Error Occurred.

Please note

The results from this method are needed in other credit card calls.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCCCustomerInfo>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>5</add:pAccountNum>
    </add:LoadCCCustomerInfo>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
```

```
//
Short companyID = 2;
String sysUser ='webuser';
Int accountNum =1234;
Int[] sourceOfCharge=null;
Int errorCode=0;

System.Data.DatSet customerCCInfo = custCreditCard.LoadCCCcustomerInfo
(companyID,sysUser,accountNum, ref sourceOfCharges, ref errorCode);

If (errorCode ==0){
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.LoadCCCustomerSourcesOfCharge()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be used to load all Sources of Charge defined for the selected credit card.

Method Signature:

public List<int> LoadCCCustomerSourcesOfCharge(**short** pCompanyID, **string** pSysUser, **int** pCCCustomerHeaderID)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCCustomerHeaderID	Long Integer	CC Customer Header ID
Return Type: List<int>			
		List<int>	

Please note

pCCCustomerHeaderID is from the LoadCCCustomerInfo method.

XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCCCustomerSourcesOfCharge>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCCustomerHeaderID>90</add:pCCCustomerHeaderID>
    </add:LoadCCCustomerSourcesOfCharge>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int errorCode =0;
int CCCustomerHeaderID =1234;
//
```

```
Int[] sourcesOfCharge=cCreditCard.LoadCCCustomerSourcesOfCharge(companyID,sysUser,  
CCCustomerHeaderID);
```

cCreditCard.SaveCCCustomerInfo()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to save all Credit Card Related Account Information. The method will handle one record at a time. If customer has 2 credit card related records to save, the method must be executed 2 times. The encrypted card number and the vault id cannot be changed once the card has been added.

Method Signature:

public DSCCustomerHeader SaveCCCustomerInfo(**short** pCompanyID, **string** pSysUser, **DSCCustomerHeader** pCCHHeader, **List<int>** pSourcesOfCharges, **out int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCHHeader	DataSet	Object of type DSCCustomerHeader
	pSourcesOfCharges	List<int>	List a sources of charge for selected credit card. Use "LoadSourcesOfCharge" to get alist of valid values, 0 (Manual Charge) is the default.
Return Values			
	pErrorCode (ByRef)	Long integer	0 – if Successful<> 0 Error Number
Return Type: DCElectronic.DSCCustomerHeader			
	System.DataSet		Object of type DSCCustomerHeader

(Note: method expects one row, if more than one row is passed only the first row will be saved)

Return code	Description
96	Unknown Error Occurred.
788	There is a problem inserting data into a table. The process did not complete. Call your System Administrator.
526	The status cannot be changed at this time because this entry has related records that exist.
565	This data has been changed by another user, please refresh the data and try again.
228	Invalid customer vault Id.
226	Invalid last 4 digits for Credit Card
113	Invalid expiration month
114	Invalid expiration year
227	Invalid number of digits in Credit Card number

Please note

The encrypted card number and the vault id cannot be changes once the card the has been added.

You must have the `<CCCustomerHeader></CCCustomerHeader>` tags

Use the Authorization number returned from ValidateCreditCard for CC_Validation_ID.

You must populate the `<add:encrypted_cc_num></add: encrypted_cc_num>` field with encrypted credit card number if you are adding a card (non vault).

For version 12.01.01+ if you are using the vault feature, you must obtain a valid vault id from the cc processing merchant and set the value for “customer_vault_id”. Refer to your vault provider’s documentation for more information on how to save CC information to the vault (ADD Systems/SmartConnect cannot handle any CC infromation when using the vault). You must also populate the last 4 digits of the card used (convert to int16), expiration month , year & the number of digits in the credit card number. The “GetCCMerchantIdAndVaultModeForCustomer/

GetCCVaultModeForCCMerchant” method(s) should be used to determine if the customer’s CC information needs to be stored in the “vault” before saving it to the ADDs database via SmartConnect.

Please refer to ‘General steps to accepting a payment’.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveCCCustomerInfo>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCHeader>
        <CCCustomerHeader>
          <cc_customer_header_id>596</cc_customer_header_id>
          <cc_status>A</cc_status>
          <account_num>5</account_num>
          <cc_type_id>1007</cc_type_id>
          <cc_company_id>74</cc_company_id>
          <description>Calvin1 - visa</description>
          <first_name>LORI</first_name>
          <middle_initial></middle_initial>
          <last_name>SMREKAR</last_name>
          <street1>P.O. BOX 9</street1>
          <street2></street2>
          <city>MT AUKUM</city>
          <state>CA</state>
          <postal_code>956840000</postal_code>
```

```

    <country>0</country>
    <cc_number>*****8291</cc_number>
    <cc_number_short>8291</cc_number_short>
    <length>16</length>
    <encrypted_cc_num>IB13nQTYKj.....vfAg</encrypted_cc_num>
    <expiration_month>3</expiration_month>
    <expiration_year>2016</expiration_year>
    <max_amount>0</max_amount>
    <cc_validation_id>2100000</cc_validation_id>
    <pc_selection>1</pc_selection>
    <default_card>0</default_card>
    <last_maintenance_dt></last_maintenance_dt>
    <status>A</status>
    <term_id>0</term_id>
    <budget_flag>0</budget_flag>
    <last_budget_charge_month>0</last_budget_charge_month>
    <last_budget_charge_year>0</last_budget_charge_year>
    <budget_charge_day>0</budget_charge_day>
    <statement_term_days>0</statement_term_days>
    <customer_vault_id></customer_vault_id>
    <vault_change_made>>false</vault_change_made>
  </CCCustomerHeader>
</add:pCCHHeader>
<add:pSourcesOfCharges>
  <add:int>0</add:int>
</add:pSourcesOfCharges>
<add:pErrorCode>0</add:pErrorCode>
</add:SaveCCCustomeInfo>
</soap:Body>
</soap:Envelope>

```

.NET Example

```

EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
Int errorCode =0;
Int[1] sourcesOfCharges;

sourcesOfCharges[0]=2; // delivery phone order

System.Data.DataSet customerCCInfo =new System.Data.DataSet;
//
// populate customerCCInfo
...

```

```
//
customerCCInfo = custCreditCard.SaveCCCcustomerInfo(companyID,sysUser,customerCCInfo,
sourcesOfCharges ,ref errorCode);
//
If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```


cCreditCard.GetTransactionChargeAmount

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will retrieve the default card for a customer/location/product and an indicator if a charge must be authorized prior to creating orders... Location / Posting code will narrow the search if non zero will also Return the amount to preauthorize if applicable (calculated charge amount...)

Method Signature:

```
public int GetTransactionChargeAmount(short pCompanyID, string pSysUser, int pAccountNum, DCE3Common.E3Enumerations.CustomerLocationTypes pLocationType, short pLocationNum, decimal pDeliveryAmount, DateTime pDeliveryDate, short pCallReason, int pDeliveryUnits, out decimal pChargeAmount)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Long integer	Account# without checkdigit
	pLocationType	Short	0=None, 1=Delivery, 2=Service, 3=WDMS
	pLocationNum	Short	Location #
	pDeliveryAmount	Decimal	If non-zero the charge amount will not be calculated based on the units; any units passed in will be ignored
	pDeliveryDate	DateTime	Date delivery is scheduled for (required for calc purposes if amount = 0...)
	pCallReason	Short	The Call reason being scheduled -- used to get the amount
	pDeliveryUnits	decimal	Units will be used instead of tank size or ideal
Return Values			
	errorCode)	Long integer	Error Code 0 = all is OK; non-zero value indicates an error
	pChargeAmount(ref)	Decimal	the amount that needs to be pre-authorized for this delivery

Please note

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
```

```

<soap:Body>
  <add:GetTransactionChargeAmount>
    <add:pCompanyID>1121</add:pCompanyID>
    <add:pSysUser>WebUser</add:pSysUser>
    <add:pAccountNum>4580</add:pAccountNum>
    <add:pLocationNum>1</add:pLocationNum>
    <add:pDeliveryAmount>34</add:pDeliveryAmount>
    <add:pDeliveryDate>2013-04-07</add:pDeliveryDate>
    <add:pCallReason>6</add:pCallReason>
    <add:pDeliveryUnits>100</add:pDeliveryUnits>
    <add:pChargeAmount>47.00</add:pChargeAmount>
  </add:GetTransactionChargeAmount>
</soap:Body>
</soap:Envelope>e

```

.NET Example

```

EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
    int accountNum =1234;
    int errorCode =0;
short locationNum =1;
decimal deliveryAmount=156.79;
Decimal deliveryUnits=4000;
int source =2; // 2 – delivery phone order, 3 – service work orders, etc...
datetime deliveryDate="05/25/09";
short callReason =0; // always 0 for logging phone order.
decimal authorizedAmt =0;
//
Call custCreditCard.GetTransactionChargeAmount (companyID, sysUser, accountNum,
locationType, locationNum, deliveryAmount, deliveryDate, callReason, deliveryUnits, out
authorizedAmount);
//
If (errorCode ==0)
{
    // success: do something (call PreAuthorizeCreditCard for the authorizedAmount)...
}
Else {
    // failure: display message associated with error code.
}

```

cCreditCard.GetCCTransactionEligibility

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will retrieve the default card for a customer/location/product and an indicator if a charge must be authorized prior to creating orders...Location / Posting code will narrow the search if non zero. It will also return the amount to preauthorize if applicable (calculated charge amount...)

Method Signature:

```
public int GetCCTransactionEligibility(short pCompanyID, string pSysUser, int
pAccountNum,DCE3Common.E3Enumerations.CustomerLocationTypes pLocationType, short
pLocationNum, int pCCSourceCode, short pPostingCode,out int pCCCustomerHdrId, out bool
pMustCharge)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Long integer	Account# without checkdigit
	pLocationType	Short	0=None, 1=Delivery, 2=Service, 3=WDMS
	pLocationNum	Short	Location #
	pCCSourceCode	Long Integer	Source as selected from LoadSourcesOfCharge
	pPostingCode	Short	Product on delivery location or 0
Return Values			
	pCCCustomerHdrId(ref)	Long integer	Unique Customer Credit Card Header ID
	pMustCharge(ref)	boolean	Y/N Flag

Please note

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCCTransactionEligibility>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>4580</add:pAccountNum>
      <add:pLocationNum>1</add:pLocationNum>
      <add:pCCSourceCode>2</add:pCCSourceCode>
```

```
<add:pPostingCode>1</add:pPostingCode>
<add:pCCCustomerHdrId>0</add:pCCCustomerHdrId>
<add:pMustCharge>true</add:pMustCharge>
</add:GetCCTransactionEligibility>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int accountNum =1234;
int errorCode =0;
short locationNum =1;
short locationType=1;
short postingCode=2;
int source =2; // 2 – delivery phone order, 3 – service work orders, etc...
datetime deliveryDate="05/25/09";
short callReason =0; // always 0 for logging phone order.
int ccHeaderId =0;
bool mustCharge =false;
//
Call custCreditCard.GetCCTransactionEligibility
(companyID,sysUser,accountNum,locationType,locationNum,source,postingCode,ref
ccHeaderId,ref mustCharge);
//
If (erroCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.GetCCMerchantIDForCustomer()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve the unique merchant id for the customer/credit card type.

Method Signature:

public int GetCCMerchantIDForCustomer(**short** pCompanyID, **string** pSysUser, **int** pAccountNum, **short** pCCTypeID, **ref int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Long integer	Account# without checkdigit
	pCCTypeID	Short Integer	Credit Card Type; DSCCCustomerHeader.cc_type_id
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number

Return code	Description
96	Unknown Error Occurred.

Please note

pCCTypeID is from the LoadCCCustomerInfo method.

XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCCMerchantIDForCustomer>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>5</add:pAccountNum>
      <add:pCCTypeID>1007</add:pCCTypeID>
      <add:pErrorCode>0</add:pErrorCode>
    </add:GetCCMerchantIDForCustomer>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
```

```
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int accountNum =1234;
short ccTypeId =1;
//
int errorCode =0;
//
int merchantID
=custCreditCard.GetCCMerchantIDForCustomer(companyID,sysUser,accountNum,ccTypeId,ref
errorCode);

If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.GetCCMerchantIDForCC()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve the unique merchant id for the credit card in question.

Method Signature:

public int GetCCMerchantIDForCC(**short** pCompanyID, **string** pSysUser, **int** pCCCHeaderID, **out int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCCHeaderID	Long integer	Credit Card customer header id
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful <> 0 Error Number

Please note

pCCCHeaderID is from the LoadCCCHeaderInfo method.

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCCMerchantIDForCC>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCCHeaderID>590</add:pCCCHeaderID>
      <add:pErrorCode>0</add:pErrorCode>
    </add:GetCCMerchantIDForCC>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
int accountNum =1234;
```

```
short ccTypeId =1;
//
int errorCode =0;
int CCHeaderID =1234;
//
int merchantID =custCreditCard.GetCCMerchantIDForCC(companyID,sysUser,CCHeaderID,ref
errorCode);

If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```


cCreditCard.PreAuthorizeCreditCard()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called before cDelivery.LogPhoneOrder to preauthorize the charge on the customer credit card.

Method Signature:

public CreditCardProcessData PreAuthorizeCreditCard(**short** pCompanyID, **string** pSysUser, **CreditCardProcessData** pCreditCardData, **out int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCData	Object of type CreditCardProcessData	Refer to explanation of structure in Data Class & dataset definition
Return Values: CreditCardProcessData			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
	CCData	CreditCardProcessData	Updated data structure

Return code	Description
96	Unknown Error Occurred.
565	This data has been changed by another user, please refresh the data and try again.

Please note:

CCCompanyID is the Merchant ID
CardValidationID is blank.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:PreAuthorizeCreditCard>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCData>
        <add:CustomerVaultId></add:CustomerVaultId>
        <add:EncryptedCreditCardInfo></add:EncryptedCreditCardInfo>
        <add:DisplayedCardNum></add:DisplayedCardNum>
        <add:CardNumShort>8291</add:CardNumShort>
      </add:pCCData>
    </add:PreAuthorizeCreditCard>
  </soap:Body>
</soap:Envelope>
```

```
<add:CardNumLength>16</add:CardNumLength>
<add:AddressOnCard>
  <add:FirstName>LORI</add:FirstName>
  <add:Title></add:Title>
  <add:NameSuffix></add:NameSuffix>
  <add:MiddleInitial></add:MiddleInitial>
  <add:LastName>SMREKAR</add:LastName>
  <add:Street1>P.O. BOX 9</add:Street1>
  <add:Street2></add:Street2>
  <add:Street3></add:Street3>
  <add:City>MT AUKUM</add:City>
  <add:State>CA</add:State>
  <add:PostalCode>956840000</add:PostalCode>
  <add:Country>0</add:Country>
  <add:UnparsedAddress></add:UnparsedAddress>
  <add:Latitude>0</add:Latitude>
  <add:Longitude>0</add:Longitude>
  <add:Telephone></add:Telephone>
  <add:FormattedAddress></add:FormattedAddress>
</add:AddressOnCard>
<add:CCCCompanyID>74</add:CCCCompanyID>
<add:CustomerHeaderID>596</add:CustomerHeaderID>
<add:CCTrxSourceCode>0</add:CCTrxSourceCode>
<add:ARBatchNumber>0</add:ARBatchNumber>
<add:ReferenceNumber>1234567890</add:ReferenceNumber>
<add:AdjustTrxHeaderID>0</add:AdjustTrxHeaderID>
<add:TrxHeaderID>0</add:TrxHeaderID>
<add:AccountNum>5</add:AccountNum>
<add:CCTypeID>1007</add:CCTypeID>
<add:EncryptedCardNumber>IB13nQ.....6BU1akXvfAg</add:EncryptedCardNumber>
<add:ExpirationMonth>3</add:ExpirationMonth>
<add:ExpirationYear>2016</add:ExpirationYear>
<add:ChargeAmount>1.03</add:ChargeAmount>
<add:WSCCVSuccess>0</add:WSCCVSuccess>
<add:HistoryTransaction>0</add:HistoryTransaction>
<add:CardValidationID></add:CardValidationID>
<add:PreAuthAuthorizationNumber></add:PreAuthAuthorizationNumber>
<add:PreAuthOldCCTVerified>0</add:PreAuthOldCCTVerified>
<add:AuthorizationNumber>0</add:AuthorizationNumber>
<add:OldCCTVerified>0</add:OldCCTVerified>
<add:WSCCVMessage1></add:WSCCVMessage1>
<add:WSCCVMessage2></add:WSCCVMessage2>
<add:WSCCVMessage3></add:WSCCVMessage3>
<add:WSCCVMessage4></add:WSCCVMessage4>
<add:WSCCVMessage5></add:WSCCVMessage5>
<add:WSCCVMessage6></add:WSCCVMessage6>
<add:HoldReason>0</add:HoldReason>
<add:HoldComments></add:HoldComments>
```

```
<add:WSErrorMsgText></add:WSErrorMsgText>
<add:TrxLMD></add:TrxLMD>
<add:CurrentCCStatus>1</add:CurrentCCStatus>
<add:CVVSecCode></add:CVVSecCode>
</add:pCCData>
<add:pErrorCode>0</add:pErrorCode>
</add:PreAuthorizeCreditCard>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
//
int errorCode =0;
//
System.Xml.XmlDocument creditCardData = new System.Xml.XmlDocument ();
//
// populate xml document
...
//
creditCardData=custCreditCard.PreAuthorizeCharge
(companyID, sysUser, creditCardData, ref errorCode);
//
If (errorCode ==0 && creditCardData.WSCCVSuccess == 1)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.ChargeCreditCard()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

Charges customer's credit card for the amount specified and creates the necessary E3 transactions.

Method Signature:

public DCE3Common.CreditCard.CreditCardProcessData ChargeCreditCard(**short** pCompanyID, **string** pSysUser, DCE3Common.CreditCard.CreditCardProcessData pCCData, **out int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCData	Object of type CreditCardProcessData	Refer to explanation of structure above.
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
	CCData	CreditCardProcessData	Updated data structure

Please note

CCCompanyID is the Merchant ID
CardValidationID is blank.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ChargeCreditCard>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCData>
        <add:CustomerVaultId></add:CustomerVaultId>
        <add:EncryptedCreditCardInfo></add:EncryptedCreditCardInfo>
        <add:DisplayedCardNum></add:DisplayedCardNum>
        <add:CardNumShort>8291</add:CardNumShort>
        <add:CardNumLength>16</add:CardNumLength>
        <add:AddressOnCard>
          <add:FirstName>LORI</add:FirstName>
          <add>Title></add>Title>
          <add>NameSuffix></add>NameSuffix>
        </add:AddressOnCard>
      </add:pCCData>
    </add:ChargeCreditCard>
  </soap:Body>
</soap:Envelope>
```

```
<add:MiddleInitial></add:MiddleInitial>
<add:LastName>SMREKAR</add:LastName>
<add:Street1>P.O. BOX 9</add:Street1>
<add:Street2></add:Street2>
<add:Street3></add:Street3>
<add:City>MT AUKUM</add:City>
<add:State>CA</add:State>
<add:PostalCode>956840000</add:PostalCode>
<add:Country>0</add:Country>
<add:UnparsedAddress></add:UnparsedAddress>
<add:Latitude>0</add:Latitude>
<add:Longitude>0</add:Longitude>
<add:Telephone></add:Telephone>
<add:FormattedAddress></add:FormattedAddress>
</add:AddressOnCard>
<add:CCCCompanyID>74</add:CCCCompanyID>
<add:CustomerHeaderID>596</add:CustomerHeaderID>
<add:CCTrxSourceCode>0</add:CCTrxSourceCode>
<add:ARBatchNumber>0</add:ARBatchNumber>
<add:ReferenceNumber>1234567890</add:ReferenceNumber>
<add:AdjustTrxHeaderID>0</add:AdjustTrxHeaderID>
<add:TrxHeaderID>0</add:TrxHeaderID>
<add:AccountNum>5</add:AccountNum>
<add:CCTypeID>1007</add:CCTypeID>
<add:EncryptedCardNumber>lB13nQTYKj.....akXvfAg</add:EncryptedCardNumber>
<add:ExpirationMonth>3</add:ExpirationMonth>
<add:ExpirationYear>2016</add:ExpirationYear>
<add:ChargeAmount>1.03</add:ChargeAmount>
<add:WSCCVSuccess>0</add:WSCCVSuccess>
<add:HistoryTransaction>0</add:HistoryTransaction>
<add:CardValidationID></add:CardValidationID>
<add:PreAuthAuthorizationNumber></add:PreAuthAuthorizationNumber>
<add:PreAuthOIDCCTVerified>0</add:PreAuthOIDCCTVerified>
<add:AuthorizationNumber>0</add:AuthorizationNumber>
<add:OIDCCTVerified>0</add:OIDCCTVerified>
<add:WSCCVMessage1></add:WSCCVMessage1>
<add:WSCCVMessage2></add:WSCCVMessage2>
<add:WSCCVMessage3></add:WSCCVMessage3>
<add:WSCCVMessage4></add:WSCCVMessage4>
<add:WSCCVMessage5></add:WSCCVMessage5>
<add:WSCCVMessage6></add:WSCCVMessage6>
<add:HoldReason>0</add:HoldReason>
<add:HoldComments></add:HoldComments>
<add:WSErrorMsgText></add:WSErrorMsgText>
<add:TrxLMD></add:TrxLMD>
<add:CurrentCCStatus>1</add:CurrentCCStatus>
<add:CVVSecCode></add:CVVSecCode>
</add:pCCData>
```

```
<add:pErrorCode>0</add:pErrorCode>
</add:ChargeCreditCard>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
//
int errorCode =0;
//
System.Xml.XmlDocument creditCardData = new System.Xml.XmlDocument ();
//
// populate xml document
...
//
creditCardData=custCreditCard.ChargeCreditCard (companyID,sysUser,creditCardData,ref
errorCode);
//
If (errorCode ==0 && creditCardData.WSCCVSuccess == 1)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.VoidCreditCard()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

Voids customer's credit card for the amount specified and creates/updates the necessary E3 transactions.

Method Signature:

public DCE3Common.CreditCard.CreditCardProcessData VoidCreditCard(**short** pCompanyID, **string** pSysUser, DCE3Common.CreditCard.CreditCardProcessData pCCData, **out int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCData	Object of type CreditCardProcessData	Refer to explanation of structure above.
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number
	CCData	CreditCardProcessData	Updated data structure

Please note

CCCompanyID is the Merchant ID

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:VoidCreditCard>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCCData>
        <add:CustomerVaultId></add:CustomerVaultId>
        <add:EncryptedCreditCardInfo></add:EncryptedCreditCardInfo>
        <add:DisplayedCardNum></add:DisplayedCardNum>
        <add:CardNumShort>8291</add:CardNumShort>
        <add:CardNumLength>16</add:CardNumLength>
        <add:AddressOnCard>
          <add:FirstName>LORI</add:FirstName>
          <add>Title></add>Title>
          <add:NameSuffix></add:NameSuffix>
          <add:MiddleInitial></add:MiddleInitial>
        </add:AddressOnCard>
      </add:pCCData>
    </add:VoidCreditCard>
  </soap:Body>
</soap:Envelope>
```

<add:LastName>SMREKAR</add:LastName>
<add:Street1>P.O. BOX 9</add:Street1>
<add:Street2></add:Street2>
<add:Street3></add:Street3>
<add:City>MT AUKUM</add:City>
<add:State>CA</add:State>
<add:PostalCode>956840000</add:PostalCode>
<add:Country>0</add:Country>
<add:UnparsedAddress></add:UnparsedAddress>
<add:Latitude>0</add:Latitude>
<add:Longitude>0</add:Longitude>
<add:Telephone></add:Telephone>
<add:FormattedAddress></add:FormattedAddress>
</add:AddressOnCard>
<add:CCCCompanyID>74</add:CCCCompanyID>
<add:CustomerHeaderID>596</add:CustomerHeaderID>
<add:CCTrxSourceCode>0</add:CCTrxSourceCode>
<add:ARBatchNumber>0</add:ARBatchNumber>
<add:ReferenceNumber>1234567890</add:ReferenceNumber>
<add:AdjustTrxHeaderID>0</add:AdjustTrxHeaderID>
<add:TrxHeaderID>0</add:TrxHeaderID>
<add:AccountNum>5</add:AccountNum>
<add:CCTypeID>1007</add:CCTypeID>
<add:EncryptedCardNumber>dadkjasdfG47d.....1akXvfAg</add:EncryptedCardNumber>
<add:ExpirationMonth>3</add:ExpirationMonth>
<add:ExpirationYear>2016</add:ExpirationYear>
<add:ChargeAmount>1.03</add:ChargeAmount>
<add:WSCCVSuccess>0</add:WSCCVSuccess>
<add:HistoryTransaction>0</add:HistoryTransaction>
<add:CardValidationID>IdReturnedFromTheTrxToVoid</add:CardValidationID>
<add:PreAuthAuthorizationNumber></add:PreAuthAuthorizationNumber>
<add:PreAuthOldCCTVerified>0</add:PreAuthOldCCTVerified>
<add:AuthorizationNumber></add:AuthorizationNumber>
<add:OldCCTVerified>0</add:OldCCTVerified>
<add:WSCCVMessage1></add:WSCCVMessage1>
<add:WSCCVMessage2></add:WSCCVMessage2>
<add:WSCCVMessage3></add:WSCCVMessage3>
<add:WSCCVMessage4></add:WSCCVMessage4>
<add:WSCCVMessage5></add:WSCCVMessage5>
<add:WSCCVMessage6></add:WSCCVMessage6>
<add:HoldReason>0</add:HoldReason>
<add:HoldComments></add:HoldComments>
<add:WSErrorMsgText></add:WSErrorMsgText>
<add:TrxLMD>Mar 21 2013 3:45:24:403PM</add:TrxLMD>
<add:CurrentCCStatus>3</add:CurrentCCStatus>
<add:CVVSecCode></add:CVVSecCode>
</add:pCCData>
<add:pErrorCode>0</add:pErrorCode>


```
</add:VoidCreditCard>
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
//
int errorCode =0;
//
CreditCardProcessData creditCardData = new CreditCardProcessData();
//
// populate xml document
...
//
creditCardData=custCreditCard.ChargeCreditCard (companyID,sysUser,creditCardData,ref
errorCode);
//
If (errorCode ==0 && creditCardData.WSCCVSuccess == 1)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard.EncryptCardNumber()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to encrypt a credit card. It only needs to be called when using a new credit card for all other instances the API(s) will provided the existing credit card number in encrypted format.

Method Signature:

public string EncryptCardNumber(**short** pCompanyID, **string** pSysUser, **string** pDecryptedCardNumber)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDecryptedCardNumber	String	Credit card number for customer (unencrypted)
Return Type: string			
		String	Encrypted credit card number

Return code	Description
Unhandled Exception	The method throw an exception if it encounters a situation where it cannot successfully encrypt the card number

Please note

pCCCustomerHeaderID is from the LoadCCCustomerInfo method.

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:EncryptCardNumber>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pDecryptedCardNumber>0000000000000000</add:pDecryptedCardNumber>
    </add:EncryptCardNumber>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser ='webuser';
string cardNumber="123456789098765";
string encryptedCardNumber =custCreditCard. EncryptCardNumber (companyID, sysUser,
cardNumber);

if (!string.IsNullOrEmpty(encryptedCardNumber) && encryptedCardNumber.Length > 0)
{
    // success: use this to add the new card
}
```

cCreditCard. GetCCMerchantIdAndVaultModeForCustomer()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve the unique merchant id and a flag indicating if this CC merchant is setup to use a PCI vault to store CC information.

Method Signature:

public int GetCCMerchantIdAndVaultModeForCustomer(**short** pCompanyID, **string** pSysUser, **int** pAccountNum, **short** pCCTypeID, **out int** pCCMerchantId, **out bool** pVaultMode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Long integer	Account# without checkdigit
	pCCTypeID	Short Integer	Credit Card Type; DSCCCustomerHeader.cc_type_id
Output	pCCMerchantId	int (32 bit integer)	Unique CC merchant ID used to perform CC processing
Output	pVaultMode	Boolean	Returns a flag indicating weather this CC merchant is setup to use a PCI vault
Return Values			
	pErrorCode (ref)	Long integer	0 – if Successful<> 0 Error Number

Return code	Description
96	Unknown Error Occurred.
122	Invalid Input Parameters

Please note

pCCTypeID is from the LoadCCCustomerInfo method.

XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCCMerchantIdAndVaultModeForCustomer>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pCCTypeID>1007</add:pCCTypeID>
    </add:GetCCMerchantIdAndVaultModeForCustomer>
```

```
</soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
Short companyID = 2;
String sysUser = "WebUser";
int accountNum =25;
short ccTypeId =1007;
int ccMerchantID = 0;
bool vaultMode = false;
//
int errorCode =0;
//
errorCode =custCreditCard.GetCCMerchantIdAndVaultModeForCustomer
(companyID,sysUser,accountNum,ccTypeId, out ccMerchantId, out vaultMode);

If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

cCreditCard. GetCCVaultModeForCCMerchant()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Payment & Credit Card Entry Business Edition**

Description

This method will be called to retrieve a flag indicating if the CC merchant is setup to use a PCI vault to store CC information.

Method Signature:

public int GetCCVaultModeForCCMerchant (**short** pCompanyID, **string** pSysUser, **int** pCCMerchantId)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCCMerchantId	int (32 bit integer)	Unique CC merchant ID used to perform CC processing
Return Values			
	pVaultMode	Long integer	Returns a flag indicating weather this CC merchant is setup to use a PCI vault

Return code	Description
122	Invalid Input Parameters

XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetCCVaultModeForCCMerchant>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>Webuser</add:pSysUser>
      <add:pCCMerchantID>79</add:pCCMerchantID>
    </add:GetCCVaultModeForCCMerchant>
  </soap:Body>
</soap:Envelope>
```

.NET Example

```
EnergyAPIWebService.cCreditCard custCreditCard = new
EnergyAPIWebService.cCreditCard();
custCreditCard.Url="http://some address" + "cCreditCard.asmx";
//
```

```
Short companyID = 2;
String sysUser = "WebUser";
int ccMerchantID = 79;
bool vaultMode = false;
//
vaultMode = custCreditCard.GetCCVaultModeForCCMerchant(companyID, sysUser,
accountNum, ccTypeId, ccMerchantId);

If (errorCode ==0)
{
    // success: do something...
}
Else {
    // failure: display message associated with error code.
}
```

General steps to accepting a payment:

- i) Load Customer information (cCreditCard.LoadCCCustomerInfo)

Existing Credit Card on file.

- 1) Get Merchant ID (cCreditCard.GetCCMerchantIDForCC)
- 2) Choose one of the following.
 - a. Preauthorize Amt (cCreditCard.PreAuthorizeCreditCard)
 - b. Charge Amt (cCreditCard.ChargeCreditCard)
 - c. Void Amt (cCreditCard.VoidCreditCard)

New Credit Card Number (not on file) Non Vault

- 1) Load CC types (cCreditCard.LoadCreditCardTypes)
- 2) Verify CC number – (must be encrypted first) offline (cCreditCard.VerifyCardNumber)
- 3) Get Merchant ID (cCreditCard.GetCCMerchantIDForCustomer)
- 4) Validate CC number (cCreditCard.ValidateCreditCard)
- 5) Save the new CC info to file (cCreditCard.SaveCCCustomerInfo)
- 6) Choose one of the following.
 - a. Preauthorize Amt (cCreditCard.PreAuthorizeCreditCard)
 - b. Charge Amt (cCreditCard.ChargeCreditCard)
 - c. Void Amt (cCreditCard.VoidCreditCard)

New Credit Card Number (not on file) Vault For Version 12.01.01+

- 1) Get Vault id from card processor
- 2) Load CC types (cCreditCard.LoadCreditCardTypes)
- 3) Get Merchant ID (cCreditCard.GetCCMerchantIDForCustomer)
- 4) Save the new CC info to file (cCreditCard.SaveCCCustomerInfo)
- 5) Choose one of the following.
 - a. Preauthorize Amt (cCreditCard.PreAuthorizeCreditCard)
 - b. Charge Amt (cCreditCard.ChargeCreditCard)
 - c. Void Amt (cCreditCard.VoidCreditCard)

Note : You must set the CCCompanyID with the GetCCMerchantID before you charge the credit card. If you don't, you will get an error.

Was the transaction successful?

When we validate, pre-authorize, charge, or void a charge, we are interacting with another system., the credit card processor. We will have to look at additional fields to determine if the transaction was successful or not

The first check is the Error Code field. If this is zero (0) then the request is good. Otherwise there is something wrong with the request.

Next, you must check the WSCCVSuccess to see if CCVerify web service successfully completed. (0 = not successful, 1= successful).

Then you look at the CurrentCCStatus to see if the transaction was accepted by the clearing house. Here are the list of status and descriptions.

- 1-Pre-authorized
- 2-Charge not submitted
- 3-Charge submitted (AR posting only)
- 4-Voided
- 5-Held
- 6-Released
- 7-Credited
- 8-Rejected
- 9-Pre-authorization expired

Finally, if it was not accepted, check the WSErrorMsgText field for the error description as returned by the clearing house.

General steps to logging a phone order with CC:

- 1) First you should do a Load Customer Delivery Information (**cDelivery.GetDeliveryGeneralInfo**). The response will contain a CC Customer header ID.
- 2) Check for non-zero CC Customer header ID to see if there is a credit card for this account. If CC Customer header ID is zero then do something else (*).
- 3) Next, we should see if the transaction is eligible to be charged with a transaction Eligibility (**cCreditCard.GetCCTransactionEligibility**), the response will contain a CC Customer header ID and a Must Charge.
 - a. Check for non zero CC Customer header ID. If CC Customer header ID is zero then do something else (*). This means that this is not eligible to be charged.
 - b. Check for Must Charge
 - c. If true: **NOTE: you cannot use the “DeliveryOrder.ChargeOnPosting” flag if “MustCharge” is true you must pre-authorize prior to logging the order.**
 - i. Call **cCreditCard.GetTransactionChargeAmount** (... , out decimal pChargeAmount)
 - ii. The charge amount MUST be preauthorized before logging the phone order **need cc customer header id and cc transaction id arguments passed to the **cDelivery.LogPhoneOrder_V2**
 - iii. Call **cCreditCard.PreAuthorizeCreditCard()**
 - iv. If Successful
 1. Call **cDelivery.LogPhoneOrder_V2()**
 - d. If False, you may ask the customer to pre-authorized the change amount or charge the card when the delivery is posted. **NOTE: you can either pre-authorize or charge on posting not both.**
 - i. If you have decided to charge the CC automatically when the delivery is posted
 1. Call **cDelivery.LogPhoneOrder_V2()** & make sure the “**CCCustomerHdrId**” is set correctly (value from step #3) & set “**ChargeOnPosting**” to true in the “**DeliveryOrder**” data structure
 - ii. If the end user chooses to preauthorize
 1. Call **cCreditCard.GetTransactionChargeAmount** (... , out decimal pChargeAmount)
 2. Preauthorize the charge before logging the phone order **need cc customer header id and cc transaction id arguments passed to the **cDelivery.LogPhoneOrder_V2**
 3. Call **cCreditCard.PreAuthorizeCreditCard()**
 4. If Successful
 - a. Call **cDelivery.LogPhoneOrder_V2()** the **CCCustomerHeaderId**, **CCTrxHdrId**, **AuthorizationAmount** must be set
 - iii. If the end user chooses NOT to preauthorize or you have decided not to automatically charge the CC when delivery is posted
 1. Log phone order (**cDelivery.LogPhoneOrder_V2**) without setting the **CCCustomerHeaderID** or **ChargeOnPosting** values

(*) something else is choice that you and your client will have to decide on. There are possibilities like display a message and/or Log phone order (**cDelivery.LogPhoneOrder_V2**).

General Credit Card Method Return Codes

0	No Error
974	The number of digits is not valid for the selected card type.
975	The range is not valid for the selected card type.
976	The card number is not valid for the selected card type.

If Return Code = 0 check the WSCCVSuccess field to check if the transaction was accepted by the clearing house

Check WSErrorMsgText for the error description as returned by the clearing house

cEFT class

Overview

This section describes the methods that will enable the programmer to retrieve/modify customer EFT information.

- LoadAccountTypes(): Retrieves all valid account type IDs and their descriptions.
- LoadTermCode(): Retrieves all valid term code IDs & their descriptions.
- LoadWeekendDrafting(): Retrieves all valid weekend drafting IDs and their descriptions.
- LoadCreditLineDrafting(): Retrieves all valid credit line IDs and their descriptions.
- LoadPostingGroup(): Retrieves all valid posting group IDs and their descriptions.
- GetEFTCustomerPostingGroups(): Retrieves all posting groups for a specific existing EFT customer record.
- GetEFTCustomerNotifications(): Retrieves all notification methods for a specific existing EFT customer record.
- GetEFTCustomerLocations(): Retrieves all locations eligible for EFT for a specific existing EFT customer record.
- GetEFTCustomerDetails(): Retrieves all EFT related customer records for a given customer.
- SaveEFTCustomerDetails(): Called to save one existing EFT customer record or add a new EFT customer record.
- TeleDraft_AccountQuery(): If you have an account with Teledraft, this will allow you to use the AccountQuery call via SmartConnect. This is a test of a bank account / routing number – see Teledraft Documentation for detailed information.

cEFT.LoadAccountTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all valid account type IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadAccountTypes(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		Typed dataset	See below for details

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

cEFT.LoadTermCode()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all valid term code IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadTermCode(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		Typed dataset	See below for details

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

cEFT.LoadWeekendDrafting()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all valid weekend drafting IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadWeekendDrafting(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		Typed dataset	See below for details

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

cEFT.LoadCreditLineDrafting()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all valid credit line drafting IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadCreditLineDrafting(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		Typed dataset	See below for details

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

cEFT.LoadPostingGroup()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all valid posting group IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

public DCAddCommon.DSNumCodeDescription LoadPostingGroup(**short** pCompanyID, **string** pSysUser)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCAddCommon.DSNumCodeDescription			
		Typed dataset	See below for details

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A'ctive,'I'nactive
bool selected	N/A – leave as false

cEFT.GetEFTCustomerDetails()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all EFT related customer records for a specific customer.

Method Signature:

public DCEnergyAPI.EFT.DSEFTCustomerDetails GetEFTCustomerDetails(**short** pCompanyID, **int** pAccountNum, **string** pSysUser, **ref int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Long integer	Energy acct# without checkdigit
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Long Integer	Error return code or 0
Return Type: DCEnergyAPI.EFT.DSEFTCustomerDetails			
		Typed dataset	See below for details

Datatype/column name	Description
string status	'A'ctive, 'I'nactive
int bank_id	Unique ID for Financial institution
string bank_routing_no	Bank routing number
string bank_account_no	Bank account number
string bank_authorized	Yes, No, & Pending
bool send_authorization	Send Authorization Flag
short bank_account_type_id	Unique ID for back account type
bool credit_balance_drafting	Draft Credit Balnsce Flag
bool credit_process_separate	Process credits separately flag
short credit_line_drafting	Credit Line Draft
decimal maximum_draft_amount	Maximum draft amount
short weekend_drafting	Weekend drafting
int reference_no	Reference number
date start_date	Star Date
date end_date	End Date
bool email_notify	Notify - Email Flag
bool buypass_notify	Notify - Buypass Flag
bool fax_notify	Notify - Fax Flag
bool fax_on_peak_hours	Fax on peak hours flag
string fax_time	Fax time
bool easylink_notify	Notify - easylink Flag
bool web_notify	Notify - Web Flag
bool dtn_notify	Notify - dtn Flag
bool paper_notify	Notify - paper Flag
short pre_defined_comment	Pre-defined comment id number

string comment	Comment
string last_maintenance_dt	Last mainenance date
int eft_customer_hdr_id	Eft customer header id
bool budget_flag	Budget flag
short term_id	Term code id
string manual_draft_only	Manual draft flag
short last_budget_draft_month	Last budget draft month
short last_budget_draft_year	Last budget draft year
string encr_bank_account_no	For internal use only

cEFT.GetEFTCustomerLocations()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all locations eligible for EFT for a specific existing EFT customer record.

Method Signature:

public DCEnergyAPI.EFT.DSEFTCustomerLocation GetEFTCustomerLocations(**short** pCompanyID, **int** pEFTHdrId, **string** pSysUser, **ref int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pEFTHdrId	Long integer	Unique id for customer EFT record
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Long Integer	Error return code or 0
Return Type: DCEnergyAPI.EFT.DSEFTCustomerLocations			
		Typed dataset	See below for details

Datatype/column name	Description
int location_type	0-no locations (trxs without location), 1-tanks,2-service,3-wdms
short location_num	Location number
string name	Location description name

Note: the “name” field is a display only field and is not required when saving location(s).

cEFT.GetEFTCustomerNotifications()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all notifications types for a specific existing EFT customer record.

Method Signature:

public DCEnergyAPI.EFT.DSEFTCustomerNotification GetEFTCustomerNotifications(**short** pCompanyID, **int** pEFTHdrId, **string** pSysUser, **ref int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pEFTHdrId	Long integer	Unique id for customer EFT record
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Long Integer	Error return code or 0
Return Type: DCEnergyAPI.EFT.DSEFTCustomerNotification			
		Typed dataset	See below for details

Datatype/column name	Description
short notification_type	1-paper,2-fax,3-email,4-easylink,5-dtn,6-buypass,7-web
string notification_value	1,2,3,4,5,6,7

cEFT.GetEFTCustomerPostingGroups()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to retrieve all posting groups for a specific existing EFT customer record.

Method Signature:

public DCEnergyAPI.EFT.DSEFTCustomerPostingGrp GetEFTCustomerPostingGroups(**short** pCompanyID, **int** pEFTHdrId, **string** pSysUser, **ref int** pErrorCode)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pEFTHdrId	Long integer	Unique id for customer EFT record
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Long Integer	Error return code or 0
Return Type: DCEnergyAPI.EFT.DSEFTCustomerPostingGrp			
		Typed dataset	See below for details

Datatype/column name	Description
short posting_group_id	Unique id identifying posting group
string description	Posting group description

cEFT.SaveEFTCustomerDetails()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method will be called to save one existing EFT customer record or add a new EFT customer record.

Method Signature:

```
public DCEnergyAPI.EFT.DSEFTCustomerDetails SaveEFTCustomerDetails(short pCompanyID,int
pAccountNum, string pSysUser, short pIncTrxsWithoutLoc,
DCEnergyAPI.EFT.DSEFTCustomerDetails pEFTCustomerDtls,
DCEnergyAPI.EFT.DSEFTCustomerLocation pLocations,
DCEnergyAPI.EFT.DSEFTCustomerNotification pNotifications,
DCEnergyAPI.EFT.DSEFTCustomerPostingGrp pPostingGroups, ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pAccountNum	Long integer	Energy acct# without checkdigit
	pSysUser	String	Calling program (IVRUser/WebUser)
	pIncTrxsWithoutLoc	short	Include transactions without location? 0 no, 1 yes
	pEFTCustomerDtls	Dataset	DSEFTCustomerDetails
	pLocations	Dataset	DSEFTCustomerLocation
	pNotifications	Dataset	DSEFTCustomerNotification
	pPostingGroups	DataSet	DSEFTCustomerPostingGrp
	pErrorCode	Long Integer	Error return code or 0
Return Type: DCEnergyAPI.EFT.DSEFTCustomerDetails			
		Typed dataset	Refer to cEFT.GetEFTCustomerDetails

Note: the “name” field in the DSEFTCustomerLocation should be left blank. This field is used by the GetCustomerEFTLocations to populate the name of the location (or blank if there isn’t any).

XML Example

The following example shows how to save an EFT record for one customer:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveEFTCustomerDetails>
      <add:pCompanyID>1121</add:pCompanyID>
      <add:pAccountNum>151</add:pAccountNum>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pIncTrxsWithoutLoc>1</add:pIncTrxsWithoutLoc>
      <add:pEFTCustomerDtls>
        <EFTCustomerDetail>
          <status>A</status>
          <bank_id>2</bank_id>
          <bank_routing_no>11</bank_routing_no>
          <bank_account_no>320060038793</bank_account_no>
          <bank_authorized>Y</bank_authorized>
          <send_authorization>>false</send_authorization>
          <bank_account_type_id>2</bank_account_type_id>
          <credit_balance_drafting>>false</credit_balance_drafting>
          <credit_process_separate>>false</credit_process_separate>
          <credit_line_drafting>0</credit_line_drafting>
          <maximum_draft_amount>0.00</maximum_draft_amount>
          <weekend_drafting>2</weekend_drafting>
          <reference_no>0</reference_no>
          <start_date>2016-09-01</start_date>
          <end_date>2049-12-01</end_date>
          <email_notify>>true</email_notify>
          <buypass_notify>>false</buypass_notify>
          <fax_notify>>false</fax_notify>
          <fax_on_peak_hours>>false</fax_on_peak_hours>
          <fax_time>0</fax_time>
          <easylink_notify>>false</easylink_notify>
          <web_notify>>false</web_notify>
          <dtm_notify>>false</dtm_notify>
          <paper_notify>>false</paper_notify>
          <pre_defined_comment>0</pre_defined_comment>
          <comment/>
          <last_maintenance_dt>2016-06-28</last_maintenance_dt>
          <eft_customer_hdr_id>0</eft_customer_hdr_id>
          <budget_flag>>false</budget_flag>
          <term_id>0</term_id>
          <manual_draft_only>Y</manual_draft_only>
          <last_budget_draft_month>0</last_budget_draft_month>
```

```
<last_budget_draft_year>0</last_budget_draft_year>
<encr_bank_account_no/>
<budget_draft_day>0</budget_draft_day>
```

```
<institution_code>0</institution_code>
```

```
</EFTCustomerDetail>
</add:pEFTCustomerDtls>
<add:pLocations>
  <EFTLocation>
    <location_type>0</location_type>
    <location_num>0</location_num>
    <name>blah</name>
  </EFTLocation>
</add:pLocations>
<add:pNotifications>
  <EftNotification>
    <notification_type>0</notification_type>
    <notification_value>0</notification_value>
    <cust_contact_id>0</cust_contact_id>
  </EftNotification>
</add:pNotifications>
<add:pPostingGroups>
  <EFTCustomerPostingGrp>
    <posting_group_id>0</posting_group_id>
    <description>desc1</description>
  </EFTCustomerPostingGrp>
</add:pPostingGroups>
<add:pErrorCode>0</add:pErrorCode>
</add:SaveEFTCustomerDetails>
</soap:Body>
</soap:Envelope>
```

cEFT.TeleDraft_AccountQuery()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API EFT Maintenance Business Edition**

Description

This method is a wrapper to the Teledraft account query call – it requires a valid account with Teledraft and configuration of the Energyapi2.ini file with the account information. It echoes back the results from the Teledraft AccountQuery service. Please refer to the Teledraft documentation for more detailed information.

Method Signature:

public string Teledraft_AccountQuery(**short** pCompanyID, **string** pSysUser, **string** pRoutingNumber, **string** pBankAccountNumber, **Decimal** pAmount, **out string** pReturnDescription)

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pRoutingNumber	String	Routing Number – 9 digits a must
	pBankAccountNumber	String	Bank Account Number
	pAmount	Decimal	If this amount is 0 or less, it will be set to 1. Teledraft requirement
	pReturnDescription	String {OUT}	Description matching the return code
Return Type: String – Return Code From Teledraft			
The return code from Teledraft query – may include other codes, but PASS/FAIL/UNKNOWN mean that the call went through with valid info.		000 Success PASS – FAIL – UNKNOWN - 995 Invalid Request 996 User Error 997 No Information Available 998 Credentials Exception 999 System Error	

cCustomerDocument class

Overview

The cCustomerDocument class provides methods to load Document types and descriptions and to load documents for a specified customer.

This section describes the methods that will enable the programmer to retrieve/modify customer document information.

- LoadDocumentTypeDesc (): Loads Document Types and their corresponding descriptions.
- LoadCustomerDocuments (): Loads documents for the specified customer and criteria -- transaction and document type OR date range and document type.
- GetDocumentImage (): Loads a pdf image of the specified document number.

Data Structures used:

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A' - Active or 'I' - Inactive
bool selected	N/A – leave as false

DataSet DSCustomerDocumentInfo Defn:

Datatype/column name	Description
short doc_type	Document Type – list from LoadDocumentTypeDesc
int doc_id	Identifies the document
datetime doc_date	Document date
int has_print_info	0 == no, 1 = yes
Short tank_num	Tank number
Short service_num	Service location number
Short location_num	WDMS location number
String description	Document type description

cCustomerDocument.LoadDocumentTypeDesc ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will be called to retrieve all document type IDs and descriptions as defined in the database (can be presented to end user for selection).

Method Signature:

```
public DCAddCommon.DSNumCodeDescription LoadDocumentTypeDesc (short
pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DSNumCodeDescription DataSet			

.NET Example

The following example shows how to retrieve a list of documents for a customer:

```
Dim companyId As Short 'set appropriate value
Dim data As DataSet
```

```
'Load document type descriptions into dataset
data = cCustomerDocument.LoadDocumentTypeDesc(companyId)
'Use dataset to display values
```

XML Example

The following example shows how to retrieve a list of document types:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDocumentTypeDesc>
      <add:pCompanyID>1122</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadDocumentTypeDesc>
  </soap:Body>
</soap:Envelope>
```

cCustomerDocument. LoadCustomerDocuments ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will be called to retrieve selected documents for a customer. It can be all documents related to a transaction (if pTrxUniqueKey parameter is non zero) or documents for a provided date range.

*Note that the cTransaction.GetARTransactions output now includes a has_documents indicator as well as the transaction identifier (trx_unique_key).

The GetDocumentImage() method can be used to pull the image of a specific document by passing the “doc_id” returned by this method.

Method Signature:

```
public DCEnergyAPI.CustomerDocument.DSCustomerDocumentInfo
LoadCustomerDocuments(short pCompanyID, string pSysUser, int pAccountNum, int
pTrxUniqueKey, DateTime pFromDate, DateTime pToDate, short pDocumentType)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	The account number to find documents for
	pTrxUniqueKey	Integer	Identifies which transaction to find the document for -- if this argument is non zero, the date arguments will not be considered
	pFromDate	DateTime	begin date of document search
	pToDate	DateTime	end date of document search
	pDocumentType	Short (16 bit integer)	the type of document to search for -- 0 = ALL document types
Return Type: DSCustomerDocumentInfo DataSet			

.NET Example

The following example shows how to retrieve a list of documents for a customer:

```
Dim companyId as Short 'set appropriate value
Dim fromDate As Date 'set appropriate value
Dim toDate As Date 'set appropriate value
Dim doctype As Short 'set appropriate value
Dim trxKey As Integer 'set appropriate value
Dim account As Integer 'set appropriate value
Dim tank As Short 'set appropriate value
Dim service As Short 'set appropriate value
Dim wdms As Short 'set appropriate value
Dim data As DataSet 'set appropriate value
```

'Load customer documents into dataset
data = cCustomerDocument.LoadCustomerDocuments(companyId, account, trxKey, fromDate, toDate,
doctype, tank, service, wdms)
'Use dataset to display values

XML Example

The following example shows how to retrieve a list of documents for a customer:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCustomerDocuments>
      <add:pCompanyID>1122</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>11</add:pAccountNum>
      <add:pTrxUniqueKey>0</add:pTrxUniqueKey>
      <add:pFromDate>2001-01-01</add:pFromDate>
      <add:pToDate>2040-12-31</add:pToDate>
      <add:pDocumentType>11</add:pDocumentType>
      <add:pTankNum>0</add:pTankNum>
      <add:pServiceNum>0</add:pServiceNum>
      <add:pLocationNum>0</add:pLocationNum>
    </add:LoadCustomerDocuments>
  </soap:Body>
</soap:Envelope>
```

cCustomerDocument.GetDocumentImage ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will be called to pull the image of a document from the E3 system. It creates the customer document as a PDF (Portable Document Format) file and returns it to the calling program as an array of bytes which can be encoded as a single reference or a multi-reference value. The rules for an array of bytes are similar to those for a string.

Please Note that for Pegasus Mobile documents, the file will be in HTML format.

```
<picture xsi:type="SOAP-ENC:base64">
  aG93IG5vDyBicm73biBjb3cNCg==
</picture>
```

Method Signature:

```
public byte[] GetDocumentImage(short pCompanyID, string pSysUser, int pDocID)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDocID	Integer	Document ID as returned by LoadCustomerDocuments() method
Return Type: byte[] Array of Bytes representing a data stream of the PDF document			

.NET Example

The following example shows how to retrieve a document image:

```
Dim companyId As Short 'set appropriate value
Dim userId As String   'set appropriate value
Dim docId As Integer   'set appropriate value
Dim document As Byte()
```

```
'Retrieve document
document = cCustomerDocument.GetDocumentImage(companyId, userId, docId)
```

```
'Save dialog
SaveFileDialog1.AddExtension = True
SaveFileDialog1.DefaultExt = ".PDF"
SaveFileDialog1.FileName = txtDocID.Text + ".PDF"
SaveFileDialog1.ShowDialog()
```

```
'Save document to file
System.IO.File.WriteAllBytes(SaveFileDialog1.FileName, document)
```


XML Example

The following example shows how to retrieve a document image:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetDocumentImage>
      <add:pCompanyID>1122</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pDocID>26249</add:pDocID>
    </add:GetDocumentImage>
  </soap:Body>
</soap:Envelope>
```

cCustomerDocument.AddOnDemandDocument()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method will be called to put the image of a document into the E3 system. The document is stored in the E3 database & is available for viewing from the E3 Customer Document Viewer.

Method Signature:

```
Public Function AddOnDemandDocument(ByVal pCompanyId As Short, ByVal pSysUser As String, ByVal pRequestData As DCEnergyAPI.CustomerDocument.AddOnDemandDocumentRequest) As DCEnergyAPI.CustomerDocument.AddOnDemandDocumentResponse
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pRequestData	AddOnDemandDocumentRequest	API Request information
Return Type: AddOnDemandDocumentResponse. API response information			

Class DCEnergyAPI.CustomerDocument.AddOnDemandDocumentRequest Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short TankNum	E3 tank number or 0 if doc is not a tank level document
short ServiceNum	E3 service number or 0 if doc is not a service level document
short LocationNum	E3 location number or 0 if doc is not a location level document
short DocumentType	The E3 document type, validated against the list of valid document types. Value should be set to 999
int DocumentReferenceNum	Document reference number
string DocumentFileExtension	The extension of the file including the “.dot” that is being saved
byte[] DocumentData	byte array containing the document blob
date DocumentDate	document date

Class DCEnergyAPI.CustomerDocument.AddOnDemandDocumentResponse Definition:

Datatype/column name	Description
int ErrCode	A non-zero error code if any

<code>int DocId</code>	The internal document id that will provide a pointer to the saved document

.NET Example

The following example shows how to retrieve a document image:

```
Dim pRequestData As DCEnergyAPI.CustomerDocument.AddOnDemandDocumentRequest = New DC
EnergyAPI.CustomerDocument.AddOnDemandDocumentRequest()
pRequestData.DocumentData = System.IO.File.ReadAllBytes(pFilePath)
pRequestData.AccountNum = 25
pRequestData.DocumentDate = DateTime.Now
pRequestData.DocumentFileExtension = System.IO.Path.GetExtension(pFilePath)
pRequestData.DocumentReferenceNum = pRequestData.DocumentData.Length
pRequestData.DocumentType = 999
pRequestData.TankNum = 1

Dim apiResponse As DCEnergyAPI.CustomerDocument.AddOnDemandDocumentResponse = AddOnD
emandDocument(2, "WebUser", pRequestData)

Return String.Format("ErrorCode = {0}; DocId = {1}", apiResponse.ErrCode, apiRespons
e.DocId)
```

XML Example

The following example shows how to retrieve a document image:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:TestMethod_AddOnDemandDocument>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pRequestData>
        <add:AccountNum>25</add:AccountNum>
        <add:TankNum>0</add:TankNum>
        <add:ServiceNum>0</add:ServiceNum>
        <add:LocationNum>1</add:LocationNum>
        <add:DocumentType>999</add:DocumentType>
        <add:DocumentReferenceNum>1234</add:DocumentReferenceNum>
        <add:DocumentFileExtension>.pdf</add:DocumentFileExtension>
        <add:DocumentData>RepresentationOfDocumentsDataBlob</add:DocumentData>
        <add:DocumentDate>2018-11-28</add:DocumentDate>
      </add:pRequestData>
    </add:TestMethod_AddOnDemandDocument>
  </soap:Body>
</soap:Envelope>
```

cContactInfo class

Overview

***** This class was introduced at release V10.01 (E3) and should be used instead of the contact info related services that exist in the cCustomerMaint class (eg: cLoadCustContactInfo and cSaveCustContactInfo). Therefore, cContactInfo must be used for databases whose E3 release is 10.x+, cCustomerMaint must be used for databases whose E3 release is < 10.x.**

The cContactInfo class provides methods to load Document types and descriptions and to load documents for a specified customer.

This section describes the methods that will enable the programmer to retrieve/modify customer document information.

- LoadContactTypeIDs (): Loads the system defined Contact Type IDs and their corresponding descriptions.
- LoadDocumentOutputFormats (): Loads the system defined Document Output Formats and their corresponding descriptions.
- LoadContactTypes (): Loads all of the defined contact types on the system.
- LoadContactInfo (): Loads the Contact Info for the specified customer.
- LoadLocations (): Loads the locations for the customer and indicate whether they are linked to the contact info record identified by pCustContactID
- LoadDocuments (): Loads the Documents for the customer and indicate whether they are linked to the contact info record identified by pCustContactID
- SaveContactInfo (): Saves any changes to the contact info and / or linked locations
- DeleteContactInfo (): Deletes the contact information linked to pCustContactID

Data Structures used:

DataSet DSNumCodeDescription Defn:

Datatype/column name	Description
int code	Unique id
string description	Description of item; used to display to end user
string status	'A' - Active or 'I' - Inactive
bool selected	N/A – leave as false

DataSet DSContactTypes Defn:

Datatype/column name	Description
short contact_type_id	Unique id for the contact type
string description	Description of item; used to display to end user
short type	The type (system defined) of contact: 1-Phone #, 2-Fax #, 3-E-mail. (List from LoadContactTypeIDs)
string status	'A' - Active or 'I' - Inactive
datetime last_maintenance_dt	Read Only – timestamp of last change to record

DataSet DSContactInfo Defn:

Datatype/column name	Description
int cust_contact_id	Unique id for the customer contact record
short contact_type_id	Identifies the contact type (Home Phone, Cel. Phone, ...)
short type	The type (system defined) of contact: 1-Phone #, 2-Fax #, 3-E-mail. (List from LoadContactTypeIDs)
string contact_value	The actual contact value phone number, email address, ...
string notes	Notes re this contact
byte notification	0 = no, 1 = yes
byte privacy	0 = no, 1 = yes
string row_status	'A' - Active or 'I' - Inactive
string last_maintenance_dt	Read Only – timestamp of last change to record
string last_maintenance_userid	Read Only – user who performed last change to record
int error_code	Stores any error code after a save (0 no error, <> 0 use cUtility.GetErrorDescription for error msg)

DataSet DSLinkedLoc Defn:

Datatype/column name	Description
int cust_contact_id	Identifies the contact record that the location list is for
byte location_type	1 = Tank, 2 = Service, 3 = WDMS
short location_num	The location number
string primary_flag	"0" = No, "1" = Yes
int selected	0 = no, 1 = yes – set to 1 to link location
string description	Read Only – the location address

DataSet DSContactInfoSummary Defn:

Datatype/column name	Description
string contact_value	The unformatted contact value (phone, email, ...)
short type	The type (system defined) of contact: 1-Phone #, 2-Fax #, 3-E-mail. (List from LoadContactTypeIDs)
short contact_type_id	Identifies the contact type (maintained values)– use list from DSContactTypes .
string primary_flag	"0" = No, "1" = Yes
byte location_type	1 = Tank, 2 = Service, 3 = WDMS
string location_num	The location number
string description	The location address
byte notification	0/1 No/Yes Is this contact set for notifications
byte privacy	0/1 No/Yes Is this contact set for privacy
string notes	Notes stored with the contact record

DataSet DSLinkedDoc Defn:

Datatype/column name	Description
int doc_type	Identifies the document
short output format	If the document is selected, this should be set – the valid output formats are obtained bu calling cContactInfo.LoadDocumentOutputFormats

byte suppress_print	If the document is selected for transmission, set to 1 for transmission only with no printed copy
int selected	0 = no, 1 = yes – set to 1 to link document for transmission
int doc_group	Identifies what location type the document is valid for 0 = Customer, 1 = Tank, 2 = Service, 3 = WDMS
string description	Describes the document eg WDMS invoice
int must_print	Unused

cContactInfo.LoadContactTypeIDs()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load the system defined contact type IDs and their corresponding descriptions.

Method Signature:

```
public DCAddCommon.DSNumCodeDescription LoadContactTypeIDs(short pCompanyID,
string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DSNumCodeDescription DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContactTypeIDs>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadContactTypeIDs>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list contact type ids in a grid.

```
protected void btnContactTypeIDs_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInfo = new EnergyAPI.cContactInfo();

    DCAddCommon.DSNumCodeDescription contactTypeIDS = new DCAddCommon.DSNumCodeDescription ();

    contactTypeIDS = contactInfo.LoadContactTypeIDs(2, "WebUser");

    gvMain.DataSource = contactTypeIDS;
    gvMain.DataBind ();
}
```

}

Result:

code	description	status	selected	display
1	Phone #	A	☐	1 - Phone #
2	Fax #	A	☐	2 - Fax #
3	E-mail	A	☐	3 - E-mail
4	DTN	A	☐	4 - DTN

cContactInfo.LoadContactTypes()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load all of the defined contact types on the system

Method Signature:

```
public DCE3Common.DSContactTypes LoadContactTypes(string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DSContactTypes DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContactTypes>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadContactTypes>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list contact types in a grid

```
protected void btnLoadContactTypes_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInfo = new EnergyAPI.cContactInfo();

    DCE3Common.DSContactTypes contactTypes = new DCE3Common.DSContactTypes ();

    contactTypes = contactInfo.LoadContactTypes("WebUser");

    gvMain.DataSource = contactTypes;
    gvMain.DataBind();

}
```

Result (partial list):

contact_type_id	description	type	status	last_maintenance_dt
0	Home Phone 1	1	A	12/22/2009 2:54:59 PM
1	Home Phone 2	1	A	06/12/2007 3:43:25 PM
2	Business Phone 1	1	A	06/12/2007 3:43:25 PM
3	Business Phone 2	1	A	12/22/2009 2:53:09 PM
4	Cell Phone 1	1	A	06/12/2007 3:43:25 PM
5	Cell Phone 2	1	A	12/22/2009 2:42:21 PM
9	Business Phone 3	1	I	12/22/2009 2:54:59 PM
10	Business Phone 4	1	A	12/22/2009 2:54:59 PM

cContactInfo.LoadContactInfo ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to load the Contact Info for the specified customer

Method Signature:

```
public DCCustomers.PushDocument.DSContactInfo LoadContactInfo(short pCompanyID, string pSysUser, int pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	string	Calling program (IVRUser/WebUser)
	pAccountNum	integer	Customer Number to load the contact info for
Return Type: DSContactInfo DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContactInfo>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>14</add:pAccountNum>
    </add:LoadContactInfo>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list contact information in a grid

```
protected void btnLoadContactInfo_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    DCCustomers.PushDocument.DSContactInfo = new DCCustomers.PushDocument.DSContactInfo ();

    //
    int AccountNum = 1;
    short Dbnum = 2;
    dsContactInfo = contactInformation.LoadContactInfo(Dbnum, "WebUser", AccountNum);
    gvMain.DataSource = dsContactInfo;
    gvMain.DataBind();
}
```

}

Result:

cust_contact_id	contact_type_id	type	contact_value	notes	notification	privacy	row_status	last_maintenance_dt	last_maintenance_userid	error_code
1	1	1	5147779999		0	0	A	May 4 2011 12:39:37:880PM	WebUser	0
6	0	1	5144289022		0	0	A	Apr 26 2011 11:50:42:266AM	chrisb	0
684	18	1	5551234567	L	0	0	A	May 4 2011 2:09:55:126PM	WebUser	0
3	8	2	9735555464		0	0	A	Jul 10 2009 9:16:12:063AM	braudie	0
2	6	3	braudie@addsys.com		3	0	A	Jan 8 2010 10:43:08:080AM	braudie	0
4	6	3	CHANCHOUROVA@ADDSYS.COM		0	0	A	Feb 6 2009 2:00:52:116PM	guelena44	0
5	7	3	dust@dust2.com		0	0	A	Apr 26 2011 11:51:24:216AM	chrisb	0
608	6	3	fomgc@addsys.com	Converted from EFT component	0	0	A	Jan 4 2011 9:32:22:013AM	INITIAL_LOAD	0
688	2	3	david.capparelli@noaddress2.com		0	1	A	May 4 2011 2:16:28:536PM	WebUser	0

cContactInfo.LoadLocations ()

- License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load all of the locations for the customer and indicate whether they are linked to the contact info record identified by pCustContactID

Method Signature:

```
public DCCustomers.PushDocument.DSLinkedLoc LoadLocations(short pCompanyID,
string pSysUser, int pAccountNum, int pCustContactID)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	string	Calling program (IVRUser/WebUser)
	pAccountNum	integer	Customer Number to load the contact info for
	pCustContactID	integer	Identifies the customer contact record
Return Type: DSLinkedLoc DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadLocations>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>14</add:pAccountNum>
      <add:pCustContactID>1191</add:pCustContactID>
    </add:LoadLocations>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list location(s) for given account/contact id in grid

```
protected void btnLoadLocations_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    DCCustomers.PushDocument.DSLinkedLoc contactLocs = new
DCCustomers.PushDocument.DSLinkedLoc ();
    short dbNum = 2;
    int accountNum = 1;
    int contactID = 4;
    contactLocs = contactInformation.LoadLocations(dbNum, "WebUser", accountNum, contactID);
    gvMain.DataSource = contactLocs;
```

```
gvMain.DataBind();
```

```
}
```

Result: (using account # and contact id from call to LoadContactInfo result)

cust_contact_id	location_type	location_num	primary_flag	selected	description
4	0	0	0	1	
4	1	2	0	0	
4	1	3	0	0	PHIL & DEBBIE HAMILTON/3703 CAMINO HILLS DRIVE/2220 SEVEN OAKS CT./EL DORADO HILLS, CA. 95630
4	1	4	0	0	PHIL & DEBBIE HAMILTON/3703 CAMINO HILLS DRIVE/2220 SEVEN OAKS CT./EL DORADO HILLS, CA. 95630
4	1	5	0	0	
4	1	6	0	0	
4	1	7	0	0	
4	1	8	0	0	
4	1	1	1	1	HAMILTON/6 LAUREL Dr./FLANDERS NJ 07836/extra line 1/ext line 2
4	2	1	0	0	SERVICE ADDRESS
4	2	2	0	0	
4	3	1	0	0	PHIL & DEBBIE HAMILTON / 3703 CAMINO HILLS DRIVE / 2220 SEVEN OAKS CT. / EL DORADO HILLS, CA. , 956300000

cContactInfo.SaveContactInfo ()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method will save any changes to the contact info and / or linked locations
Note that you can save only one contact info record at a time along with its linked location information – pass a pDSContactInfo dataset with only one record in it – the one that is to be saved.

Method Signature:

```
public DCCustomers.PushDocument.DSContactInfo SaveContactInfo(short
pCompanyID, string pSysUser, DCCustomers.PushDocument.DSContactInfo
pDSContactInfo, DCCustomers.PushDocument.DSLinkedLoc pDSLinkedLoc, int
pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSContactInfo	DSContactInfo	DataSet containing changed Contact Info values
	pDSLinkedLoc	DSLinkedLoc	DataSet containing changed or unchanged linked location values
	pAccountNum	integer	Customer Number linked to the contact info that is being saved
Return Type: DSContactInfo DataSet (updated values)			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveContactInfo>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pDSContactInfo>
        <CONTACT_INFO_HDR>
          <cust_contact_id>4503</cust_contact_id>
          <contact_type_id>11</contact_type_id>
          <type>3</type>
          <contact_value>fongs@addsys.com</contact_value>
          <notes></notes>
          <notification>0</notification>
          <privacy>0</privacy>
          <row_status>A</row_status>
          <last_maintenance_dt>Jul 25 2012 10:38:43:186AM</last_maintenance_dt>
        </CONTACT_INFO_HDR>
      </add:pDSContactInfo>
    </add:SaveContactInfo>
  </soap:Body>
</soap:Envelope>
```

```

        <last_maintenance_userid>WebUser</last_maintenance_userid>
        <error_code>0</error_code>
    </CONTACT_INFO_HDR>
</add:pDSContactInfo>
<add:pDSLlinkedLoc>
    <CONTACT_INFO_DTL>
        <cust_contact_id>4503</cust_contact_id>
        <location_type>0</location_type>
        <location_num>0</location_num>
        <primary_flag>1</primary_flag>
        <selected>1</selected>
        <description"&></description>
    </CONTACT_INFO_DTL>
    <CONTACT_INFO_DTL>
        <cust_contact_id>4503</cust_contact_id>
        <location_type>1</location_type>
        <location_num>1</location_num>
        <primary_flag>0</primary_flag>
        <selected>0</selected>
        <description>STEVEN W BERRIER/Mr. Jim Smith/./, 956670000</description>
    </CONTACT_INFO_DTL>
    <CONTACT_INFO_DTL>
        <cust_contact_id>4503</cust_contact_id>
        <location_type>2</location_type>
        <location_num>1</location_num>
        <primary_flag>0</primary_flag>
        <selected>0</selected>
        <description>INACTIVE SERV. ADDRESS</description>
    </CONTACT_INFO_DTL>
</add:pDSLlinkedLoc>
<add:pAccountNum>14</add:pAccountNum>
</add:SaveContactInfo>
</soap:Body>
</soap:Envelope>

```

.NET Example

Eg: Adding home phone and email address contact information.

```

protected void btnContactInfo_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    EnergyAPI.DSContactInfo dsContactInfo = new EnergyAPI.DSContactInfo ();
    EnergyAPI.DSLinkedLoc dsLocs = new EnergyAPI.DSLinkedLoc ();
    int AccountNum = 1;
    short Dbnum = 2;
    DataRow dsNewRow;
    int errorCode = 0;
    string Csysuser = "WebUser";

```



```
//creating new contact info for customer
dsContactInfo = contactInformation.LoadContactInfo(Dbnum , "WebUser", AccountNum);
//There are significant changes in 10.x You can only update one header record at a time therefore the
above call is made to get the DataSet/Schema. If data already exists we will
//clear all and simply retain the structure.
// ONCE ONLY
dsLocs = contactInformation.LoadLocations(Dbnum , "WebUser", AccountNum, 0); //ALWAYS send 0
string homePhone = "9871214545";
string email = "somename@noaddress.com";
if (homePhone.Length > 0)
{
    dsContactInfo.Clear();
    dsNewRow = dsContactInfo.Tables[0].NewRow();
    //update fields
    dsNewRow["cust_contact_id"] = 0;
    dsNewRow["contact_type_id"] = 0;
    dsNewRow["type"] = 1;
    dsNewRow["contact_value"] = homePhone;
    dsNewRow["notes"] = "";
    dsNewRow["notification"] = 0;
    dsNewRow["privacy"] = 1;
    dsNewRow["row_status"] = "A";
    dsNewRow["last_maintenance_dt"] = DateTime.Now.ToString();
    dsNewRow["last_maintenance_userid"] = Csysuser;
    dsNewRow["error_code"] = 0;

    dsContactInfo.Tables[0].Rows.Add(dsNewRow);
    dsContactInfo = contactInformation.SaveContactInfo(Dbnum, Csysuser, dsContactInfo, dsLocs,
AccountNum);
    // go thru the dataset and examine ["error_code"] to make sure save went thru
}
//add a email row
if (email.Length > 0)
{
    dsContactInfo.Clear(); // to remove prev
    dsNewRow = dsContactInfo.Tables[0].NewRow();
    //update fields
    dsNewRow["cust_contact_id"] = 0;
    dsNewRow["contact_type_id"] = 5;
    dsNewRow["type"] = 3;
    dsNewRow["contact_value"] = email;
    dsNewRow["notes"] = "";
    dsNewRow["notification"] = 0;
    dsNewRow["privacy"] = 1;
    dsNewRow["row_status"] = "A";
    dsNewRow["last_maintenance_dt"] = DateTime.Now.ToString();
    dsNewRow["last_maintenance_userid"] = Csysuser;
    dsNewRow["error_code"] = 0;

    dsContactInfo.Tables[0].Rows.Add(dsNewRow);
```

```
        dsContactInfo = contactInformation.SaveContactInfo(Dbnum, Csysuser, dsContactInfo, dsLocs,
AccountNum);
    ...
    // go thru the dataset and examine ["error_code"] to make sure save went thru
}
```

cContactInfo.DeleteContactInfo ()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will delete the contact information linked to pCustContactID

Method Signature:

```
public int DeleteContactInfo(short pCompanyID, string pSysUser, int pCustContactID)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCustContactID	integer	Identifies the customer contact record to delete
Return Type: int – Error Code 0 = no error <>, 0 = Error code (use cUtility.GetErrorDescription for error msg)			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add>DeleteContactInfo>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCustContactID>4499</add:pCustContactID>
    </add>DeleteContactInfo>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: deleting contact information.

```
protected void btnDeleteContact_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    //
    Utility.cUtility errDesc = new Utility.cUtility();

    int errorCode = 0;
    int contactID = 4;
    short DBnum = 2;
    errorCode = contactInformation.DeleteContactInfo(DBnum, "WebUser", contactID);
```

```
if (errorCode == 0)
{
    // success
}
else
{
    // failure
    string errorMsg = "";
    errorMsg=errDesc.GetErrorDescription(errorCode);
    // ...
}
```

cContactInfo.LoadContactSummary()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load a summary of contact info for a given customer for display purposes.

Method Signature:

```
public DCEnergyAPI.ContactInfo.DSContactInfoSummary LoadContactSummary(short pCompanyID, string pSysUser, int pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	Identifies the customer for which to load the summary.
Return Type: DSContactInfoSummary DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContactSummary>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>12</add:pAccountNum>
    </add:LoadContactSummary>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list summary in grid.

```
protected void btnContactSummary_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    DCEnergyAPI.ContactInfo.DSContactInfoSummary contactSummary = new
DCEnergyAPI.ContactInfo.DSContactInfoSummary ();
    short DBNum = 2;
    int accountNum = 1;
    contactSummary = contactInformation.LoadContactSummary(DBNum , "WebUser", accountNum );
    gvMain.DataSource = contactSummary;
    gvMain.DataBind();
}
```

Result:

contact_value	type	contact_type_id	primary_flag	location_type	location_num	description	notification	privacy	notes
5144289022	1	0	0	0	0		0	0	
5144289022	1	0	1	3	1	PHIL & DEBBIE HAMILTON / 3703 CAMINO HILLS DRIVE / 2220 SEVEN OAKS CT. / EL DORADO HILLS, CA , 956300000	0	0	
5147779999	1	1	0	0	0		0	0	
5551234567	1	18	1	0	0		0	0	L
9735555464	2	8	1	0	0		0	0	
CHANCHOUROVA@ADDSYS.COM	3	6	0	0	0		0	0	
CHANCHOUROVA@ADDSYS.COM	3	6	1	1	1	HAMILTON/6 LAUREL Dr//FLANDERS NJ 07836/extra line 1/ext line 2	0	0	
braudie@addsys.com	3	6	1	0	0		3	0	
david.capparelli@noaddress2.com	3	2	0	0	0		0	1	
dust@dust2.com	3	7	0	0	0		0	0	
dust@dust2.com	3	7	0	1	1	HAMILTON/6 LAUREL Dr//FLANDERS NJ 07836/extra line 1/ext line 2	0	0	
fomgc@addsys.com	3	6	0	0	0		0	0	Converted from EFT component

cContactInfo.LoadDocuments()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load a list of documents that can be selected for delivery as well as an indicator to show whether the document is selected for the current contact record. If you plan to link documents via the SaveContactInfoWithDocuments method, you will need this list.

Method Signature:

```
public DCCustomers.PushDocument.DSLinkedDoc LoadDocuments (short pCompanyID,
string pSysUser, int pCustContactID)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pCustContactID	Int (32 bit integer)	The ID of the customer contact record that the document selection will be set for
Return Type: DSLinkedDoc DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDocuments>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pCustContactID>4500</add:pCustContactID>
    </add:LoadDocuments>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list location(s) for given account/contact id in grid

```
protected void btnLoadDocuments_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    DCCustomers.PushDocument.DSLinkedDoc contactDocs = new
DCCustomers.PushDocument.DSLinkedDoc ();
    short dbNum = 2;
    int contactID = 4;
    contactDocs = contactInformation.LoadDocuments(dbNum, "WebUser", contactID);
    gvMain.DataSource = contactDocs;
```

```
gvMain.DataBind();
```

```
}
```

Result (partial list):

	doc_type	output_format	suppress_print	selected	doc_group	description	must_print
	32	0	0	0	0	Credit Card Notifications	0
	33	0	0	0	0	EFT Notifications	0
	20	0	0	0	1	Delivery Tickets	1
	6	0	0	0	2	Service Contract Invoices	0
	7	0	0	0	2	Mailer Service Contract Invoices	0
	23	9	0	1	3	WDMS Invoices	0
	27	9	0	1	3	WDMS Work Orders	0

cContactInfo.LoadDocumentOutputFormats()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will load a list of supported document output formats specified for a document that is selected for delivery. If you plan to link documents via the new SaveContactInfoWithDocuments method, you will need this list.

Method Signature:

```
public DCAddCommon.DSNumCodeDescription LoadDocumentOutputFormats (short
pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DSNumCodeDescription DataSet			

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDocumentOutputFormats>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadDocumentOutputFormats>
  </soap:Body>
</soap:Envelope>
```

.NET Example

Eg: load/list of available Document Output Formats

```
protected void btnLoadDocuments_click(object sender, EventArgs e)
{
    EnergyAPI.cContactInfo contactInformation = new EnergyAPI.cContactInfo();
    DCAddCommon.DSNumCodeDescription op = new DCAddCommon.DSNumCodeDescription()
    short dbNum = 2;
    op = contactInformation.LoadDocumentOutputFormats (dbNum, "WebUser");
    gvMain.DataSource = contactDocs;
    gvMain.DataBind();
}
```

Result (current list):

	code	description	status	selected	display
	2	Rich text (*.rtf)	A	False	2 - Rich text (*.rtf)
	3	Plain text (*.txt)	A	False	3 - Plain text (*.txt)

	4	Word 97 - 03 (*.doc)	A	False	4 - Word 97 - 03 (*.doc)
	5	Word 2007 (*.docx)	A	False	5 - Word 2007 (*.docx)
	9	Portable doc format (*.pdf)	A	False	9 - Portable doc format (*.pdf)
	10	Web page (*.html)	A	False	10 - Web page (*.html)

cContactInfo.SaveContactInfoWithDocuments ()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API Customer Add & Modify Bus Edition**

Description

This method will save any changes to the contact info and / or linked locations and / or linked documents (selected for automatic transmission to the contact record)

Note that you can save only one contact info record at a time along with its linked location information and documents – pass a pDSContactInfo dataset with only one record in it – the one that is to be saved. DSContactInfo is required, DSLinkedLocations and DSLinkedDocuments are optional and will only be saved if passed.

Method Signature:

```
public DCCustomers.PushDocument.DSContactInfo
SaveContactInfoWithDocuments(short pCompanyID, string pSysUser,
DCCustomers.PushDocument.DSContactInfo pDSContactInfo,
DCCustomers.PushDocument.DSLinkedLoc pDSLinkedLocations,
DCCustomers.PushDocument.DSLinkedDoc pDSLinkedDocuments, int pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pDSContactInfo	DSContactInfo	DataSet containing changed Contact Info values (Only One Row)
	pDSLinkedLocations	DSLinkedLoc	DataSet containing changed or unchanged linked location values
	pDSLinkedDocuments	DSLinkedDoc	DataSet containing Linked Documents
	pAccountNum	integer	Customer Number linked to the contact info that is being saved
Return Type: DSContactInfo DataSet (updated values)			

This method is similar to the [SaveContactInfo](#) method so only the steps for linking the documents will be shown. Note that the doc_group value of the [DSLinkedDoc](#) DataSet indicates which location type it is used for and the Contact Info record must be linked to at least 1 location of this type – a document cannot be linked to a telephone (DSContactInfo row with type = 1).

Step 1 – Load the ContactInfo dataset and select 1 ContactInfo record.

Step 2 – (Optional) Load the DSLinkedLoc Locations for the selected contact.

Step 3 – Load the DSLinkedDoc for the selected contact using the [LoadDocuments](#) call

Set the “selected” value to 1 for any contact info record that you want to link and set the output format to one of the values listed in the [LoadDocumentOutputFormats](#) result.

Call the SaveContactInfoWithDocuments method.

.XML Document

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveContactInfoWithDocuments>
      <add:pCompanyID>2</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pDSContactInfo>
        <CONTACT_INFO_HDR>
          <cust_contact_id>0</cust_contact_id>
          <contact_type_id>29</contact_type_id>
          <type>3</type>
          <contact_value>fongs@addsys.com</contact_value>
          <notes></notes>
          <notification>0</notification>
          <privacy>0</privacy>
          <row_status>A</row_status>
          <last_maintenance_dt></last_maintenance_dt>
          <last_maintenance_userid></last_maintenance_userid>
          <error_code>0</error_code>
        </CONTACT_INFO_HDR>
      </add:pDSContactInfo>
      <add:pDSLinkedLocations>
        <CONTACT_INFO_DTL>
          <cust_contact_id>0</cust_contact_id>
          <location_type>0</location_type>
          <location_num>0</location_num>
          <primary_flag>1</primary_flag>
          <selected>1</selected>
          <description></description>
        </CONTACT_INFO_DTL>
        <CONTACT_INFO_DTL>
          <cust_contact_id>0</cust_contact_id>
          <location_type>1</location_type>
          <location_num>1</location_num>
          <primary_flag>0</primary_flag>
          <selected>0</selected>
          <description>DANIEL MURPHY/551 Shady Oak Ln./Diamond Springs,
CA,95619</description>
        </CONTACT_INFO_DTL>
        <CONTACT_INFO_DTL>
          <cust_contact_id>0</cust_contact_id>
          <location_type>2</location_type>
          <location_num>1</location_num>
          <primary_flag>0</primary_flag>
          <selected>0</selected>
```

```

        <description>DANIEL MURPHY/551 SHADY OAK LANEDIAMOND SPRINGS/./,
,956670000</description>
    </CONTACT_INFO_DTL>
</add:pDSLinkedLocations>
<add:pDSLinkedDocuments>
    <DOC_CUST_NOTIFY>
        <doc_type>1</doc_type>
        <output_format>9</output_format>
        <suppress_print>0</suppress_print>
        <selected>1</selected>
        <doc_group>0</doc_group>
        <description>Statements</description>
        <must_print>0</must_print>
    </DOC_CUST_NOTIFY>
</add:pDSLinkedDocuments>
<add:pAccountNum>14</add:pAccountNum>
</add:SaveContactInfoWithDocuments>
</soap:Body>
</soap:Envelope>

```

.NET Example

Example linking and saving a document:

```

DCCustomers.PushDocument.DSContactInfo dsContactInfoLoad = null;
DCCustomers.PushDocument.DSContactInfo dsContactInfoSave = new
DCCustomers.PushDocument.DSContactInfo();
DCCustomers.PushDocument.DSLinkedDoc dsDocuments = null;

short company_id = 2;
string sysuser = "WebUser";
int accountnum = 498;
int contactID = 0;
int errorCode = 0;

using (EnergyAPI.cContactInfo contactAPI = new EnergyAPI.cContactInfo())
{
    dsContactInfoLoad = contactAPI.LoadContactInfo(company_id, sysuser, accountnum);
    if (dsContactInfoLoad != null && dsContactInfoLoad.Tables[0].Rows.Count > 0)
    {
        //Take the first row
        dsContactInfoSave.CONTACT_INFO_HDR.Rows.Add
            (dsContactInfoLoad.CONTACT_INFO_HDR.Rows[0].ItemArray);
        dsContactInfoSave.AcceptChanges();
        dsContactInfoLoad.Clear();
        dsContactInfoLoad.Dispose();
        if (int.TryParse(dsContactInfoSave.CONTACT_INFO_HDR.Rows[0]["cust_contact_id"]
            .ToString(), out contactID))
        {
            dsDocuments = contactAPI.LoadDocuments(company_id, sysuser, contactID);
            if (dsDocuments != null && dsDocuments.Tables[0].Rows.Count > 0)
            {
                //select the first document for delivery
                dsDocuments.Tables[0].Rows[0]["selected"] = 1;
            }
        }
    }
}

```

```
dsDocuments.Tables[0].Rows[0]["output_format"] = 9;
dsContactInfoLoad = contactAPI.SaveContactInfoWithDocuments(company_id,
    sysuser, dsContactInfoSave, null, dsDocuments, accountnum);
if (dsContactInfoLoad != null &&
    dsContactInfoLoad.Tables[0].Rows.Count > 0)
{
    if (int.TryParse(dsContactInfoLoad.Tables[0].Rows[0]["error_code"]
        .ToString(), out errorCode))
    {
        if (errorCode != 0)
        {
            // Get Error Description
            using (EnergyAPI.cUtility util = new EnergyAPI.cUtility())
            {
                string msg = util.GetErrorDescription(errorCode);
                Console.WriteLine(msg);
                Console.ReadKey();
            }
        }
        //else save was successful
    }
    //else something went wrong
}
}
}
}}}
```

cOrder class

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect API WDMS Order Creation/Maintenance**

Overview

***** This class must be used for databases whose E3 release is 14.x.**

The cOrder class provides methods create and maintain orders.

This section describes the methods that will enable the programmer to create/modify orders in E3.

- SaveOrders(): Creates/modifies orders

Data Structures used:

Order Definition:

Datatype/column name	Description
<code>int</code> order_sequence_number	Intended to be used by the client, to uniquely identify records. No check is performed by the web method to validate uniqueness.
<code>string</code> order_type	Order type. Maximum length: 3.
<code>string</code> clerk	Clerk code. Maximum length: 5.
<code>short</code> cost_center	Cost center. Range: 0 to 32766
<code>int</code> account_num	Customer account number. Range: 1 to 999999
<code>short</code> location_num	Location number. Range: 1 to 32766
<code>string</code> customer_po_number	Customer PO number. Maximum length: 25.
<code>string</code> text	Notes. Maximum length: 50.
<code>byte</code> base_price_code	Base price code
<code>string</code> ship_via	Hauler code or free form text. Maximum length: 10.
<code>decimal</code> freight_charge	Freight charge
<code>string</code> sales_person	Sales person code. Maximum length: 3.
<code>string</code> reference_num	Reference number. Maximum length: 15.
<code>string</code> priority_order	Priority order. Acceptable values if not null: Y or N. Maximum length: 1.
<code>DateTime</code> requested_delivery_dt	Requested delivery date and time
<code>int</code> hauler_office_id	Hauler office id. Range: 0 to 999999
<code>int</code> modified_order_num	The order number to be modified. Should be 0 for creating new order. Range: 0 to 999999
<code>bool</code> flag_for_dispatch	Flag order for dispatch
<code>AddressOverride</code> address_override	See details below
<code>OrderLine[]</code> order_lines	See details below

AddressOverride Definition:

Datatype/column name	Description
string telephone	Telephone number. Maximum length: 10
string name	Name. Maximum length: 40.
string street_1	Street 1. Maximum length: 40.
string street_2	Street 2. Maximum length: 40.
string city	City. Maximum length: 40.
string state	State. Maximum length: 4.
byte country	Country. 0 = USA, 1 = Canada
string postal_code	Postal/Zip code. Maximum length: 10.

OrderLine Definition:

Datatype/column name	Description
short order_line	Order line. Value is ignored if action is 'insert'.
Enums.Action action	insert = create new order line, update = update existing order line, delete = delete existing order line. Currently, update is not supported.
string mfg_code	Manufacturer code. Maximum length: 4.
string group_code	Group code. Maximum length: 6.
string stock_id	Stock Id. Maximum length: 12. The combination of mfg_code, group_code and stock_id must match to a valid part in inventory
string unit_name	Unit of measure. Maximum length: 4.
decimal order_qty	Quantity. Range: 0 to 999999999.999
decimal unit_price	Price per unit. Range: 0 to 99999.9999
string whse_code*	Warehouse code. Maximum length: 3.
string bin_code*	Position. Maximum length: 4.
string location_code*	Location code. Maximum length: 10.
string ms_code*	Supplier code. Maximum length: 4.
string loading_auth_num	Loading authorization number. Maximum length: 4.

* Provide either a whse_code/bin_code or location_code/ms_code combination depending on if the product is coming from a storage location or being purchased from a supplier.

SaveOrderResults Definition:

Datatype/column name	Description
List<SaveOrderResult> result	See below
SaveOrderException exception	See below

SaveOrderResult Definition:

Datatype/column name	Description
int order_sequence_number	Sequence number supplied by client
int order_number	If success, a number greater than 0. If failure, 0.
SaveOrderException exception	See below

SaveOrderException Definition:

Datatype/column name	Description
Enums.ExceptionObjectType exception_object_type	none = general error, order = if error occurred specific to order object, order_line = if error occurred specific to order_line object
int error_object_sequence_number	If exception_object_type = order, it represents order_sequence_number supplied by client If exception_object_type = order_line, it represents order_line number supplied by client If exception_object_type = none, this value would always be 0 and should be ignored by client
int error_number	Error number
string error_message	Error message corresponding to error number

cOrder.SaveOrders()

- License Required: SmartConnect API Customer Look-up Business Edition**

Description

This method will create and modify orders.

Method Signature:

```
Public Function SaveOrders(p_Orders As DCEnergyAPI.WDMS.POS.Orders, ByVal p_iCompanyID As Short, ByVal p_sSysUser As String) As DCEnergyAPI.WDMS.POS.SaveOrdersResults
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	p_Orders	Orders	List of Order objects
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: SaveOrdersResults class			

.XML Documents:

Example of a successful request and response	REQUEST <pre><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:pos="http://www.addsys.com/WDMS/POS"> <soapenv:Header/> <soapenv:Body> <pos:SaveOrders> <pos:orders_root> <pos:orders> <pos:order> <pos:order_sequence_number>1</pos:order_sequence_number> <pos:order_type>BOL</pos:order_type> <pos:clerk>ADD</pos:clerk> <pos:cost_center>5</pos:cost_center></pre>

```

<pos:account_num>209</pos:account_num>
<pos:location_num>1</pos:location_num>
<pos:customer_po_number>CP02048</pos:customer_po_number>
<pos:text>CALL 10 MINUTES PRIOR TO DELIVERY</pos:text>
<pos:base_price_code>0</pos:base_price_code>
<pos:ship_via/>
<pos:freight_charge>0</pos:freight_charge>
<pos:sales_person>99</pos:sales_person>
<pos:reference_num>10935</pos:reference_num>
<pos:priority_order/>
<pos:requested_delivery_dt>2014-12-
25T21:30:00</pos:requested_delivery_dt>
<pos:hauler_office_id>0</pos:hauler_office_id>
<pos:modified_order_num>0</pos:modified_order_num>
<pos:flag_for_dispatch>false</pos:flag_for_dispatch>
<pos:address_override>
  <pos:telephone>5145145145</pos:telephone>
  <pos:name>ADDR694</pos:name>
  <pos:street_1>694 Avenue 22</pos:street_1>
  <pos:street_2/>
  <pos:city>PHL</pos:city>
  <pos:state>PA</pos:state>
  <pos:country>0</pos:country>
  <pos:postal_code>715857</pos:postal_code>
</pos:address_override>
<pos:order_lines>
  <pos:order_line>
    <pos:order_line>1</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>7</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>11</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code>105</pos:whse_code>
    <pos:bin_code>2</pos:bin_code>
    <pos:location_code/>
    <pos:ms_code/>
    <pos:loading_auth_num>LAN22</pos:loading_auth_num>
  </pos:order_line>
  <pos:order_line>
    <pos:order_line>2</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>9</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>10</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code>100</pos:whse_code>
    <pos:bin_code>1</pos:bin_code>
    <pos:location_code/>
    <pos:ms_code/>
    <pos:loading_auth_num>AF022</pos:loading_auth_num>
  </pos:order_line>
</pos:order_lines>
</pos:order>
<pos:order>

```

```

<pos:order_sequence_number>2</pos:order_sequence_number>
<pos:order_type>DS</pos:order_type>
<pos:clerk>ADD</pos:clerk>
<pos:cost_center>5</pos:cost_center>
<pos:account_num>209</pos:account_num>
<pos:location_num>1</pos:location_num>
<pos:customer_po_number>CP019372</pos:customer_po_number>
<pos:text>DO NOT DELIVER AFTER 10 PM</pos:text>
<pos:base_price_code>0</pos:base_price_code>
<pos:ship_via/>
<pos:freight_charge>0</pos:freight_charge>
<pos:sales_person>14</pos:sales_person>
<pos:reference_num>1</pos:reference_num>
<pos:priority_order>Y</pos:priority_order>
<pos:requested_delivery_dt>2014-12-
30T09:30:00</pos:requested_delivery_dt>
<pos:hauler_office_id>0</pos:hauler_office_id>
<pos:modified_order_num>0</pos:modified_order_num>
<pos:flag_for_dispatch>false</pos:flag_for_dispatch>
<pos:address_override>
  <pos:telephone/>
  <pos:name>SM_ADDR</pos:name>
  <pos:street_1>209 Wildwest Drive</pos:street_1>
  <pos:street_2>Terrace 1</pos:street_2>
  <pos:city>Dallas</pos:city>
  <pos:state>TX</pos:state>
  <pos:country>0</pos:country>
  <pos:postal_code>125695</pos:postal_code>
</pos:address_override>
<pos:order_lines>
  <pos:order_line>
    <pos:order_line>1</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>7</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>11</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code/>
    <pos:bin_code/>
    <pos:location_code>324</pos:location_code>
    <pos:ms_code>A22</pos:ms_code>
    <pos:loading_auth_num>22987</pos:loading_auth_num>
  </pos:order_line>
  <pos:order_line>
    <pos:order_line>2</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>9</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>22</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code/>
    <pos:bin_code/>
    <pos:location_code>BUCK</pos:location_code>
    <pos:ms_code>LIQ</pos:ms_code>
    <pos:loading_auth_num>LAN2576</pos:loading_auth_num>
  </pos:order_line>

```

	<pre> </pos:order_line> <pos:order_line> <pos:order_line>3</pos:order_line> <pos:action>insert</pos:action> <pos:mfg_code>LNX</pos:mfg_code> <pos:group_code>CTR</pos:group_code> <pos:stock_id>0G1901</pos:stock_id> <pos:unit_name>EACH</pos:unit_name> <pos:order_qty>2</pos:order_qty> <pos:unit_price>0</pos:unit_price> <pos:whse_code/> <pos:bin_code/> <pos:location_code>BALT</pos:location_code> <pos:ms_code>EXX0</pos:ms_code> <pos:loading_auth_num>69542576</pos:loading_auth_num> </pos:order_line> </pos:order_lines> </pos:order> </pos:orders> </pos:orders_root> <pos:p_iCompanyID>2</pos:p_iCompanyID> <pos:p_sSysUser>WebUser</pos:p_sSysUser> </pos:SaveOrders> </soapenv:Body> </soapenv:Envelope> </pre> RESPONSE <pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <soap:Body> <SaveOrdersResponse xmlns="http://www.addsys.com/WDMS/POS"> <results_root> <results> <result> <order_sequence_number>1</order_sequence_number> <order_number>21304</order_number> </result> <result> <order_sequence_number>2</order_sequence_number> <order_number>21305</order_number> </result> </results> </results_root> </SaveOrdersResponse> </soap:Body> </soap:Envelope> </pre>
Example of a general exception	REQUEST <pre> <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:pos="http://www.addsys.com/WDMS/POS"> <soapenv:Header/> <soapenv:Body> <pos:SaveOrders> <pos:orders_root> <pos:orders> <pos:order> <pos:order_sequence_number>1</pos:order_sequence_number> <pos:order_type>BOL</pos:order_type> <pos:clerk>ADD</pos:clerk> </pre>

```

<pos:cost_center>5</pos:cost_center>
<pos:account_num>209</pos:account_num>
<pos:location_num>1</pos:location_num>
<pos:customer_po_number>CP02048</pos:customer_po_number>
<pos:text>CALL 10 MINUTES PRIOR TO DELIVERY</pos:text>
<pos:base_price_code>0</pos:base_price_code>
<pos:ship_via/>
<pos:freight_charge>0</pos:freight_charge>
<pos:sales_person>99</pos:sales_person>
<pos:reference_num>10935</pos:reference_num>
<pos:priority_order/>
<pos:requested_delivery_dt>2014-12-
25T21:30:00</pos:requested_delivery_dt>
<pos:hauler_office_id>0</pos:hauler_office_id>
<pos:modified_order_num>0</pos:modified_order_num>
<pos:flag_for_dispatch>false</pos:flag_for_dispatch>
<pos:address_override>
  <pos:telephone>5145145145</pos:telephone>
  <pos:name>ADDR694</pos:name>
  <pos:street_1>694 Avenue 22</pos:street_1>
  <pos:street_2/>
  <pos:city>PHL</pos:city>
  <pos:state>PA</pos:state>
  <pos:country>0</pos:country>
  <pos:postal_code>715857</pos:postal_code>
</pos:address_override>
<pos:order_lines>
  <pos:order_line>
    <pos:order_line>1</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>7</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>11</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code>105</pos:whse_code>
    <pos:bin_code>2</pos:bin_code>
    <pos:location_code/>
    <pos:ms_code/>
    <pos:loading_auth_num>LAN22</pos:loading_auth_num>
  </pos:order_line>
  <pos:order_line>
    <pos:order_line>2</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>9</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>10</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code>100</pos:whse_code>
    <pos:bin_code>1</pos:bin_code>
    <pos:location_code/>
    <pos:ms_code/>
    <pos:loading_auth_num>AF022</pos:loading_auth_num>
  </pos:order_line>
</pos:order_lines>
</pos:order>

```

```

30T09:30:00<pos:order>
  <pos:order_sequence_number>2</pos:order_sequence_number>
  <pos:order_type>DS</pos:order_type>
  <pos:clerk>ADD</pos:clerk>
  <pos:cost_center>5</pos:cost_center>
  <pos:account_num>209</pos:account_num>
  <pos:location_num>1</pos:location_num>
  <pos:customer_po_number>CP019372</pos:customer_po_number>
  <pos:text>DO NOT DELIVER AFTER 10 PM</pos:text>
  <pos:base_price_code>0</pos:base_price_code>
  <pos:ship_via/>
  <pos:freight_charge>0</pos:freight_charge>
  <pos:sales_person>14</pos:sales_person>
  <pos:reference_num>1</pos:reference_num>
  <pos:priority_order>Y</pos:priority_order>
  <pos:requested_delivery_dt>2014-12-
  </pos:requested_delivery_dt>
  <pos:hauler_office_id>0</pos:hauler_office_id>
  <pos:modified_order_num>0</pos:modified_order_num>
  <pos:flag_for_dispatch>false</pos:flag_for_dispatch>
  <pos:address_override>
    <pos:telephone/>
    <pos:name>SM_ADDR</pos:name>
    <pos:street_1>209 Wildwest Drive</pos:street_1>
    <pos:street_2>Terrace 1</pos:street_2>
    <pos:city>Dallas</pos:city>
    <pos:state>TX</pos:state>
    <pos:country>0</pos:country>
    <pos:postal_code>125695</pos:postal_code>
  </pos:address_override>
  <pos:order_lines>
    <pos:order_line>
      <pos:order_line>1</pos:order_line>
      <pos:action>insert</pos:action>
      <pos:mfg_code>LIQ</pos:mfg_code>
      <pos:group_code>GAS</pos:group_code>
      <pos:stock_id>7</pos:stock_id>
      <pos:unit_name>GAL</pos:unit_name>
      <pos:order_qty>11</pos:order_qty>
      <pos:unit_price>0</pos:unit_price>
      <pos:whse_code/>
      <pos:bin_code/>
      <pos:location_code>324</pos:location_code>
      <pos:ms_code>A22</pos:ms_code>
      <pos:loading_auth_num>22987</pos:loading_auth_num>
    </pos:order_line>
    <pos:order_line>
      <pos:order_line>2</pos:order_line>
      <pos:action>insert</pos:action>
      <pos:mfg_code>LIQ</pos:mfg_code>
      <pos:group_code>GAS</pos:group_code>
      <pos:stock_id>9</pos:stock_id>
      <pos:unit_name>GAL</pos:unit_name>
      <pos:order_qty>22</pos:order_qty>
      <pos:unit_price>0</pos:unit_price>
      <pos:whse_code/>
      <pos:bin_code/>
      <pos:location_code>BUCK</pos:location_code>
      <pos:ms_code>LIQ</pos:ms_code>

```

```

        <pos:loading_auth_num>LAN2576</pos:loading_auth_num>
      </pos:order_line>
    <pos:order_line>
      <pos:order_line>3</pos:order_line>
      <pos:action>insert</pos:action>
      <pos:mfg_code>LNX</pos:mfg_code>
      <pos:group_code>CTR</pos:group_code>
      <pos:stock_id>0G1901</pos:stock_id>
      <pos:unit_name>EACH</pos:unit_name>
      <pos:order_qty>2</pos:order_qty>
      <pos:unit_price>0</pos:unit_price>
      <pos:whse_code/>
      <pos:bin_code/>
      <pos:location_code>BALT</pos:location_code>
      <pos:ms_code>EXX0</pos:ms_code>
      <pos:loading_auth_num>69542576</pos:loading_auth_num>
    </pos:order_line>
  </pos:order_lines>
</pos:order>
</pos:orders>
</pos:orders_root>
<pos:p_iCompanyID>2</pos:p_iCompanyID>
<pos:p_sSysUser>WebUser2</pos:p_sSysUser>
</pos:SaveOrders>
</soapenv:Body>
</soapenv:Envelope>

```

RESPONSE

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <SaveOrdersResponse xmlns="http://www.addsys.com/WDMS/POS">
      <results_root>
        <results/>
        <exception>
          <exception_object_type>none</exception_object_type>
          <error_object_sequence_number>0</error_object_sequence_number>
          <error_number>96</error_number>
          <error_message>
            System.Exception: Invalid database connection requested
              at EnergyAPI.mErrLibrary.RaiseError(Exception p_oEx, String
p_sModule, String p_sMethod, String p_sAltDesc, String& p_sSysUser,
Int16 p_iLogVerbosity) in D:\_DEVEL\DVL\Smart
Connect\EnergyAPI\mErrLibrary.vb:line 43
              at EnergyAPI.cUtility.GetDatabaseVersion(Int32 p_iCompanyID,
String p_sSysUser) in D:\_DEVEL\DVL\Smart
Connect\EnergyAPI\Utility\cUtility.vb:line 153
              at EnergyAPI.WDMS.POS.cOrder.ValidateDatabaseVersion(String
pMethod) in d:\_DEVEL\DVL\Smart
Connect\EnergyAPI2\WDMS\POS\cOrder.cs:line 566
          </error_message>
        </exception>
      </results_root>
    </SaveOrdersResponse>
  </soap:Body>
</soap:Envelope>

```

Example of error
processing one order

REQUEST

```

<soapenv:Envelope

```

and success in
another

```

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:pos="http://www.addsys.com/WDMS/POS">
  <soapenv:Header/>
  <soapenv:Body>
    <pos:SaveOrders>
      <pos:orders_root>
        <pos:orders>
          <pos:order>
            <pos:order_sequence_number>1</pos:order_sequence_number>
            <pos:order_type>BOL</pos:order_type>
            <pos:clerk>ADD</pos:clerk>
            <pos:cost_center>5</pos:cost_center>
            <pos:account_num>209</pos:account_num>
            <pos:location_num>1</pos:location_num>
            <pos:customer_po_number>CPO2048</pos:customer_po_number>
            <pos:text>CALL 10 MINUTES PRIOR TO DELIVERY</pos:text>
            <pos:base_price_code>0</pos:base_price_code>
            <pos:ship_via/>
            <pos:freight_charge>0</pos:freight_charge>
            <pos:sales_person>99</pos:sales_person>
            <pos:reference_num>10935</pos:reference_num>
            <pos:priority_order/>
            <pos:requested_delivery_dt>2014-12-
25T21:30:00</pos:requested_delivery_dt>
            <pos:hauler_office_id>0</pos:hauler_office_id>
            <pos:modified_order_num>21304</pos:modified_order_num>
            <pos:flag_for_dispatch>false</pos:flag_for_dispatch>
            <pos:address_override>
              <pos:telephone>5145145145</pos:telephone>
              <pos:name>ADDR694</pos:name>
              <pos:street_1>694 Avenue 22</pos:street_1>
              <pos:street_2/>
              <pos:city>PHL</pos:city>
              <pos:state>PA</pos:state>
              <pos:country>0</pos:country>
              <pos:postal_code>715857</pos:postal_code>
            </pos:address_override>
            <pos:order_lines>
              <pos:order_line>
                <pos:order_line>3</pos:order_line>
                <pos:action>delete</pos:action>
                <pos:mfg_code>LIQ</pos:mfg_code>
                <pos:group_code>GAS</pos:group_code>
                <pos:stock_id>7</pos:stock_id>
                <pos:unit_name>GAL</pos:unit_name>
                <pos:order_qty>11</pos:order_qty>
                <pos:unit_price>0</pos:unit_price>
                <pos:whse_code>105</pos:whse_code>
                <pos:bin_code>2</pos:bin_code>
                <pos:location_code/>
                <pos:ms_code/>
                <pos:loading_auth_num>LAN22</pos:loading_auth_num>
              </pos:order_line>
            </pos:order_lines>
          </pos:order>
          <pos:order>
            <pos:order_sequence_number>2</pos:order_sequence_number>
            <pos:order_type>DS</pos:order_type>
            <pos:clerk>ADD</pos:clerk>

```



```

<pos:cost_center>5</pos:cost_center>
<pos:account_num>209</pos:account_num>
<pos:location_num>1</pos:location_num>
<pos:customer_po_number>CP019372</pos:customer_po_number>
<pos:text>DO NOT DELIVER AFTER 10 PM</pos:text>
<pos:base_price_code>0</pos:base_price_code>
<pos:ship_via/>
<pos:freight_charge>0</pos:freight_charge>
<pos:sales_person>14</pos:sales_person>
<pos:reference_num>1</pos:reference_num>
<pos:priority_order>Y</pos:priority_order>
<pos:requested_delivery_dt>2014-12-
30T09:30:00</pos:requested_delivery_dt>
<pos:hauler_office_id>0</pos:hauler_office_id>
<pos:modified_order_num>21305</pos:modified_order_num>
<pos:flag_for_dispatch>false</pos:flag_for_dispatch>
<pos:address_override>
  <pos:telephone/>
  <pos:name>SM_ADDR</pos:name>
  <pos:street_1>209 Wildwest Drive</pos:street_1>
  <pos:street_2>Terrace 1</pos:street_2>
  <pos:city>Dallas</pos:city>
  <pos:state>TX</pos:state>
  <pos:country>0</pos:country>
  <pos:postal_code>125695</pos:postal_code>
</pos:address_override>
<pos:order_lines>
  <pos:order_line>
    <pos:order_line>1</pos:order_line>
    <pos:action>delete</pos:action>
    <pos:mfg_code>LIQ</pos:mfg_code>
    <pos:group_code>GAS</pos:group_code>
    <pos:stock_id>7</pos:stock_id>
    <pos:unit_name>GAL</pos:unit_name>
    <pos:order_qty>11</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code/>
    <pos:bin_code/>
    <pos:location_code>324</pos:location_code>
    <pos:ms_code>A22</pos:ms_code>
    <pos:loading_auth_num>22987</pos:loading_auth_num>
  </pos:order_line>
  <pos:order_line>
    <pos:order_line>4</pos:order_line>
    <pos:action>insert</pos:action>
    <pos:mfg_code>LNX</pos:mfg_code>
    <pos:group_code>CTR</pos:group_code>
    <pos:stock_id>0G1901</pos:stock_id>
    <pos:unit_name>EACH</pos:unit_name>
    <pos:order_qty>25</pos:order_qty>
    <pos:unit_price>0</pos:unit_price>
    <pos:whse_code/>
    <pos:bin_code/>
    <pos:location_code>BALT</pos:location_code>
    <pos:ms_code>EXX0</pos:ms_code>
    <pos:loading_auth_num>69542576</pos:loading_auth_num>
  </pos:order_line>
</pos:order_lines>
</pos:order>

```

	<pre> </pos:orders> </pos:orders_root> <pos:p_iCompanyID>2</pos:p_iCompanyID> <pos:p_sSysUser>WebUser</pos:p_sSysUser> </pos:SaveOrders> </soapenv:Body> </soapenv:Envelope> </pre>
	RESPONSE <pre> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <soap:Body> <SaveOrdersResponse xmlns="http://www.addsys.com/WDMS/POS"> <results_root> <results> <result> <order_sequence_number>1</order_sequence_number> <order_number>0</order_number> <exception> <exception_object_type>order_line</exception_object_type> <error_object_sequence_number>3</error_object_sequence_number> <error_number>100017</error_number> <error_message>Invalid order line</error_message> </exception> </result> <result> <order_sequence_number>2</order_sequence_number> <order_number>21305</order_number> </result> </results> </results_root> </SaveOrdersResponse> </soap:Body> </soap:Envelope> </pre>

cBestBuy class

- **License Required: SmartConnect API WDMS Order Creation/Maintenance**

Overview

***** This class must be used for databases whose E3 release is 14.x.**

The cBestBuy class provides methods for best buy information.

URL: <http://<WEBSERVICE>/wdms/pos/bestbuy/cBestBuy.asmx>

This section describes the methods that will enable the programmer to get best buy information from E3.

- **GetBestBuy():** Gets best buy information for orders

cBestBuy.GetBestBuy()

- License Required: SmartConnect API Customer Look-up Business Edition

Description

This method will retrieve best buy information for an order.

SOAP 1.2

The following is a sample SOAP 1.2 request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /wdms/pos/bestbuy/cBestBuy.asmx HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetBestBuy xmlns="http://www.addsys.com/WDMS/POS/BestBuy">
      <p_iCompanyID>short</p_iCompanyID>
      <p_sSysUser>string</p_sSysUser>
      <p_iOrderNumber>int</p_iOrderNumber>
      <p_iOrderLine>short</p_iOrderLine>
      <p_bIncludeOutOfPriceMarket>boolean</p_bIncludeOutOfPriceMarket>
      <p_bIncludeZeroFreightCost>boolean</p_bIncludeZeroFreightCost>
      <p_bIncludeUnbrandedSupplier>boolean</p_bIncludeUnbrandedSupplier>
    </GetBestBuy>
  </soap12:Body>
</soap12:Envelope>
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetBestBuyResponse xmlns="http://www.addsys.com/WDMS/POS/BestBuy">
      <best_buy_root>
        <best_buys>
          <best_buy>
            <mkt_desc>string</mkt_desc>
            <hauler_code>string</hauler_code>
            <hauler_office_id>int</hauler_office_id>
            <hauler_office_desc>string</hauler_office_desc>
            <supplier>string</supplier>
            <supplier_desc>string</supplier_desc>
            <supply_location>string</supply_location>
            <supply_location_desc>string</supply_location_desc>
            <totals xsi:nil="true" />
            <order_lines xsi:nil="true" />
            <exception xsi:nil="true" />
          </best_buy>
          <best_buy>
            <mkt_desc>string</mkt_desc>
            <hauler_code>string</hauler_code>
            <hauler_office_id>int</hauler_office_id>
            <hauler_office_desc>string</hauler_office_desc>
            <supplier>string</supplier>
            <supplier_desc>string</supplier_desc>
            <supply_location>string</supply_location>
            <supply_location_desc>string</supply_location_desc>
            <totals xsi:nil="true" />
            <order_lines xsi:nil="true" />
            <exception xsi:nil="true" />
          </best_buy>
        </best_buys>
      </best_buy_root>
    </GetBestBuyResponse>
  </soap12:Body>
</soap12:Envelope>
```

HTTP POST

The following is a sample HTTP POST request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /wdms/pos/bestbuy/cBestBuy.asmx/GetBestBuy HTTP/1.1
Host: localhost
Content-Type: application/x-www-form-urlencoded
Content-Length: length
```

p_iCompanyID=string&p_sSysUser=string&p_iOrderNumber=string&p_iOrderLine=string&p_bIncludeOutOfPriceMarket=string&p_bIncludeZeroFreightCost=string&p_bIncludeUnbrandedSupplier=string
 HTTP/1.1 200 OK
 Content-Type: text/xml; charset=utf-8
 Content-Length: **length**

```

<?xml version="1.0" encoding="utf-8"?>
<best_buy_root xmlns="http://www.addsys.com/WDMS/POS/BestBuy">
  <best_buys>
    <best_buy>
      <mkt_desc>string</mkt_desc>
      <hauler_code>string</hauler_code>
      <hauler_office_id>int</hauler_office_id>
      <hauler_office_desc>string</hauler_office_desc>
      <supplier>string</supplier>
      <supplier_desc>string</supplier_desc>
      <supply_location>string</supply_location>
      <supply_location_desc>string</supply_location_desc>
      <totals>
        <quantity>decimal</quantity>
        <price>double</price>
        <unit_freight>decimal</unit_freight>
        <unit_add_ons>decimal</unit_add_ons>
        <total_unit_cost>decimal</total_unit_cost>
        <total_cost>decimal</total_cost>
        <difference>decimal</difference>
      </totals>
      <order_lines>
        <order_line d5p1:nil="true" xmlns:d5p1="http://www.w3.org/2001/XMLSchema-instance" />
        <order_line d5p1:nil="true" xmlns:d5p1="http://www.w3.org/2001/XMLSchema-instance" />
      </order_lines>
      <exception>
        <error_number>int</error_number>
        <error_message>string</error_message>
      </exception>
    </best_buy>
    <best_buy>
      <mkt_desc>string</mkt_desc>
      <hauler_code>string</hauler_code>
      <hauler_office_id>int</hauler_office_id>
      <hauler_office_desc>string</hauler_office_desc>
      <supplier>string</supplier>
      <supplier_desc>string</supplier_desc>
      <supply_location>string</supply_location>
      <supply_location_desc>string</supply_location_desc>
      <totals>
        <quantity>decimal</quantity>
        <price>double</price>
        <unit_freight>decimal</unit_freight>
        <unit_add_ons>decimal</unit_add_ons>
        <total_unit_cost>decimal</total_unit_cost>
        <total_cost>decimal</total_cost>
        <difference>decimal</difference>
      </totals>
      <order_lines>
        <order_line d5p1:nil="true" xmlns:d5p1="http://www.w3.org/2001/XMLSchema-instance" />
        <order_line d5p1:nil="true" xmlns:d5p1="http://www.w3.org/2001/XMLSchema-instance" />
      </order_lines>
      <exception>
        <error_number>int</error_number>
        <error_message>string</error_message>
      </exception>
    </best_buy>
  </best_buys>
  <exception>
    <error_number>int</error_number>
    <error_message>string</error_message>
  </exception>
</best_buy_root>

```

REQUEST PARAMETERS

short p_iCompanyID	Company id
string p_sSysUser	System user id
int p_iOrderNumber	The order number for which to fetch the best buy information
short p_iOrderLine	If p_iOrderLine = 0, the best buy options for the order summary will be returned. If p_iOrderLine > 0, the best buy options for the order line will be returned.
boolean p_bIncludeOutOfPriceMarket	Whether search should include out of price market
boolean p_bIncludeZeroFreightCost	Whether search should include zero freight cost
boolean p_bIncludeUnbrandedSupplier	Whether search should include unbranded supplier

RESPONSE CLASSES:

Class Name: best buy root	
best buy[] best buys	See class definition below
class exception	See class definition below

Class Name: best buy	
string mkt desc	Market description
string hauler code	Hauler code
int hauler office id	Hauler office id
string hauler office desc	Hauler office description
string supplier	Supplier
string supplier desc	Supplier description
string supply location	Supply location
string supply location desc	Supply location description
class totals	See class definition below
order line[] order lines	See class definition below
class exception	See class definition below

Class Name: totals	
decimal quantity	Sum of quantity from all order line in order lines
double price	Sum of price from all order line in order lines
decimal unit freight	Sum of unit freight from all order line in order lines
decimal unit add ons	Sum of unit add ons from all order line in order lines
decimal total unit cost	Sum of total unit cost from all order line in order lines
decimal total cost	Sum of total cost from all order line in order lines
decimal difference	Sum of difference from all order line in order lines

Class Name: order line	
decimal quantity	Quantity
double price	Price
decimal unit freight	Unit Freight
decimal unit add ons	Unit Add-ons
decimal total unit cost	Total unit cost
decimal total cost	Total cost
decimal difference	Difference
string mfg code	Manufacturing code
string group code	Group code
string stock id	Stock id
string product desc	Product description
short order line	Order line number
dateTime effective date	Effective date
class exception	See details below

Class Name: exception	
int error number	Error number if applicable
string error message	Error message

.XML Documents:

Example of a
successful request
and response

SAMPLE REQUEST

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bes="http://www.addsys.com/WDMS/POS/BestBuy">
  <soapenv:Header/>
  <soapenv:Body>
    <bes:GetBestBuy>
      <bes:p_iCompanyID>1600</bes:p_iCompanyID>
      <bes:p_sSysUser>agohel</bes:p_sSysUser>
      <bes:p_iOrderNumber>63653</bes:p_iOrderNumber>
      <bes:p_iOrderLine>0</bes:p_iOrderLine>
      <bes:p_bIncludeOutOfPriceMarket>false</bes:p_bIncludeOutOfPriceMarket>
      <bes:p_bIncludeZeroFreightCost>false</bes:p_bIncludeZeroFreightCost>
      <bes:p_bIncludeUnbrandedSupplier>false</bes:p_bIncludeUnbrandedSupplier>
    </bes:GetBestBuy>
  </soapenv:Body>
```

</soapenv:Envelope>

SAMPLE RESPONSE

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

<soap:Body>

<GetBestBuyResponse xmlns="http://www.addsys.com/WDMS/POS/BestBuy">

<best_buy_root>

<best_buys>

<best_buy>

<mkt_desc>North west Canada</mkt_desc>

<hauler_code>HAU3</hauler_code>

<hauler_office_id>0</hauler_office_id>

<hauler_office_desc></hauler_office_desc>

<supplier>SVC</supplier>

<supplier_desc>SERVICE PARTS</supplier_desc>

<supply_location>MNT</supply_location>

<supply_location_desc>Montreal</supply_location_desc>

<totals>

<quantity>4200</quantity>

<price>1.3852380952380952</price>

<unit_freight>0.04260000000000000000000000000000</unit_freight>

<unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>

<total_unit_cost>1.42783800000000000000000000000000</total_unit_cost>

<total_cost>5996.920000</total_cost>

<difference>0</difference>

</totals>

<order_lines>

<order_line>

<quantity>1000</quantity>

<price>1.1</price>

<unit_freight>0.04260000000000000000000000000000</unit_freight>

<unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>

<total_unit_cost>1.14260000000000000000000000000000</total_unit_cost>

<total_cost>1142.600000</total_cost>

<difference>0</difference>

<mfg_code>LIQ</mfg_code>

<group_code>DISTIL</group_code>

<stock_id>1</stock_id>

<product_desc>KEROSENE - LIGHT OIL # 1</product_desc>

<order_line>1</order_line>

<effective_date>2015-10-07T00:00:00</effective_date>

</order_line>

<order_line>

<quantity>1200</quantity>

<price>1.3199999999999998</price>

<unit_freight>0.04260000000000000000000000000000</unit_freight>

<unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>

<total_unit_cost>1.36260000000000000000000000000000</total_unit_cost>

[illegible]

```

<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.172000000000000000000000</total_unit_cost>
<total_cost>1172.000000</total_cost>
<difference>123.48</difference>
<mfg_code>LIQ</mfg_code>
<group_code>DISTIL</group_code>
<stock_id>1</stock_id>
<product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
<order_line>1</order_line>
<effective_date>2015-10-07T00:00:00</effective_date>
</order_line>
<order_line>
<quantity>1200</quantity>
<price>1.3199999999999998</price>
<unit_freight>0.072000000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.392000000000000000000000</total_unit_cost>
<total_cost>1670.400000</total_cost>
<difference>123.48</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>2</stock_id>
<product_desc>BULK PROPANE</product_desc>
<order_line>2</order_line>
<effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
<quantity>2000</quantity>
<price>1.567</price>
<unit_freight>0.072000000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.639000000000000000000000</total_unit_cost>
<total_cost>3278.000000</total_cost>
<difference>123.48</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>4</stock_id>
<product_desc>PROPANE</product_desc>
<order_line>3</order_line>
<effective_date>2015-10-06T00:00:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
<mkt_desc>North west Canada</mkt_desc>
<hauler_code>HAU9</hauler_code>
<hauler_office_id>0</hauler_office_id>
<hauler_office_desc></hauler_office_desc>
<supplier>SVC</supplier>

```


	<pre> <supplier_desc>SERVICE PARTS</supplier_desc> <supply_location>MNT</supply_location> <supply_location_desc>Montreal</supply_location_desc> <totals> <quantity>4200</quantity> <price>1.3852380952380952</price> <unit_freight>0.158571000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.543810000000000000000000000000</total_unit_cost> <total_cost>6484.000000</total_cost> <difference>487.08</difference> </totals> <order_lines> <order_line> <quantity>1000</quantity> <price>1.1</price> <unit_freight>0.222000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.322000000000000000000000000000</total_unit_cost> <total_cost>1322.000000</total_cost> <difference>487.08</difference> <mfg_code>LIQ</mfg_code> <group_code>DISTIL</group_code> <stock_id>1</stock_id> <product_desc>KEROSENE - LIGHT OIL # 1</product_desc> <order_line>1</order_line> <effective_date>2015-10-07T00:00:00</effective_date> </order_line> <order_line> <quantity>1200</quantity> <price>1.31999999999999998</price> <unit_freight>0.185000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.505000000000000000000000000000</total_unit_cost> <total_cost>1806.000000</total_cost> <difference>487.08</difference> <mfg_code>LIQ</mfg_code> <group_code>PROP</group_code> <stock_id>2</stock_id> <product_desc>BULK PROPANE</product_desc> <order_line>2</order_line> <effective_date>2016-08-04T00:01:00</effective_date> </order_line> <order_line> <quantity>2000</quantity> <price>1.567</price> <unit_freight>0.111000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.678000000000000000000000000000</total_unit_cost> </pre>

```

<total_cost>3356.000000</total_cost>
<difference>487.08</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>4</stock_id>
<product_desc>PROPANE</product_desc>
<order_line>3</order_line>
<effective_date>2015-10-06T00:00:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU3</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>LIQ</supplier>
  <supplier_desc>LIQUID PRODUCTS</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
</best_buy>
<totals>
  <quantity>4200</quantity>
  <price>1.5143809523809524</price>
  <unit_freight>0.04260000000000000000000000000000</unit_freight>
  <unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.55698100000000000000000000000000</total_unit_cost>
  <total_cost>6539.320000</total_cost>
  <difference>542.4</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.55</price>
    <unit_freight>0.04260000000000000000000000000000</unit_freight>
    <unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.59260000000000000000000000000000</total_unit_cost>
    <total_cost>1592.600000</total_cost>
    <difference>542.4</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-11-23T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>
    <price>1.342</price>
    <unit_freight>0.04260000000000000000000000000000</unit_freight>

```

```

<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.384600000000000000000000</total_unit_cost>
<total_cost>1661.520000</total_cost>
<difference>542.4</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>2</stock_id>
<product_desc>BULK PROPANE</product_desc>
<order_line>2</order_line>
<effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
<quantity>2000</quantity>
<price>1.6</price>
<unit_freight>0.042600000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.642600000000000000000000</total_unit_cost>
<total_cost>3285.200000</total_cost>
<difference>542.4</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>4</stock_id>
<product_desc>PROPANE</product_desc>
<order_line>3</order_line>
<effective_date>2012-11-05T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
<mkt_desc>North west Canada</mkt_desc>
<hauler_code>HAUL</hauler_code>
<hauler_office_id>0</hauler_office_id>
<hauler_office_desc></hauler_office_desc>
<supplier>SVC</supplier>
<supplier_desc>SERVICE PARTS</supplier_desc>
<supply_location>MNT</supply_location>
<supply_location_desc>Montreal</supply_location_desc>
<totals>
<quantity>4200</quantity>
<price>1.3852380952380952</price>
<unit_freight>0.197619000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.582857000000000000000000</total_unit_cost>
<total_cost>6648.000000</total_cost>
<difference>651.08</difference>
</totals>
<order_lines>
<order_line>
<quantity>1000</quantity>

```

```

<price>1.1</price>
<unit_freight>0.23000000000000000000000000</unit_freight>
<unit_add_ons>0.00000000000000000000000000</unit_add_ons>
<total_unit_cost>1.33000000000000000000000000</total_unit_cost>
<total_cost>1330.000000</total_cost>
<difference>651.08</difference>
<mfg_code>LIQ</mfg_code>
<group_code>DISTIL</group_code>
<stock_id>1</stock_id>
<product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
<order_line>1</order_line>
<effective_date>2015-10-07T00:00:00</effective_date>
</order_line>
<order_line>
  <quantity>1200</quantity>
  <price>1.31999999999999999999999999</price>
  <unit_freight>0.21666700000000000000000000</unit_freight>
  <unit_add_ons>0.00000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.53666700000000000000000000</total_unit_cost>
  <total_cost>1844.000000</total_cost>
  <difference>651.08</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>2</stock_id>
  <product_desc>BULK PROPANE</product_desc>
  <order_line>2</order_line>
  <effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>2000</quantity>
  <price>1.567</price>
  <unit_freight>0.17000000000000000000000000</unit_freight>
  <unit_add_ons>0.00000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.73700000000000000000000000</total_unit_cost>
  <total_cost>3474.000000</total_cost>
  <difference>651.08</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2015-10-06T00:00:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU6</hauler_code>
  <hauler_office_id>0</hauler_office_id>

```

```

<hauler_office_desc></hauler_office_desc>
<supplier>LIQ</supplier>
<supplier_desc>LIQUID PRODUCTS</supplier_desc>
<supply_location>MNT</supply_location>
<supply_location_desc>Montreal</supply_location_desc>
<totals>
  <quantity>4200</quantity>
  <price>1.5143809523809524</price>
  <unit_freight>0.0720000000000000000000000000000000</unit_freight>
  <unit_add_ons>0.0000000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.5863810000000000000000000000000000</total_unit_cost>
  <total_cost>6662.800000</total_cost>
  <difference>665.88</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.55</price>
    <unit_freight>0.0720000000000000000000000000000000</unit_freight>
    <unit_add_ons>0.0000000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.6220000000000000000000000000000000</total_unit_cost>
    <total_cost>1622.000000</total_cost>
    <difference>665.88</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-11-23T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>
    <price>1.342</price>
    <unit_freight>0.0720000000000000000000000000000000</unit_freight>
    <unit_add_ons>0.0000000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.4140000000000000000000000000000000</total_unit_cost>
    <total_cost>1696.800000</total_cost>
    <difference>665.88</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>2</stock_id>
    <product_desc>BULK PROPANE</product_desc>
    <order_line>2</order_line>
    <effective_date>2016-08-04T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>2000</quantity>
    <price>1.6</price>
    <unit_freight>0.0720000000000000000000000000000000</unit_freight>

```

```

<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.672000000000000000000000</total_unit_cost>
<total_cost>3344.000000</total_cost>
<difference>665.88</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>4</stock_id>
<product_desc>PROPANE</product_desc>
<order_line>3</order_line>
<effective_date>2012-11-05T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU9</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>LIQ</supplier>
  <supplier_desc>LIQUID PRODUCTS</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
</best_buy>
<totals>
  <quantity>4200</quantity>
  <price>1.5143809523809524</price>
  <unit_freight>0.158571000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.672952000000000000000000</total_unit_cost>
  <total_cost>7026.400000</total_cost>
  <difference>1029.48</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.55</price>
    <unit_freight>0.222000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.772000000000000000000000</total_unit_cost>
    <total_cost>1772.000000</total_cost>
    <difference>1029.48</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-11-23T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>

```

```

<price>1.342</price>
<unit_freight>0.185000000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.527000000000000000000000</total_unit_cost>
<total_cost>1832.400000</total_cost>
<difference>1029.48</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>2</stock_id>
<product_desc>BULK PROPANE</product_desc>
<order_line>2</order_line>
<effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>2000</quantity>
  <price>1.6</price>
  <unit_freight>0.111000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.711000000000000000000000</total_unit_cost>
  <total_cost>3422.000000</total_cost>
  <difference>1029.48</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2012-11-05T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU3</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>KAMP</supplier>
  <supplier_desc>KAMS PROPANE</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
<totals>
  <quantity>4200</quantity>
  <price>1.6339142857142859</price>
  <unit_freight>0.042600000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.676514000000000000000000</total_unit_cost>
  <total_cost>7041.360000</total_cost>
  <difference>1044.44</difference>
</totals>
<order_lines>

```

```
<order_line>
  <quantity>1000</quantity>
  <price>1.234</price>
  <unit_freight>0.042600000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.276600000000000000000000</total_unit_cost>
  <total_cost>1276.600000</total_cost>
  <difference>1044.44</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>DISTIL</group_code>
  <stock_id>1</stock_id>
  <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
  <order_line>1</order_line>
  <effective_date>2016-03-24T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>1200</quantity>
  <price>1.7237</price>
  <unit_freight>0.042600000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.766300000000000000000000</total_unit_cost>
  <total_cost>2119.560000</total_cost>
  <difference>1044.44</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>2</stock_id>
  <product_desc>BULK PROPANE</product_desc>
  <order_line>2</order_line>
  <effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>2000</quantity>
  <price>1.78</price>
  <unit_freight>0.042600000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.822600000000000000000000</total_unit_cost>
  <total_cost>3645.200000</total_cost>
  <difference>1044.44</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2012-06-10T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
```


	<pre> <hauler_code>HAU6</hauler_code> <hauler_office_id>0</hauler_office_id> <hauler_office_desc></hauler_office_desc> <supplier>KAMP</supplier> <supplier_desc>KAMS PROPANE</supplier_desc> <supply_location>MNT</supply_location> <supply_location_desc>Montreal</supply_location_desc> <totals> <quantity>4200</quantity> <price>1.6339142857142859</price> <unit_freight>0.072000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.705914000000000000000000000000</total_unit_cost> <total_cost>7164.840000</total_cost> <difference>1167.92</difference> </totals> <order_lines> <order_line> <quantity>1000</quantity> <price>1.234</price> <unit_freight>0.072000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.306000000000000000000000000000</total_unit_cost> <total_cost>1306.000000</total_cost> <difference>1167.92</difference> <mfg_code>LIQ</mfg_code> <group_code>DISTIL</group_code> <stock_id>1</stock_id> <product_desc>KEROSENE - LIGHT OIL # 1</product_desc> <order_line>1</order_line> <effective_date>2016-03-24T00:01:00</effective_date> </order_line> <order_line> <quantity>1200</quantity> <price>1.7237</price> <unit_freight>0.072000000000000000000000000000</unit_freight> <unit_add_ons>0.000000000000000000000000000000</unit_add_ons> <total_unit_cost>1.795700000000000000000000000000</total_unit_cost> <total_cost>2154.840000</total_cost> <difference>1167.92</difference> <mfg_code>LIQ</mfg_code> <group_code>PROP</group_code> <stock_id>2</stock_id> <product_desc>BULK PROPANE</product_desc> <order_line>2</order_line> <effective_date>2016-08-04T00:01:00</effective_date> </order_line> <order_line> <quantity>2000</quantity> </pre>

```

<price>1.78</price>
<unit_freight>0.072000000000000000000000</unit_freight>
<unit_add_ons>0.000000000000000000000000</unit_add_ons>
<total_unit_cost>1.852000000000000000000000</total_unit_cost>
<total_cost>3704.000000</total_cost>
<difference>1167.92</difference>
<mfg_code>LIQ</mfg_code>
<group_code>PROP</group_code>
<stock_id>4</stock_id>
<product_desc>PROPANE</product_desc>
<order_line>3</order_line>
<effective_date>2012-06-10T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAUL</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>LIQ</supplier>
  <supplier_desc>LIQUID PRODUCTS</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
</best_buy>
<totals>
  <quantity>4200</quantity>
  <price>1.5143809523809524</price>
  <unit_freight>0.197619000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.712000000000000000000000</total_unit_cost>
  <total_cost>7190.400000</total_cost>
  <difference>1193.48</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.55</price>
    <unit_freight>0.230000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.780000000000000000000000</total_unit_cost>
    <total_cost>1780.000000</total_cost>
    <difference>1193.48</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-11-23T00:01:00</effective_date>
  </order_line>

```

```

<order_line>
  <quantity>1200</quantity>
  <price>1.342</price>
  <unit_freight>0.216667000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.558667000000000000000000000000</total_unit_cost>
  <total_cost>1870.400000</total_cost>
  <difference>1193.48</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>2</stock_id>
  <product_desc>BULK PROPANE</product_desc>
  <order_line>2</order_line>
  <effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>2000</quantity>
  <price>1.6</price>
  <unit_freight>0.170000000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.770000000000000000000000000000</total_unit_cost>
  <total_cost>3540.000000</total_cost>
  <difference>1193.48</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2012-11-05T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU9</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>KAMP</supplier>
  <supplier_desc>KAMS PROPANE</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
<totals>
  <quantity>4200</quantity>
  <price>1.6339142857142859</price>
  <unit_freight>0.158571000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.792486000000000000000000000000</total_unit_cost>
  <total_cost>7528.440000</total_cost>
  <difference>1531.52</difference>

```

```
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.234</price>
    <unit_freight>0.22200000000000000000000000000000</unit_freight>
    <unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.45600000000000000000000000000000</total_unit_cost>
    <total_cost>1456.000000</total_cost>
    <difference>1531.52</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-03-24T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>
    <price>1.7237</price>
    <unit_freight>0.18500000000000000000000000000000</unit_freight>
    <unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.90870000000000000000000000000000</total_unit_cost>
    <total_cost>2290.440000</total_cost>
    <difference>1531.52</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>2</stock_id>
    <product_desc>BULK PROPANE</product_desc>
    <order_line>2</order_line>
    <effective_date>2016-08-04T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>2000</quantity>
    <price>1.78</price>
    <unit_freight>0.11100000000000000000000000000000</unit_freight>
    <unit_add_ons>0.00000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.89100000000000000000000000000000</total_unit_cost>
    <total_cost>3782.000000</total_cost>
    <difference>1531.52</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>4</stock_id>
    <product_desc>PROPANE</product_desc>
    <order_line>3</order_line>
    <effective_date>2012-06-10T00:01:00</effective_date>
  </order_line>
</order_lines>
</best_buy>
```

```

<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAUL</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>KAMP</supplier>
  <supplier_desc>KAMS PROPANE</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
  <totals>
    <quantity>4200</quantity>
    <price>1.6339142857142859</price>
    <unit_freight>0.197619000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.831533000000000000000000000000</total_unit_cost>
    <total_cost>7692.440000</total_cost>
    <difference>1695.52</difference>
  </totals>
  <order_lines>
    <order_line>
      <quantity>1000</quantity>
      <price>1.234</price>
      <unit_freight>0.230000000000000000000000000000</unit_freight>
      <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
      <total_unit_cost>1.464000000000000000000000000000</total_unit_cost>
      <total_cost>1464.000000</total_cost>
      <difference>1695.52</difference>
      <mfg_code>LIQ</mfg_code>
      <group_code>DISTIL</group_code>
      <stock_id>1</stock_id>
      <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
      <order_line>1</order_line>
      <effective_date>2016-03-24T00:01:00</effective_date>
    </order_line>
    <order_line>
      <quantity>1200</quantity>
      <price>1.7237</price>
      <unit_freight>0.216667000000000000000000000000</unit_freight>
      <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
      <total_unit_cost>1.940367000000000000000000000000</total_unit_cost>
      <total_cost>2328.440000</total_cost>
      <difference>1695.52</difference>
      <mfg_code>LIQ</mfg_code>
      <group_code>PROP</group_code>
      <stock_id>2</stock_id>
      <product_desc>BULK PROPANE</product_desc>
      <order_line>2</order_line>
      <effective_date>2016-08-04T00:01:00</effective_date>
    </order_line>
  </order_lines>

```

```
<order_line>
  <quantity>2000</quantity>
  <price>1.78</price>
  <unit_freight>0.17000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.950000000000000000000000</total_unit_cost>
  <total_cost>3900.000000</total_cost>
  <difference>1695.52</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2012-06-10T00:01:00</effective_date>
</order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU3</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>MOB</supplier>
  <supplier_desc>MOBILE OIL</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
<totals>
  <quantity>4200</quantity>
  <price>1.8034380952380953</price>
  <unit_freight>0.04260000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.846038000000000000000000</total_unit_cost>
  <total_cost>7753.360000</total_cost>
  <difference>1756.44</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.6</price>
    <unit_freight>0.04260000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.642600000000000000000000</total_unit_cost>
    <total_cost>1642.600000</total_cost>
    <difference>1756.44</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
```

```

    <effective_date>2016-08-24T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>
    <price>1.8337</price>
    <unit_freight>0.042600000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.876300000000000000000000000000</total_unit_cost>
    <total_cost>2251.560000</total_cost>
    <difference>1756.44</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>2</stock_id>
    <product_desc>BULK PROPANE</product_desc>
    <order_line>2</order_line>
    <effective_date>2016-08-04T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>2000</quantity>
    <price>1.887</price>
    <unit_freight>0.042600000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.929600000000000000000000000000</total_unit_cost>
    <total_cost>3859.200000</total_cost>
    <difference>1756.44</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>4</stock_id>
    <product_desc>PROPANE</product_desc>
    <order_line>3</order_line>
    <effective_date>2015-10-06T00:00:00</effective_date>
  </order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU6</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>MOB</supplier>
  <supplier_desc>MOBILE OIL</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
<totals>
  <quantity>4200</quantity>
  <price>1.8034380952380953</price>
  <unit_freight>0.072000000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>1.875438000000000000000000000000</total_unit_cost>

```

```
<total_cost>7876.840000</total_cost>
<difference>1879.92</difference>
</totals>
<order_lines>
  <order_line>
    <quantity>1000</quantity>
    <price>1.6</price>
    <unit_freight>0.07200000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.672000000000000000000000</total_unit_cost>
    <total_cost>1672.000000</total_cost>
    <difference>1879.92</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>DISTIL</group_code>
    <stock_id>1</stock_id>
    <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
    <order_line>1</order_line>
    <effective_date>2016-08-24T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>1200</quantity>
    <price>1.8337</price>
    <unit_freight>0.07200000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.905700000000000000000000</total_unit_cost>
    <total_cost>2286.840000</total_cost>
    <difference>1879.92</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>2</stock_id>
    <product_desc>BULK PROPANE</product_desc>
    <order_line>2</order_line>
    <effective_date>2016-08-04T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>2000</quantity>
    <price>1.887</price>
    <unit_freight>0.07200000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.959000000000000000000000</total_unit_cost>
    <total_cost>3918.000000</total_cost>
    <difference>1879.92</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>4</stock_id>
    <product_desc>PROPANE</product_desc>
    <order_line>3</order_line>
    <effective_date>2015-10-06T00:00:00</effective_date>
  </order_line>
```



```

</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAU9</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>MOB</supplier>
  <supplier_desc>MOBILE OIL</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
  <totals>
    <quantity>4200</quantity>
    <price>1.8034380952380953</price>
    <unit_freight>0.158571000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.962010000000000000000000000000</total_unit_cost>
    <total_cost>8240.440000</total_cost>
    <difference>2243.52</difference>
  </totals>
  <order_lines>
    <order_line>
      <quantity>1000</quantity>
      <price>1.6</price>
      <unit_freight>0.222000000000000000000000000000</unit_freight>
      <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
      <total_unit_cost>1.822000000000000000000000000000</total_unit_cost>
      <total_cost>1822.000000</total_cost>
      <difference>2243.52</difference>
      <mfg_code>LIQ</mfg_code>
      <group_code>DISTIL</group_code>
      <stock_id>1</stock_id>
      <product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
      <order_line>1</order_line>
      <effective_date>2016-08-24T00:01:00</effective_date>
    </order_line>
    <order_line>
      <quantity>1200</quantity>
      <price>1.8337</price>
      <unit_freight>0.185000000000000000000000000000</unit_freight>
      <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
      <total_unit_cost>2.018700000000000000000000000000</total_unit_cost>
      <total_cost>2422.440000</total_cost>
      <difference>2243.52</difference>
      <mfg_code>LIQ</mfg_code>
      <group_code>PROP</group_code>
      <stock_id>2</stock_id>
      <product_desc>BULK PROPANE</product_desc>
      <order_line>2</order_line>

```

```

    <effective_date>2016-08-04T00:01:00</effective_date>
  </order_line>
  <order_line>
    <quantity>2000</quantity>
    <price>1.887</price>
    <unit_freight>0.111000000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>1.998000000000000000000000000000</total_unit_cost>
    <total_cost>3996.000000</total_cost>
    <difference>2243.52</difference>
    <mfg_code>LIQ</mfg_code>
    <group_code>PROP</group_code>
    <stock_id>4</stock_id>
    <product_desc>PROPANE</product_desc>
    <order_line>3</order_line>
    <effective_date>2015-10-06T00:00:00</effective_date>
  </order_line>
</order_lines>
</best_buy>
<best_buy>
  <mkt_desc>North west Canada</mkt_desc>
  <hauler_code>HAUL</hauler_code>
  <hauler_office_id>0</hauler_office_id>
  <hauler_office_desc></hauler_office_desc>
  <supplier>MOB</supplier>
  <supplier_desc>MOBILE OIL</supplier_desc>
  <supply_location>MNT</supply_location>
  <supply_location_desc>Montreal</supply_location_desc>
  <totals>
    <quantity>4200</quantity>
    <price>1.8034380952380953</price>
    <unit_freight>0.197619000000000000000000000000</unit_freight>
    <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
    <total_unit_cost>2.001057000000000000000000000000</total_unit_cost>
    <total_cost>8404.440000</total_cost>
    <difference>2407.52</difference>
  </totals>
  <order_lines>
    <order_line>
      <quantity>1000</quantity>
      <price>1.6</price>
      <unit_freight>0.230000000000000000000000000000</unit_freight>
      <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
      <total_unit_cost>1.830000000000000000000000000000</total_unit_cost>
      <total_cost>1830.000000</total_cost>
      <difference>2407.52</difference>
      <mfg_code>LIQ</mfg_code>
      <group_code>DISTIL</group_code>
      <stock_id>1</stock_id>

```

```
<product_desc>KEROSENE - LIGHT OIL # 1</product_desc>
<order_line>1</order_line>
<effective_date>2016-08-24T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>1200</quantity>
  <price>1.8337</price>
  <unit_freight>0.216667000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>2.050367000000000000000000000000</total_unit_cost>
  <total_cost>2460.440000</total_cost>
  <difference>2407.52</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>2</stock_id>
  <product_desc>BULK PROPANE</product_desc>
  <order_line>2</order_line>
  <effective_date>2016-08-04T00:01:00</effective_date>
</order_line>
<order_line>
  <quantity>2000</quantity>
  <price>1.887</price>
  <unit_freight>0.170000000000000000000000000000</unit_freight>
  <unit_add_ons>0.000000000000000000000000000000</unit_add_ons>
  <total_unit_cost>2.057000000000000000000000000000</total_unit_cost>
  <total_cost>4114.000000</total_cost>
  <difference>2407.52</difference>
  <mfg_code>LIQ</mfg_code>
  <group_code>PROP</group_code>
  <stock_id>4</stock_id>
  <product_desc>PROPANE</product_desc>
  <order_line>3</order_line>
  <effective_date>2015-10-06T00:00:00</effective_date>
</order_line>
</order_lines>
</best_buy>
</best_buys>
</best_buy_root>
</GetBestBuyResponse>
</soap:Body>
</soap:Envelope>
```

PriceQuote class

- **License Required: SmartConnect API WDMS**

Overview

***** This class must be used for databases whose E3 release is 12.x.**

The PriceQuote class provides an interface for price quote related operations.

URL: `http://<WEBSERVICE>/WDMS/PriceQuote/PriceQuote.asmx`

This section describes the methods that will enable the programmer to get price quote information from E3.

- `GetPriceQuotes ()`: Gets price quotes information

PriceQuote. GetPriceQuotes()

Description

This method will retrieve price quote information based on the selection criteria.

SOAP 1.2

The following is a sample SOAP 1.2 request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /WDMS/PriceQuote/PriceQuote.asmx HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetPriceQuotes xmlns="http://www.addsys.com/WDMS/PriceQuote">
      <p_iCompanyID>short</p_iCompanyID>
      <p_sSysUser>string</p_sSysUser>
      <p_cSelCrit>
        <MaxOfDays>short</MaxOfDays>
        <EffectiveDate>dateTime</EffectiveDate>
        <GroupCode>
          <string>string</string>
          <string>string</string>
        </GroupCode>
        <MfgCode>
          <string>string</string>
          <string>string</string>
        </MfgCode>
        <StockId>
          <string>string</string>
          <string>string</string>
        </StockId>
        <AllProducts>boolean</AllProducts>
        <SupplyLocations>
          <string>string</string>
          <string>string</string>
        </SupplyLocations>
        <Suppliers>
          <string>string</string>
          <string>string</string>
        </Suppliers>
        <LocationNum>
          <short>short</short>
          <short>short</short>
        </LocationNum>
        <AccountNum>int</AccountNum>
        <RowStatus>string</RowStatus>
      </p_cSelCrit>
    </GetPriceQuotes>
  </soap12:Body>
```

```

</soap12:Envelope>

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

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetPriceQuotesResponse xmlns="http://www.addsys.com/WDMS/PriceQuote">
      <PriceQuotesRoot>
        <PriceQuotes>
          <PriceQuote>
            <QuoteId>int</QuoteId>
            <Status>string</Status>
            <ContractId>int</ContractId>
            <TotalOrders>int</TotalOrders>
            <LocationNum>short</LocationNum>
            <LocationName>string</LocationName>
            <MfgCode>string</MfgCode>
            <GroupCode>string</GroupCode>
            <StockId>string</StockId>
            <ProductDescription>string</ProductDescription>
            <EffectiveDate>dateTime</EffectiveDate>
            <ExpiryDate>dateTime</ExpiryDate>
            <SupplierCode>string</SupplierCode>
            <SupplierName>string</SupplierName>
            <LocationCode>string</LocationCode>
            <LocationDescription>string</LocationDescription>
            <ProductUnitPrice>decimal</ProductUnitPrice>
            <AddonUnitPrice>decimal</AddonUnitPrice>
            <FreightUnitPrice>decimal</FreightUnitPrice>
            <TaxUnitPrice>decimal</TaxUnitPrice>
            <QuotedPrice>decimal</QuotedPrice>
            <NoContractPrice>decimal</NoContractPrice>
            <LoadFeeUnitPrice>decimal</LoadFeeUnitPrice>
          </PriceQuote>
          <PriceQuote>
            <QuoteId>int</QuoteId>
            <Status>string</Status>
            <ContractId>int</ContractId>
            <TotalOrders>int</TotalOrders>
            <LocationNum>short</LocationNum>
            <LocationName>string</LocationName>
            <MfgCode>string</MfgCode>
            <GroupCode>string</GroupCode>
            <StockId>string</StockId>
            <ProductDescription>string</ProductDescription>
            <EffectiveDate>dateTime</EffectiveDate>
            <ExpiryDate>dateTime</ExpiryDate>
            <SupplierCode>string</SupplierCode>
            <SupplierName>string</SupplierName>
            <LocationCode>string</LocationCode>
            <LocationDescription>string</LocationDescription>
            <ProductUnitPrice>decimal</ProductUnitPrice>
            <AddonUnitPrice>decimal</AddonUnitPrice>
            <FreightUnitPrice>decimal</FreightUnitPrice>
            <TaxUnitPrice>decimal</TaxUnitPrice>
            <QuotedPrice>decimal</QuotedPrice>
            <NoContractPrice>decimal</NoContractPrice>
            <LoadFeeUnitPrice>decimal</LoadFeeUnitPrice>
          </PriceQuote>
        </PriceQuotes>
        <Exception>
          <ErrorNumber>int</ErrorNumber>
          <ErrorMessage>string</ErrorMessage>
        </Exception>
      </PriceQuotesRoot>
    </GetPriceQuotesResponse>
  </soap12:Body>
</soap12:Envelope>

```

REQUEST PARAMETERS

short p_iCompanyID	Company id
string p_sSysUser	System user id
class p_cSelCrit	See class definition below

Class Name: p_cSelCrit	
short MaxOfDays	The number of days to add to the EffectiveDate value to derive a time frame.
dateTime EffectiveDate	Effective date of the quote
string[] GroupCode	String array representing group codes. If AllProducts = true or if MfgCode and/or StockId are empty, the value will be ignored and will be considered empty.

string[] MfgCode	String array representing mfg codes. If AllProducts = true or if GroupCode and/or StockId are empty, the value will be ignored and will be considered empty.
string[] StockId	String array representing stock ids. If AllProducts = true or if GroupCode and/or MfgCode are empty, the value will be ignored and will be considered empty.
boolean AllProducts	If true, all products will be returned. GroupCode, MfgCode and StockId values will be ignored if true.
string[] SupplyLocations	String array representing supply location codes.
string[] Suppliers	String array representing supplier codes.
short[] LocationNum	short array representing location numbers.
int AccountNum	The customer account number
string RowStatus	The status of the quote record in the database. "A" = Active, "I" = InActive, "" = Both

RESPONSE CLASSES:

Class Name: PriceQuotesRoot	
PriceQuote[] PriceQuotes	See class definition below
class Exception	See class definition below

Class Name: PriceQuote	
int QuoteId	Quote Id
string Status	Row status in the database
int ContractId	Contract id
int TotalOrders	Number of orders linked to the quote
short LocationNum	Location number
string LocationName	Location name
string MfgCode	Mfg code
string GroupCode	Group code
string StockId	Stock id
string ProductDescription	Product description
dateTime EffectiveDate	Effective date and time of quote
dateTime ExpiryDate	Expiry date and time of quote
string SupplierCode	Supplier code
string SupplierName	Supplier name
string LocationCode	Location code
string LocationDescription	Location description
decimal ProductUnitPrice	Product unit price
decimal AddonUnitPrice	Addon unit price
decimal FreightUnitPrice	Freight unit price
decimal TaxUnitPrice	Tax unit price
decimal QuotedPrice	Quoted price
decimal NoContractPrice	No contract price
decimal LoadFeeUnitPrice	Load fee unit price

Class Name: Exception	
int ErrorNumber	Error number if applicable
string ErrorMessage	Error message

.XML Documents:

Example of a successful request and response	SAMPLE REQUEST
	<pre><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:pric="http://www.addsys.com/WDMS/PriceQuote"> <soapenv:Header/> <soapenv:Body> <pric:GetPriceQuotes></pre>

```

<pric:p_iCompanyID>1600</pric:p_iCompanyID>
<pric:p_sSysUser>WebUser</pric:p_sSysUser>
<pric:p_cSelCrit>
  <pric:MaxOfDays>5</pric:MaxOfDays>
  <pric:EffectiveDate>2017-03-20T18:00:00</pric:EffectiveDate>
  <pric:GroupCode>
    <pric:string>DISTIL</pric:string>
  </pric:GroupCode>
  <pric:MfgCode >
    <pric:string>LIQ</pric:string>
  </pric:MfgCode>
  <pric:StockId >
    <pric:string>2</pric:string>
  </pric:StockId>
  <pric:AllProducts>false</pric:AllProducts>
  <pric:SupplyLocations >
    <pric:string>NYC</pric:string>
  </pric:SupplyLocations>
  <pric:Suppliers>
    <pric:string>CITG</pric:string>
  </pric:Suppliers>
  <pric:LocationNum>
    <pric:short>12</pric:short>
    <pric:short>17</pric:short>
  </pric:LocationNum>
  <pric:AccountNum>100504</pric:AccountNum>
  <pric:RowStatus>A</pric:RowStatus>
</pric:p_cSelCrit>
</pric:GetPriceQuotes>
</soapenv:Body>
</soapenv:Envelope>

```

SAMPLE RESPONSE

```

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <GetPriceQuotesResponse
xmlns="http://www.addsys.com/WDMS/PriceQuote">
      <PriceQuotesRoot>
        <PriceQuotes>
          <PriceQuote>
            <QuotId>198800</QuotId>
            <Status>A</Status>
            <ContractId>0</ContractId>
            <TotalOrders>0</TotalOrders>
            <LocationNum>17</LocationNum>
            <LocationName>Richard location 17</LocationName>
            <MfgCode>LIQ</MfgCode>
            <GroupCode>DISTIL</GroupCode>

```

```

        <StockId>2</StockId>
        <ProductDescription>HEATING OIL</ProductDescription>
        <EffectiveDate>2017-03-20T18:00:00</EffectiveDate>
        <ExpiryDate>2017-03-21T17:59:00</ExpiryDate>
        <SupplierCode>CITG</SupplierCode>
        <SupplierName>CITGO PETROLEUM</SupplierName>
        <LocationCode>NYC</LocationCode>
        <LocationDescription>New York Terminal</LocationDescription>
        <ProductUnitPrice>1.960000</ProductUnitPrice>
        <AddonUnitPrice>0.000000</AddonUnitPrice>
        <FreightUnitPrice>0.032100</FreightUnitPrice>
        <TaxUnitPrice>0.842305</TaxUnitPrice>
        <QuotedPrice>2.834405</QuotedPrice>
        <NoContractPrice>0.000000</NoContractPrice>
        <LoadFeeUnitPrice>0.000000</LoadFeeUnitPrice>
    </PriceQuote>
    <PriceQuote>
        <QuotId>198799</QuotId>
        <Status>A</Status>
        <ContractId>0</ContractId>
        <TotalOrders>0</TotalOrders>
        <LocationNum>12</LocationNum>
        <LocationName>Richard location 12</LocationName>
        <MfgCode>LIQ</MfgCode>
        <GroupCode>DISTIL</GroupCode>
        <StockId>2</StockId>
        <ProductDescription>HEATING OIL</ProductDescription>
        <EffectiveDate>2017-03-20T18:00:00</EffectiveDate>
        <ExpiryDate>2017-03-21T17:59:00</ExpiryDate>
        <SupplierCode>CITG</SupplierCode>
        <SupplierName>CITGO PETROLEUM</SupplierName>
        <LocationCode>NYC</LocationCode>
        <LocationDescription>New York Terminal</LocationDescription>
        <ProductUnitPrice>2.199600</ProductUnitPrice>
        <AddonUnitPrice>0.000000</AddonUnitPrice>
        <FreightUnitPrice>0.030000</FreightUnitPrice>
        <TaxUnitPrice>0.866501</TaxUnitPrice>
        <QuotedPrice>3.096101</QuotedPrice>
        <NoContractPrice>0.000000</NoContractPrice>
        <LoadFeeUnitPrice>0.000000</LoadFeeUnitPrice>
    </PriceQuote>
</PriceQuotes>
</PriceQuotesRoot>
</GetPriceQuotesResponse>
</soap:Body>
</soap:Envelope>

```


CreditAction Class

Overview

This section describes the methods that will enable the programmer to create credit actions on accounts, terminate and reinstate accounts.

CreditAction.LoadAcquisitionReasonCodes()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to retrieve the list of valid acquisition reason codes.

```
public DCEnergyAPI.Credit.AcquisitionReasonCode[] LoadAcquisitionReasonCodes(
    short pCompanyID,
    string pSysUser,
    ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Int (32 bit integer)	0:success, <>0: error code
Return Values			
		DCEnergyAPI.Credit.AcquisitionReasonCode[]	Array of type AcquisitionReasonCode .

DCEnergyAPI.Credit.AcquisitionReasonCode

field name	Description
short AcqReasonCode	Numeric acquisition reason code
string Description	Description of acquisition reason code.

C# Example

```
private void btnAcqReasonCode_Click(object sender, EventArgs e)
{
    DCEnergyAPI.Credit.AcquisitionReasonCode[] cacs = new
DCEnergyAPI.Credit.AcquisitionReasonCode[] { };
    EnergyAPI.Credit.CreditAction creditaction = new EnergyAPI.Credit.CreditAction();
    int errorCode = 0;
    short companyID = 2;
    string userID = "WebUser";

    cacs = creditaction.LoadAcquisitionReasonCodes(companyID, userID, ref errorCode);
    if (errorCode == 0)
    {
        // success
    }
    else
    {
        // retrieve message associated with the error code.
    }
}
```

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:LoadAcquisitionReasonCodes>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadAcquisitionReasonCodes>
  </soapenv:Body>
</soapenv:Envelope>
```

CreditAction.LoadCreditActionCodes()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to retrieve the list of valid credit action codes.

```
public DCEnergyAPI.Credit.ActionCode[] LoadCreditActionCodes(
    short pCompanyID,
    string pSysUser,
    ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Int (32 bit integer)	0:success, <>0: error code
Return Values			
		DCEnergyAPI.Credit.ActionCode []	Array of type ActionCode .

DCEnergyAPI.Credit.ActionCode

field name	Description
short Code	Numeric action code
string Description	Description of action code.
short PostingCode	Posting code associated with the action code. This value will be used for the TRANS_MAIN transaction that is created by the process.
short DefaultDays	Default suggested days associated with the action code. This value (days) + "today" = the default <i>suggested</i> date for either the reminder date/release until date.

C# Example

```
private void btnLoadCACs_Click(object sender, EventArgs e)
{
    DCEnergyAPI.Credit.ActionCode[] cacs = new DCEnergyAPI.Credit.ActionCode[] { };
    EnergyAPI.Credit.CreditAction creditaction = new EnergyAPI.Credit.CreditAction();
    int errorCode = 0;
    short companyID = 2;
    string userID = "Webuser";
    cacs = creditaction.LoadCreditActionCodes(companyID, userID, ref errorCode);
    if (errorCode == 0)
    {
        // success
    }
    else
    {
        // retrieve message associated with the error code.
    }
}
```

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:LoadCreditActionCodes>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadCreditActionCodes>
  </soapenv:Body>
</soapenv:Envelope>
```

CreditAction.LoadTerminationCodes()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to retrieve the list of valid termination codes.

```
public DCEnergyAPI.Credit.ActionCode[] LoadTerminationCodes(
    short pCompanyID,
    string pSysUser,
    ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pErrorCode	Int (32 bit integer)	0:success, <>0: error code
Return Values			
		DCEnergyAPI.Credit.ActionCode []	Array of type ActionCode .

DCEnergyAPI.Credit.ActionCode

field name	Description
short Code	Numeric action code
string Description	Description of action code.
short PostingCode	Posting code associated with the action code. This value will be used for the TRANS_MAIN transaction that is created by the process.
short DefaultDays	n/a

C# Example

```
private void btnTermCodes_Click(object sender, EventArgs e)
{
    DCEnergyAPI.Credit.ActionCode[] cacs = new DCEnergyAPI.Credit.ActionCode[] { };
    EnergyAPI.Credit.CreditAction creditaction = new EnergyAPI.Credit.CreditAction();
    int errorCode = 0;
    short companyID = 2;
    string userID = "Webuser";
    cacs = creditaction.LoadTerminationCodes(companyID, userID, ref errorCode);
    if (errorCode == 0)
    {
        // success
    }
    else
    {
        // retrieve message associated with the error code.
    }
}
```

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:LoadTerminationCodes>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadTerminationCodes>
  </soapenv:Body>
</soapenv:Envelope>
```

CreditAction.CreateCreditAction()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to create an action on an account.

```
public int CreateCreditAction(  
    short pCompanyID,  
    string pSysUser,  
    int pAccountNum,  
    short pCreditActionCode,  
    short pPostingCode,  
    DCEnergyAPI.Credit.CreateCreditActionData pCreateCreditActionData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit Integer)	Account # (without checkdigit)
	pCreditActionCode	Short (16 bit integer)	ActionCode.Code (as retrieved by LoadCreditActionCodes)
	pPostingCode	Short (16 bit Integer)	ActionCode.PostingCode (as retrieved by LoadCreditActionCodes)
	pCreateCreditActionData	DCEnergyAPI.Credit.CreateCreditActionData	
Return Values			
	pErrorCode	Int (32 bit integer)	0:success, <>0: error code

DCEnergyAPI.Credit.CreateCreditActionData

field name	Description
date ActionDate ⁴	Enter the action date.
date ReminderOrReleaseUntilDate ⁴	Reminder or Release Until Date
double PromisedAmount	Amount promised by customer
bool DeliveryHold ⁵	True: will set "Delivery Hold" to "Y" False: will set "Delivery Hold" to "N"
bool ServiceHold ⁵	True: will set "Service Hold" to "Y" False: will set "Service Hold" to "N"
bool ClearAfterUpdate	True: If you want the credit action recorded on the customer and then cleared from these fields Credit Action Code (CAC), Credit Action Date (CAD), Reminder Date (RED) and Promised Amount (PRA).
bool MustChargeCreditCard	True: to force the customer's credit card to be pre-authorized for future transactions. The Must Charge Credit Card flag will cause the customer to be put on credit hold, if the card authorization fails or the charge transaction is rejected by the clearing house. This could prevent a delivery or service call from being made, if there is a credit problem. This only applies to pre-authorizations for service work orders or delivery phone orders and only if those options are set up as automatic charge types for the credit card in question.
string LogNote	Add up to 255 characters of comments to this credit action. Comments will be saved in Customer Log
bool RemoveFromReleaseUntil	True: To remove the account from Release Until. This option will only play a role if the account was previously put on Release Until.

⁴: Date will default to "today" if provided date is less than "today" (master file date)

⁵: example usage –

delivery_hold CURRENT VALUE

Y

N

TO CHANGE VALUE

DeliveryHold =false

DeliveryHold =true

TO LEAVE VALUE "AS IS"

DeliveryHold =true

DeliveryHold =false

C# Example

```
private void btnCreditActionCodes_Click(object sender, EventArgs e)
{
    DCEnergyAPI.Credit.CreateCreditActionData cad = new
DCEnergyAPI.Credit.CreateCreditActionData();

    cad.ActionDate = Convert.ToDateTime("05/15/2017");
    cad.LogNote = "THIS IS CREDIT ACTION CODE CREATED VIA API";
    cad.ReminderOrReleaseUntilDate = Convert.ToDateTime("05/31/2017");
    cad.PromisedAmount = Convert.ToDouble(tbPromised.Text);
    EnergyAPI.Credit.CreditAction ca = new EnergyAPI.Credit.CreditAction();
    short companyID = 2;
    string userID = "Webuser";
    int accountNum = 100643;
    short creditActionCode = 11;
    short postingCode = 111;
    int errorcode = ca.CreateCreditAction(companyID, userID,
        accountNum,
        creditActionCode, postingCode, cad);
    if (errorcode == 0)
    {
        // success
    }
    else
    {
        // retrieve message associated with the error code.
    }
}
```

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:CreateCreditAction>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>100643</add:pAccountNum>
      <add:pCreditActionCode>11</add:pCreditActionCode>
      <add:pPostingCode>111</add:pPostingCode>
      <!--Optional:-->
      <add:pCreateCreditActionData>
        <add:ActionDate>2017-05-15</add:ActionDate>
        <add:ReminderOrReleaseUntilDate>2017-05-31</add:ReminderOrReleaseUntilDate>
        <add:PromisedAmount>1.11</add:PromisedAmount>
        <add:DeliveryHold>>false</add:DeliveryHold>
        <add:ServiceHold>>false</add:ServiceHold>
        <add:ClearAfterUpdate>>false</add:ClearAfterUpdate>
        <add:MustChargeCreditCard>>false</add:MustChargeCreditCard>
      </add:pCreateCreditActionData>
    </add:CreateCreditAction>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<!--Optional:-->
<add:LogNote>Action via API</add:LogNote>
<add:RemoveFromReleaseUntil>>false</add:RemoveFromReleaseUntil>
</add:pCreateCreditActionData>
</add:CreateCreditAction>
</soapenv:Body>
</soapenv:Envelope>
```

CreditAction.TerminateAccount()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to terminate an account.

```
public int TerminateAccount(
    short pCompanyID,
    string pSysUser,
    int pAccountNum,
    short pCreditActionCode,
    short pPostingCode,
    DCEnergyAPI.Credit.TerminateAccountData pTerminateAccountData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit Integer)	Account # (without checkdigit)
	pCreditActionCode	Short (16 bit integer)	ActionCode.Code (as retrieved by LoadTerminationCodes)
	pPostingCode	Short (16 bit Integer)	ActionCode.PostingCode (as retrieved by LoadTerminationCodes)
	pTerminateAccountData	DCEnergyAPI.Credit.TerminateAccountData	
Return Values			
	pErrorCode	Integer (32 bit integer)	0:success, <>0: error code

DCEnergyAPI.Credit.TerminateAccountData

field name	Description
date TerminationDate ⁴	Termination Date
date ActionDate ⁴	Enter the action date.
bool RemoveServiceContract	Remove service contract
bool RemoveBudget	Remove from budget
string LogNote	Add up to 255 characters of comments to this credit action. Comments will be saved in Customer Log

⁴: Date will default to “today” if provided date is less than “today” (master file date)

C# Example

```

        private void btnTerminateAccount_Click(object sender, EventArgs e)
        {
            DCEnergyAPI.Credit.TerminateAccountData td = new
DCEnergyAPI.Credit.TerminateAccountData();
            td.CreditActionDate = Convert.ToDateTime ("05/15/2017");
            td.LogNote = "THIS IS THE LOG NOTE - API";
            td.RemoveBudget = true;
            td.RemoveServiceContract = false;
            td.TerminationDate = Convert.ToDateTime ("05/15/2017");

            EnergyAPI.Credit.CreditAction ca = new EnergyAPI.Credit.CreditAction();
            short companyID = 2;
            string userID = "Webuser";
            int accountNum = 100643;
            short creditActionCode = 21;
            short postingCode = 121;
            int errorcode = ca.TerminateAccount(companyID, userID,
                accountNum,
                creditActionCode,
                postingCode, td);
            if (errorcode == 0)
            {
                // success
            }
            else
            {
                // retrieve message associated with the error code.
            }
        }

```

XML Example

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:TerminateAccount>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>100643</add:pAccountNum>
      <add:pCreditActionCode>21</add:pCreditActionCode>
      <add:pPostingCode>122</add:pPostingCode>
      <!--Optional:-->
      <add:pTerminateAccountData>
        <add:TerminationDate>2017-05-15</add:TerminationDate>
        <add:ActionDate>2017-05-15</add:ActionDate>
        <add:RemoveServiceContract>>false</add:RemoveServiceContract>
        <add:RemoveBudget>>false</add:RemoveBudget>
        <!--Optional:-->

```

```
<add:LogNote>API TESTING TERMINATE!!</add:LogNote>
</add:pTerminateAccountData>
</add:TerminateAccount>
</soapenv:Body>
</soapenv:Envelope>
```

CreditAction.ReinstateAccount()

- **License Required: SmartConnect API Customer Look-up Business Edition**

Description

Call this method to reinstate an account.

```
public int ReinstateAccount(  
    short pCompanyID,  
    string pSysUser,  
    int pAccountNum,  
    DCEnergyAPI.Credit.ReinstateAccountData pReinstateAccountData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyID	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer (32 bit Integer)	Account # (without checkdigit)
	pReinstateAccountData	DCEnergyAPI.Credit.ReinstateAccountData	
Return Values			
	pErrorCode	Integer (32 bit integer)	0:success, <>0: error code

DCEnergyAPI.Credit.ReinstateAccountData

field name	Description
date ActionDate ⁴	Enter the action date.
date ReinstateDate ⁴	Enter the date that the account should be reinstated.
short AcquisitionReasonCode	Enter the acquisition reason code. If the customer is being acquired from another company, for example, you can enter the reason code here. This code will be displayed in the Acquisition Info section of the Billing Info tab. (a valid reason code from list retrieved by LoadAcquisitionCodes)
double NextYearVolume	Enter the number of gallons/units you project the customer will purchase in the coming year.
string LogNote	Add up to 255 characters of comments to this credit action. Comments will be saved in Customer Log
bool DeliveryHold	True: put customer on delivery hold False: remove from delivery hold, if on hold.
bool ServiceHold	True: put customer on service hold False: remove from service hold, if on hold.
bool ClearAfterUpdate	True: If you want the credit action recorded on the customer and then cleared from these fields Credit Action Code (CAC), Credit Action Date (CAD), Reminder Date (RED) and Promised Amount (PRA).
DCEnergyAPI.Credit.ReinstateDeliveryLocation[] ActivateDeliveryLocations ¹	Array of type ReinstateDeliveryLocation Null/empty: No delivery locations will be activated. [0] = -1: ALL delivery locations will be activated. The "RemoveIfDeliveryStop" will apply

	to all [0]-[99]: specific delivery locations will be activated.
<code>short[]</code> ActivateServiceLocations ²	Array of type short Null/empty: No service locations will be activated. [0] = -1: ALL service locations will be activated. [0]-[99]: specific service locations will be activated. Eg: [0]=1 [1]=3 [2]=5 Service locations 1, 3 & 5 will be activated.
<code>short[]</code> ActivateWDMSLocations ³	Array of type short Null/empty: No WDMS locations will be activated. [0] = -1: ALL WDMS locations will be activated. [0]-[99]: specific WDMS locations will be activated. Eg: [0]=1 [1]=3 [2]=5 WDMS locations 1, 3 & 5 will be activated.

¹: If needed, the list of delivery locations could be obtained with a call to `cCustomerInfo.GetDeliveryLocations()`

²: If needed, the list of service locations could be obtained with a call to `cCustomerInfo.GetServiceLocations()`

³: If needed, the list of WDMS locations could be obtained with a call to `cCustomerInfo.GetWDMSLocations()`

NOTE - when specifying specific locations no warning will occur if the location # does not exist. Only locations that are valid and exist will be activated.

⁴: Date will default to "today" if date that is provided is less than "today" (master file date)

DCEnergyAPI.Credit.ReinstateDeliveryLocation

field name	Description
short LocationNum	Delivery location # to reinstate
bool RemoveIfDeliveryStop	True: if on delivery stop, remove from delivery stop.

C# Example

```
private void btnReinstateAccount_Click(object sender, EventArgs e)
{
    DCEnergyAPI.Credit.ReinstateAccountData ri = new
DCEnergyAPI.Credit.ReinstateAccountData();
    ri.AcquisitionReasonCode = 45;
    ri.ActionDate = Convert.ToDateTime("05/15/2017");
    ri.ReinstateDate= Convert.ToDateTime("05/15/2017");
    ri.ActivateDeliveryLocations = new DCEnergyAPI.Credit.ReinstateDeliveryLocation[] {
new DCEnergyAPI.Credit.ReinstateDeliveryLocation (-1,false)};
    short[] ss = new short[] { 1, 2 };
    ri.ActivateServiceLocations = ss;
    ri.ActivateWDMSLocations = new short[] { -1 };
    ri.ClearAfterUpdate = false;

    EnergyAPI.Credit.CreditAction ca = new EnergyAPI.Credit.CreditAction();
    short companyID = 2;
    string userID = "Webuser";
    int accountNum = 100643;

    int errorCode = ca.ReinstateAccount(companyID, userID, accountNum,ri);
    if (errorCode == 0)
    {
        // success
    }
    else
    {
        // retrieve message associated with the error code.
    }
}
```

XML Example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:add="http://www.addsys.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <add:ReinstateAccount>
      <add:pCompanyID>2</add:pCompanyID>
      <!--Optional:-->
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>100643</add:pAccountNum>
      <!--Optional:-->
      <add:pReinstateAccountData>
        <add:ActionDate>2017-05-15</add:ActionDate>
        <add:ReinstateDate>2017-05-15</add:ReinstateDate>
        <add:AcquisitionReasonCode>45</add:AcquisitionReasonCode>
        <add:NextYearVolume>0</add:NextYearVolume>
        <add:DeliveryHold>>false</add:DeliveryHold>
        <add:ServiceHold>>false</add:ServiceHold>
        <add:ClearAfterUpdate>>false</add:ClearAfterUpdate>
      <!--Optional:-->
    </add:pReinstateAccountData>
  </add:ReinstateAccount>
</soapenv:Body>
</soapenv:Envelope>
```

```
<add:ActivateDeliveryLocations>
  <!--Zero or more repetitions:-->
  <add:ReinstateDeliveryLocation>
    <add:LocationNum>-1</add:LocationNum>
    <add:RemoveIfDeliveryStop>>false</add:RemoveIfDeliveryStop>
  </add:ReinstateDeliveryLocation>
</add:ActivateDeliveryLocations>
<!--Optional:-->
<add:ActivateServiceLocations>
  <!--Zero or more repetitions:-->
  <add:short>1</add:short>
  <add:short>3</add:short>
</add:ActivateServiceLocations>
<!--Optional:-->
<add:ActivateWDMSLocations>
  <!--Zero or more repetitions:-->
  <add:short>2</add:short>
</add:ActivateWDMSLocations>
<!--Optional:-->
<add:LogNote>THIS IS API TESTING</add:LogNote>
</add:pReinstateAccountData>
</add:ReinstateAccount>
</soapenv:Body>
</soapenv:Envelope>
```

CreditAction:Error Codes

Error code	Description
7	Your new data was not saved.
96	Unknown error
337	Another user has already modified the selected information. Press OK to refresh this information then redo the action you just performed.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
745	Invalid values entered. Change the values and try again.
788	There is a problem inserting data into a table. The process did not complete. Call your System Administrator.
818	Customer has to be terminated in order to be reinstated
819	Customer has to be active in order to be terminated
951	To create a log entry, the log template must be selected.
1535	The action date and/or termination date cannot be greater than the current master file date.
1536	The acquisition reason code provided is invalid.
1537	The reminder date cannot exceed the maximum allowed date as defined by the E3 system option 'Credit - Reminder Max. Days'
1538	The release until date cannot exceed the maximum allowed date as defined by the E3 system option 'Credit - Release Until Max. Days'
1539	The credit action code, termination code and/or posting code is invalid.
1540	A credit action cannot be entered on an account that is terminated. Please reinstate the account first and try again.

LocationStatus class

Overview

This section describes the methods that will enable the programmer to retrieve/cancel/reinstate E3 delivery/service/wdms locations, it also provides lookups for retrieving pre-defined cancel & reinstate reason codes & descriptions.

- LoadCancelReasons(): Retrieves all valid pre-defined cancel reason codes & descriptions.
- LoadReinstateReasons(): Retrieves all valid pre-defined reinstate reason codes & descriptions.
- GetDeliveryLocationStatus(): Returns the current delivery location status and related information.
- CancelDeliveryLocation(): Canceles/Inactivates the Delivery location on the account.
- ReinstateDeliveryLocation(): Reinstates/Reactivates the Delivery location on the account.
- GetServiceLocationStatus(): Returns the current service location status and related information.
- CancelServiceLocation(): Canceles/Inactivates the Service location on the account.
- ReinstateServiceLocation(): Reinstates/Reactivates the Service location on the account.
- GetWdmsLocationStatus(): Returns the current wdms location status and related information.
- CancelWdmsLocation(): Canceles/Inactivates the Wdms location on the account.
- ReinstateWdmsLocation(): Reinstates/Reactivates the Wdms location on the account.

Data Classes and Datasets

Class CodeDescriptionStatus Definition:

Datatype/column name	Description
int Code	Numeric value for the reason code
string Description	Descriptive reason text for the numeric code
string Status	Status of either "A"ctive or "I"nactive
bool Selected	Internal use only

Class DeliveryLocStatusData Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
string CurrentStatus	The current status for the Delivery/Service/Wdms location
DateTime CanceledDate	The date when the DeliveryService/Wdms location was last canceled or 01/01/1900
short CancelReason	Numeric cancel reason code
string CancelReasonDesc	Description for the cancellation reason, stored in customer log
DateTime ReinstateDate	The date when the DeliveryService/Wdms location was last reinstated or 01/01/1900
short ReinstateReason	Numeric reinstate reason code
string ReinstateReasonDesc	Description for the reinstated reason, stored in customer log
String CurrentLMD	String representation of the current last mainenance date and time stamp of when the Delivery/Service/Wdms location was last modified
DateTime AcquisitionDate	Date the Delivery/Service/Wdms location was last acquired.
bool AllowStatusChange	Determines if the calling user has permissions to Cancel or reinstate the Delivery/Service/Wdms location
int ErrCode	Non-zero error code if an error is encountered by the process, else 0
string PhoneOrderExists	Y/N flag indicating if a phone order exists on the delivery location
string TicketStatus	
DateTime DDDate	The delivery due date
bool DDOverDue	Flag indicating if a delivery has not be made (based on DDDate)

Class ServiceLocStatusData Definition:

Datatype/column name	Description

int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
string CurrentStatus	The current status for the Delivery/Service/Wdms location
DateTime CanceledDate	The date when the DeliveryService/Wdms location was last canceled or 01/01/1900
short CancelReason	Numeric cancel reason code
string CancelReasonDesc	Description for the cancellation reason, stored in customer log
DateTime ReinstateDate	The date when the DeliveryService/Wdms location was last reinstated or 01/01/1900
short ReinstateReason	Numeric reinstate reason code
string ReinstateReasonDesc	Description for the reinstated reason, stored in customer log
String CurrentLMD	String representation of the current last mainenance date and time stamp of when the Delivery/Service/Wdms location was last modified
DateTime AcquisitionDate	Date the Delivery/Service/Wdms location was last acquired.
bool AllowStatusChange	Determines if the calling user has permissions to Cancel or reinstate the Delivery/Service/Wdms location
int ErrCode	Non-zero error code if an error is encountered by the process, else 0
short AccountDivision	The division on the account
short ContractServiceNum	The service contract number
string ServiceContract	The service contract code
bool ServiceContractActive	Flag indicating if a service contract is active in the service location
DateTime ContractRenewalDate	Renewal date on the service contract
decimal TankWarrAmount	The tank warranty amount
DateTime TankWarrDate	The tank warranty date
bool TankWarrActive	Flag indicating if the tank warranty is active
int NumOpenWorkOrders	Number of open workorders for this service location
int[] WorkOrdersToCancel	An array of work order numbers that would need to be canceled if the service location were to be canceled

Class WdmsLocStatusData Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
string CurrentStatus	The current status for the Delivery/Service/Wdms location
DateTime CanceledDate	The date when the DeliveryService/Wdms location was last canceled or 01/01/1900

short CancelReason	Numeric cancel reason code
string CancelReasonDesc	Description for the cancellation reason, stored in customer log
DateTime ReinstateDate	The date when the DeliveryService/Wdms location was last reinstated or 01/01/1900
short ReinstateReason	Numeric reinstate reason code
string ReinstateReasonDesc	Description for the reinstated reason, stored in customer log
String CurrentLMD	String representation of the current last maintenance date and time stamp of when the Delivery/Service/Wdms location was last modified
DateTime AcquisitionDate	Date the Delivery/Service/Wdms location was last acquired.
bool AllowStatusChange	Determines if the calling user has permissions to Cancel or reinstate the Delivery/Service/Wdms location
int ErrCode	Non-zero error code if an error is encountered by the process, else 0
int WorkOrderCount	Number of work orders on the WDMS location

Class DeliveryLocStatusUpdateData Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
String CurrentLMD	String representation of the current last maintenance date and time stamp of when the Delivery/Service/Wdms location was last modified
short ReasonCode	Numeric cancel/reinstate reason code
String(50) ReasonDesc	Description for the cancel/reinstated reason, stored in customer log
string(255) UserComments	Free form user comments
bool CancelPhoneOrder	A flag indicating that any phone orders on the Delivery location will be canceled if the location is canceled
bool CancelTicket	A flag indicating that any tickets on the Delivery location will be canceled if the location is canceled
bool DDOverDue	Flag indicating the a delivery is overdue when reinstating a delivery location

Class ServiceLocStatusUpdateData Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
String CurrentLMD	String representation of the current last

	maintenance date and time stamp of when the Delivery/Service/Wdms location was last modified
short ReasonCode	Numeric cancel/reinstate reason code
String(50) ReasonDesc	Description for the cancel/reinstated reason, stored in customer log
string(255) UserComments	Free form user comments
short AccountDivision	The division on the account
bool CancelContract	A flag indicating that any contracts on the Service location will be canceled if the location is canceled
bool CancelWorkOrders	A flag indicating that any work orders on the Service location will be canceled if the location is canceled
bool CancelTankWarranty	A flag indicating that any tank warranty on the Service location will be canceled if the location is canceled
short ContractServiceNum	The service contract number
short OrderCount	The count of open orders on the WdmsLocation

Class WdmsLocStatusUpdateData Definition:

Datatype/column name	Description
int AccountNum	E3 account number without check digit
short LocationNum	Delivery/Service/Wdms location number
String CurrentLMD	String representation of the current last maintenance date and time stamp of when the Delivery/Service/Wdms location was last modified
short ReasonCode	Numeric cancel/reinstate reason code
String(50) ReasonDesc	Description for the cancel/reinstated reason, stored in customer log
string(255) UserComments	Free form user comments
bool CancelPhoneOrder	A flag indicating that any phone orders on the Delivery location will be canceled if the location is canceled
bool CancelTicket	A flag indicating that any tickets on the Delivery location will be canceled if the location is canceled
bool DDOverDue	Flag indicating the a delivery is overdue when reinstating a delivery location
short AccountDivision	The division on the account
bool CancelContract	A flag indicating that any contracts on the Service location will be canceled if the location is canceled
bool CancelWorkOrders	A flag indicating that any work orders on the Service location will be canceled if the location is canceled
bool CancelTankWarranty	A flag indicating that any tank warranty on the Service location will be canceled if the location is

	canceled
short ContractServiceNum	The service contract number
short OrderCount	The count of open orders on the WdmsLocation

Class ServiceLocStatusUpdateResults Definition:

Datatype/column name	Description
int ErrCode	Non-zero error code if process encountered an error else 0
PegasusWoErrCodes Description	Custom data structure containing error code for pegasus workorders than could not be canceled

Class PegasusWoErrCodes Definition:

Datatype/column name	Description
int WorkOrderNumber	Workorder number that could not be canceled in Pegasus
string ErrDesc	An error description for why the pegasus workorder was not canceled

LocationStatus.LoadCanceledReasons()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to retrieve all pre-defined cancellation reasons in E3 Codes and Descriptions.

Method Signature:

```
public DCEnergyAPI.LookUp.CodeDescriptionStatus[] LoadCancelReasons(short pCompanyId, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCEnergyAPI.LookUp.CodeDescriptionStatus			
		Array of CodeDescriptionStatus data structure	List of valid pre-defined cancel reasons

Error Codes:

Return code	Description

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCancelReasons>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>webUser</add:pSysUser>
    </add:LoadCancelReasons>
  </soap:Body>
</soap:Envelope>
```

LocationStatus.LoadReinstateReasons()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to retrieve all pre-defined reinstate reasons in E3 Codes and Descriptions.

Method Signature:

```
public DCEnergyAPI.LookUp.CodeDescriptionStatus[] LoadReinstateReasons(short pCompanyId, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Type: DCEnergyAPI.LookUp.CodeDescriptionStatus			
		Array of CodeDescriptionStatus data structure	List of valid pre-defined reinstate reasons

Error Codes:

Return code	Description

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadReinstateReasons>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadReinstateReasons>
  </soap:Body>
</soap:Envelope>
```

LocationStatus.GetDeliveryLocationStatus()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to retrieve the current status of a given E3 customer Location.

Method Signature:

```
public DeliveryLocStatusData GetDeliveryLocationStatus(short pCompanyId,
string pSysUser, int pAccountNum, short pLocationNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Int (32 bit integer)	E3 account number withoput check digit
	pLocationNum	Short (16 bit integer)	E3 Delivery location number
Return Type: DCEnergyAPI.LocationStatus.DeliveryLocStatusData			
		Data structure containing information about the current status of an E3 customer Location	

Error Codes:

Return code	Description
122	Invalid input parameters

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetDeliveryLocationStatus>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>webUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLocationNum>1</add:pLocationNum>
    </add:GetDeliveryLocationStatus>
  </soap:Body>
</soap:Envelope>
```

LocationStatus.CancelDeliveryLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to cancel/inactivate a given E3 customer Location.

Method Signature:

```
public int CancelDeliveryLocation(short pCompanyId, string pSysUser,
DeliveryLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	DeliveryLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		Int	Non-zero error if any

Error Codes:

Return code	Description
122	Invalid input parameters
1543	Energy was unable to process your request because the Cancel Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CancelDeliveryLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
        <add:CurrentLMD>May 24 2017 11:09:58:463AM</add:CurrentLMD>
        <add:ErrCode>0</add:ErrCode>
        <add:ReasonCode>4</add:ReasonCode>
      </add:pUpdateData>
    </add:CancelDeliveryLocation>
  </soap:Body>
</soap:Envelope>
```

```
<add:ReasonDesc>blah</add:ReasonDesc>
<add:UserComments>User Comments</add:UserComments>
<add:CancelPhoneOrder>true</add:CancelPhoneOrder>
<add:CancelTicket>true</add:CancelTicket>
<add:DDOverDue>false</add:DDOverDue>
</add:pUpdateData>
</add:CancelDeliveryLocation>
</soap:Body>
</soap:Envelope>
```


LocationStatus.ReinstateDeliveryLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to reinstate/activate a given E3 customer Location.

Method Signature:

```
public int ReinstateDeliveryLocation(short pCompanyId, string pSysUser,
DeliveryLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	DeliveryLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		Int	Non-zero error if any

Error Codes:

Return code	Description
122	Invalid input parameters
949	Energy was unable to process your request because the Reinstate Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ReinstateDeliveryLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
        <add:CurrentLMD>May 24 2017 11:09:58:463AM</add:CurrentLMD>
        <add:ErrCode>0</add:ErrCode>
        <add:ReasonCode>1</add:ReasonCode>
      </add:pUpdateData>
    </add:ReinstateDeliveryLocation>
  </soap:Body>
</soap:Envelope>
```

```
      <add:ReasonDesc>who cares?</add:ReasonDesc>
      <add:UserComments>Nobody cares!</add:UserComments>
      <add:CancelPhoneOrder>true</add:CancelPhoneOrder>
      <add:CancelTicket>true</add:CancelTicket>
      <add:DDOverDue>false</add:DDOverDue>
    </add:pUpdateData>
  </add:ReinstateDeliveryLocation>
</soap:Body>
</soap:Envelope>
```

LocationStatus.GetServiceLocationStatus()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to retrieve the current status of a given E3 customer Location.

Method Signature:

```
public ServiceLocStatusData GetServiceLocationStatus(short pCompanyId, string pSysUser, int pAccountNum, short pLocationNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Int (32 bit integer)	E3 account number withoput check digit
	pLocationNum	Short (16 bit integer)	E3 Delivery location number
Return Type: DCEnergyAPI.LocationStatus.ServiceLocStatusData			
		Data structure containing information about the current status of an E3 customer Location	

Error Codes:

Return code	Description
122	Invalid input parameters

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetServiceLocationStatus>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>webUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLocationNum>1</add:pLocationNum>
    </add:GetServiceLocationStatus>
  </soap:Body>
</soap:Envelope>
```

LocationStatus.CancelServiceLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to cancel/inactivate a given E3 customer Location.

Method Signature:

```
public ServiceLocStatusUpdateResults CancelServiceLocation(short pCompanyId,
string pSysUser, ServiceLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	ServiceLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		ServiceLocationStatusUpdateResults	Custom data structure, detailing return error codes and any Pegasus update errors

Error Codes:

Return code	Description
122	Invalid input parameters
1543	Energy was unable to process your request because the Cancel Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CancelServiceLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>webUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
      </add:pUpdateData>
    </add:CancelServiceLocation>
  </soap:Body>
</soap:Envelope>
```

```
<add:CurrentLMD>May 24 2017 11:18:26:446AM</add:CurrentLMD>
<add:ErrCode>0</add:ErrCode>
<add:ReasonCode>1</add:ReasonCode>
<add:ReasonDesc>We don't care</add:ReasonDesc>
<add:UserComments>Ah, but you do</add:UserComments>
<add:AccountDivision>1</add:AccountDivision>
<add:CancelContract>true</add:CancelContract>
<add:CancelWorkOrders>true</add:CancelWorkOrders>
<add:CancelTankwarranty>true</add:CancelTankwarranty>
<add:ContractServiceNum>0</add:ContractServiceNum>
</add:pUpdateData>
</add:CancelServiceLocation>
</soap:Body>
</soap:Envelope>
```

LocationStatus.ReinstateServiceLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to reinstate/activate a given E3 customer Location.

Method Signature:

```
public int ReinstateServiceLocation(short pCompanyId, string pSysUser,
ServiceLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	ServiceLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		Int	Non-zero error if any

Error Codes:

Return code	Description
122	Invalid input parameters
949	Energy was unable to process your request because the Reinstate Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ReinstateServiceLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
        <add:CurrentLMD>May 24 2017 11:18:53:310AM</add:CurrentLMD>
        <add:ErrCode>0</add:ErrCode>
        <add:ReasonCode>1</add:ReasonCode>
      </add:pUpdateData>
    </add:ReinstateServiceLocation>
  </soap:Body>
</soap:Envelope>
```

```
<add:ReasonDesc>We don't care</add:ReasonDesc>
<add:UserComments>Ah, but you do</add:UserComments>
<add:AccountDivision>1</add:AccountDivision>
<add:CancelContract>true</add:CancelContract>
<add:CancelWorkOrders>true</add:CancelWorkOrders>
<add:CancelTankWarranty>true</add:CancelTankWarranty>
<add:ContractServiceNum>0</add:ContractServiceNum>
</add:pUpdateData>
</add:ReinstateServiceLocation>
</soap:Body>
</soap:Envelope>
```

LocationStatus.GetWdmsLocationStatus()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to retrieve the current status of a given E3 customer Location.

Method Signature:

```
public WdmsLocStatusData GetDeliveryLocationStatus(short pCompanyId, string pSysUser, int pAccountNum, short pLocationNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Int (32 bit integer)	E3 account number without check digit
	pLocationNum	Short (16 bit integer)	E3 Delivery location number
Return Type: DCEnergyAPI.LocationStatus.WdmsLocStatusData			
		Data structure containing information about the current status of an E3 customer Location	

Error Codes:

Return code	Description
122	Invalid input parameters

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:GetWdmsLocationStatus>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>webUser</add:pSysUser>
      <add:pAccountNum>25</add:pAccountNum>
      <add:pLocationNum>1</add:pLocationNum>
    </add:GetWdmsLocationStatus>
  </soap:Body>
</soap:Envelope>
```


LocationStatus.CancelWdmsLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to cancel/inactivate a given E3 customer Location.

Method Signature:

```
public int CancelWdmsLocation(short pCompanyId, string pSysUser,
DeliveryLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	WdmsLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		Int	Non-zero error if any

Error Codes:

Return code	Description
122	Invalid input parameters
1543	Energy was unable to process your request because the Cancel Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:CancelWdmsLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
        <add:CurrentLMD>May 23 2017 4:25:21:120PM</add:CurrentLMD>
        <add:ErrCode>0</add:ErrCode>
        <add:ReasonCode>1</add:ReasonCode>
      </add:pUpdateData>
    </add:CancelWdmsLocation>
  </soap:Body>
</soap:Envelope>
```

```
        <add:ReasonDesc>We don't care</add:ReasonDesc>
        <add:UserComments>Ah, but you do</add:UserComments>
        <add:OrderCount>0</add:OrderCount>
      </add:pUpdateData>
    </add:CancelWdmsLocation>
  </soap:Body>
</soap:Envelope>
```

LocationStatus.ReinstateWdmsLocation()

- **License Required: SmartConnect API Customer Look-up Business Edition**
- **License Required: SmartConnect Customer Maintenance**

Description

This method will be called to reinstate/activate a given E3 customer Location.

Method Signature:

```
public int ReinstateWdmsLocation(short pCompanyId, string pSysUser,
DeliveryLocStatusUpdateData pUpdateData)
```

INPUT PARAMETERS			
Accepts	Variable	Data type	Description
	pCompanyId	Short (16 bit integer)	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pUpdateData	WdmsLocStatusUpdateData	Data structure containing information required to update the location status
Return Type:			
		Int	Non-zero error if any

Error Codes:

Return code	Description
122	Invalid input parameters
949	Energy was unable to process your request because the Reinstate Reason code is invalid. Please contact ADD Systems Customer Support.
521	You do not have permission to run this function.
565	This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
96	Unknown error.

XML document:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:ReinstateWdmsLocation>
      <add:pCompanyId>2</add:pCompanyId>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pUpdateData>
        <add:AccountNum>25</add:AccountNum>
        <add:LocationNum>1</add:LocationNum>
        <add:CurrentLMD>May 24 2017 11:32:28:746AM</add:CurrentLMD>
        <add:ErrCode>0</add:ErrCode>
        <add:ReasonCode>1</add:ReasonCode>
      </add:pUpdateData>
    </add:ReinstateWdmsLocation>
  </soap:Body>
</soap:Envelope>
```

```
        <add:ReasonDesc>We don't care</add:ReasonDesc>
        <add:UserComments>Ah, but you do</add:UserComments>
        <add:OrderCount>0</add:OrderCount>
      </add:pUpdateData>
    </add:ReinstateWdmsLocation>
  </soap:Body>
</soap:Envelope>
```

Contract Class

Overview

This section describes the methods that will enable the developer to create/modify WDMS Customer Contracts.

Contract.LoadCostTypes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all valid cost types as defined in the system.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadCostTypes(short pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of cost types.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadCostTypes>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadCostTypes>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadDepartmentCodes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve valid department codes as defined in the system.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadDepartmentCodes(short pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of department codes.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadDepartmentCodes>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadDepartmentCodes>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadMFGCodes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all manufacturer codes as defined in the system.

```
public DCWDMS.Maintenance.Contracts.AlphaCodeDescription[] LoadMFGCodes(short pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.AlphaCodeDescription			
		Array of AlphaCodeDescription	List of manufacturer codes.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadMFGCodes>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadMFGCodes>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadGroupCodes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all group codes associated with the manufacturer code.

```
public DCWDMS.Maintenance.Contracts.AlphaCodeDescription[] LoadGroupCodes(short pCompanyID,  
    string pSysUser, string pMFGCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pMFGCode	String	Manufacturer
Return Values: DCWDMS.Maintenance.Contracts.AlphaCodeDescription			
		Array of AlphaCodeDescription	List of group codes.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:LoadGroupCodes>  
      <add:pCompanyID>1800</add:pCompanyID>  
      <add:pSysUser>WebUser</add:pSysUser>  
      <add:pMFGCode>LIQ</add:pMFGCode>  
    </add:LoadGroupCodes>  
  </soap:Body>  
</soap:Envelope>
```


Contract.LoadStockIDS()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all parts associated with the manufacturer & group code combination.

```
public DCWDMS.Maintenance.Contracts.AlphaCodeDescription[] LoadStockIDS(short pCompanyID, string pSysUser, string pMFGCode, string pGroupCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pMFGCode	String	Manufacturer
	pGroupCode	String	Group Code
Return Values: DCWDMS.Maintenance.Contracts.AlphaCodeDescription			
		Array of AlphaCodeDescription	List of parts.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadStockIDS>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pMFGCode>LIQ</add:pMFGCode>
      <add:pGroupCode>DIST</add:pGroupCode>
    </add:LoadStockIDS>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadPostingCodes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all posting codes as defined in the system.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadPostingCodes(short pCompanyID,  
string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of posting codes.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:LoadPostingCodes>  
      <add:pCompanyID>1800</add:pCompanyID>  
      <add:pSysUser>WebUser</add:pSysUser>  
    </add:LoadPostingCodes>  
  </soap:Body>  
</soap:Envelope>
```

Contract.LoadSalesPersons()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all sales people as defined in the system.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadSalesPersons(short pCompanyID,  
    string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of sales persons.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:LoadSalesPersons>  
      <add:pCompanyID>1800</add:pCompanyID>  
      <add:pSysUser>WebUser</add:pSysUser>  
    </add:LoadSalesPersons>  
  </soap:Body>  
</soap:Envelope>
```

Contract.LoadSalesTypes()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all sales types as defined in the system.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadSalesTypes(short pCompanyID, string pSysUser)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of sales types.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadSalesTypes>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
    </add:LoadSalesTypes>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadSuppliers()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all suppliers by supply location (optional).

```
public DCWDMS.Maintenance.Contracts.AlphaCodeDescription[] LoadSuppliers(short pCompanyID,  
string pSysUser, string pSupplyLocation)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pSupplyLocation	String	Provide a supply location for filtering or blank for all
Return Values: DCWDMS.Maintenance.Contracts.AlphaCodeDescription			
		Array of AlphaCodeDescription	List of suppliers.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"  
xmlns:add="http://www.addsys.com/">  
  <soap:Header/>  
  <soap:Body>  
    <add:LoadSuppliers>  
      <add:pCompanyID>1800</add:pCompanyID>  
      <add:pSysUser>WebUser</add:pSysUser>  
      <add:pSupplyLocation>ABC</add:pSupplyLocation>  
    </add:LoadSuppliers>  
  </soap:Body>  
</soap:Envelope>
```

Contract.LoadSupplyLocations()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all supply locations by supplier code (optional).

```
public DCWDMS.Maintenance.Contracts.AlphaCodeDescription[] LoadSupplyLocations(short pCompanyID, string pSysUser, string pSupplierCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pSupplierCode	String	Provide a supplier for filtering or blank for all
Return Values: DCWDMS.Maintenance.Contracts.AlphaCodeDescription			
		Array of AlphaCodeDescription	List of supply locations.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadSupplyLocations>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pSupplierCode>DEF</add:pSupplierCode>
    </add:LoadSupplyLocations>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadWdmsLocations()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all WDMS locations linked to the customer.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadWdmsLocations(short pCompanyID
, string pSysUser, int pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	Account # (without checkdigit)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of WDMS locations.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadWdmsLocations>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>100</add:pAccountNum>
    </add:LoadWdmsLocations>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadContracts()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve all existing contracts linked to the customer.

```
public DCWDMS.Maintenance.Contracts.NumCodeDescription[] LoadContracts(short pCompanyID, string pSysUser, int pAccountNum)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	Account # (without checkdigit)
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of NumCodeDescription	List of defined contracts for customer (if any).

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContracts>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pAccountNum>1234</add:pAccountNum>
    </add:LoadContracts>
  </soap:Body>
</soap:Envelope>
```


Contract.LoadContract()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve details for one contract linked to the customer.

```
public DCWDMS.Maintenance.Contracts.ContractData LoadContract(short pCompanyID, string
pSysUser ,int pAccountNum, int pContractID, ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pAccountNum	Integer	Account # (without checkdigit)
	pContractID	Integer	Contract id
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
	pErrorCode	Int	0:success, <>0: error code
		Array of NumCodeDescription	List of defined contracts for customer.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadContract>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pAccountNum>4921</add:pAccountNum>
      <add:pContractID>535</add:pContractID>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadContract>
  </soap:Body>
</soap:Envelope>
```

Contract.LoadSlaveContracts()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to retrieve list of subordinate contracts (if any).

```
public DCWDMS.Maintenance.Contracts.AccountNumAndContractCodeDescription[] LoadSlaveContracts(short pCompanyID, string pSysUser, int pContractID)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pContractID	Integer	A valid contract id
Return Values: DCWDMS.Maintenance.Contracts.NumCodeDescription			
		Array of AccountNumAndContractCodeDescription	List of subordinate contracts or empty

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:LoadSlaveContracts>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pContractID>535</add:pContractID>
      <add:pErrorCode>0</add:pErrorCode>
    </add:LoadSlaveContracts>
  </soap:Body>
</soap:Envelope>
```

AccountNumAndContractCodeDescription		
Field Name	Data Type	Description
AccountNum	int	Account # (without checkdigit)
ContractID	int	A valid contract id (LoadContract and LoadSlaveContracts) or 0 when creating a new contract record.
ContractCode	string	A contract code that you define.
Description	string	A contract description

Contract.SaveContract()

- **License Required: SmartConnect WDMS Order Creation/Maintenance**

Description

This method will be called to create/update one contract record for a given customer.

```
public DCWDMS.Maintenance.Contracts.ContractData SaveContract(short pCompanyID, string pSysUser, DCWDMS.Maintenance.Contracts.ContractData pContractData, ref int pErrorCode)
```

INPUT PARAMETERS			
Accepts	Variable	Data Type	Description
	pCompanyID	Short Integer	The Energy Company ID
	pSysUser	String	Calling program (IVRUser/WebUser)
	pContractData	ContractData	Contract related data
Return Values: DCWDMS.Maintenance.Contracts. ContractData			
	pErrorCode	Integer	0:success, <>0: error code
	pContractData	<u>ContractData</u>	Updated Data Structure

IMPORTANT: WHEN MODIFYING AN EXISTING RECORD, ALL DATA MUST BE PROVIDED FOR EACH FIELD EVEN IF THE DATA IS NOT CHANGING. A CALL TO **LOADCONTRACT()** IS REQUIRED TO GET THE EXISTING DATA.

Example

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:add="http://www.addsys.com/">
  <soap:Header/>
  <soap:Body>
    <add:SaveContract>
      <add:pCompanyID>1800</add:pCompanyID>
      <add:pSysUser>WebUser</add:pSysUser>
      <add:pContractData>
        <add:AccountNumAndContractDescription>
          <add:AccountNum>4291</add:AccountNum>
          <add:ContractID>535</add:ContractID>
          <add:ContractCode>DC6</add:ContractCode>
          <add:Description>DC-WDMS</add:Description>
        </add:AccountNumAndContractDescription>
        <add:ContractDate>2018-08-11T00:00:00</add:ContractDate>
        <add:StartDate>2018-08-11T00:00:00</add:StartDate>
      </add:pContractData>
    </add:SaveContract>
  </soap:Body>
</soap:Envelope>
```

```
<add:StartDateEffectiveTime>14:21</add:StartDateEffectiveTime>
<add:EndDate>2020-08-12T00:00:00</add:EndDate>
<add:EndDateEffectiveTime>15:20</add:EndDateEffectiveTime>
<add:OpenEnded>false</add:OpenEnded>
<add:OriginalUnits>40002</add:OriginalUnits>
<add:RemainingUnits>40002</add:RemainingUnits>
<add:Bonded>false</add:Bonded>
<add:Terminated>false</add:Terminated>
<add:FreightIncluded>false</add:FreightIncluded>
<add:Price>1.25</add:Price>
<add:ContractType>1</add:ContractType>
<add:Supplier>CITG</add:Supplier>
<add:MaxPrice>0</add:MaxPrice>
<add:BasePriceCode>0</add:BasePriceCode>
<add:CostType>3</add:CostType>
<add>Status>A</add>Status>
<add:SalesType>4</add:SalesType>
<add:DepartmentCode>1100</add:DepartmentCode>
<add:MfgCode>LIQ</add:MfgCode>
<add:GroupCode>DIST</add:GroupCode>
<add:StockId>22</add:StockId>
<add:AssignedCost>0.25</add:AssignedCost>
<add>LastMaintenanceDt>Jun 18 2018
4:13:27:146PM</add>LastMaintenanceDt>
<add:MasterAccountNum>0</add:MasterAccountNum>
<add:MasterContractId>0</add:MasterContractId>
<add:HasSlaves>false</add:HasSlaves>
<add:UseMasterContractPrice>false</add:UseMasterContractPrice>
<add:SalesPerson>3</add:SalesPerson>
<add:OpenEndedUnits>false</add:OpenEndedUnits>
<add:MinimumLoadQty>10</add:MinimumLoadQty>
<add:MaximumLoadQty>110</add:MaximumLoadQty>
<add:SplitLastLoad>false</add:SplitLastLoad>
<add>LastLoadUnlimited>false</add>LastLoadUnlimited>
<add>LastLoadUnitsLimit>1000</add>LastLoadUnitsLimit>
<add:SupplierLocationCode>IL-West</add:SupplierLocationCode>
<add:ConsigneeNumber>111</add:ConsigneeNumber>
<add:BaseSubLevel>0</add:BaseSubLevel>
<add:IncludeInQuoting>false</add:IncludeInQuoting>
<add:QuoteCostPlus>true</add:QuoteCostPlus>
<add:Locations>
  <add:short>1</add:short>
</add:Locations>
<add:Products/>
<add:AddOnSuppliers/>
<add>CreateePriceQuote>false</add>CreateePriceQuote>
<add:EPriceQuoteEffectiveTime>00:01</add:EPriceQuoteEffectiveTime>
<add:UnitAllocations>
  <add:int>0</add:int>
  <add:int>0</add:int>
  <add:int>0</add:int>
```

```
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
<add:int>0</add:int>
</add:UnitAllocations>
<add:TotalQuotesCreated>0</add:TotalQuotesCreated>
</add:pContractData>
<add:pErrorCode>0</add:pErrorCode>
</add:SaveContract>
</soap:Body>
```

ContractData		
Field Name	Data Type	Description
accountNumAndContract Description	AccountNumAndContractCode Description	contains the accountNum, contract id or 0, contract code & description refer to AccountNumAndContractCodeDescription definition
ContractDate	date	Indicates the date that the contract was made. This date can be used for reporting purposes when calculating remaining quantities as of a certain point in time.
StartDate	date	The contract starting date.
StartDateEffectiveTime	string	The time that the contract will start. If value not supplied it will default to 00:01. Entered in military time as HH:mm.
EndDate	date	The contract ending date.
EndDateEffectiveTime	string	The time that the contract will end. If value not supplied it will default to 23:59. Entered in military time as HH:mm
OpenEnded	bool	Contracts may have a fixed term or they may be open-ended with no ending date. An open-ended contract price will remain in effect indefinitely until a new contract is established. True=Open Ended; in this case EndDate/EndDateEffectiveTime will be ignored.
OriginalUnits	int	The original number of gallons for this contract. (1 – 99999999).
RemainingUnits	int	As product is sold, the system reduces the remaining gallons value until it reaches zero. Must be <= OriginalUnits.
Bonded	bool	If True, this flag indicates that the customer has bonded the contract with a lending institution. If false the contract has not been bonded. This field is intended for information only.
Terminated	bool	If True, causes the contract to become inactive.

FreightIncluded	bool	If True, this flag indicates that the contract price already includes freight.
Price	decimal	contract type 0: the value represents the actual price contract type 1: the value represents the deviation or amount which will be added to the unit cost of a specific load contract type 2: the value represents the per-gallon deviation from the base price code. The deviation can be positive or negative. contract type 3: the value represents a percent of sale deviation from the base price code, which overrides the location price category. This can be plus or minus.
ContractType	ContractType	0:Actual Price 1: Supplier Cost Deviation 2:Dollar Deviation 3:Percent Deviation
Status	string	"A"ctive, "I"nactive
Supplier	string	Supplier code as retrieved by LoadSuppliers() or blank *NOTE* only valid when Pricing Contract Type is 1: Supplier Cost Deviation
MaxPrice	decimal	The value represents the price cap for the contract.
BasePriceCode	short	The value represents the base price code for this contract. *NOTE* only valid when Pricing Contract Type is 2:Dollar Deviation or 3:Percent Deviation
CostType	short	cost type as retrieved by LoadCostTypes () or 0
ConsigneeNumber	string	A PIN number that identifies the customer and contract that should be used for pricing, or blank
SalesType	short	sales type code as retrieved by LoadSalesTypes() or 0
DepartmentCode	short	department code as retrieved by LoadDepartmentCodes() or 0
MfgCode ¹	string	Specify a manufacturer, as retrieved by LoadMFGCodes(), to be covered under the contract or blank.

GroupCode ¹	string	Specify a group code, as retrieved by LoadGroupCodes (), associated with the manufacturer code or blank.
StockId ¹	string	Specify a part, as retrieved by LoadStockIDs(), associated with the manufacturer & group code or blank.
AssignedCost	decimal	This field is primarily intended for reporting purposes. It provides an additional cost field for the purpose of capturing special cost information that may apply to the invoice. When an assigned cost is entered in this field, it will override the cost type assigned from the Cost Type Table.
LastMaintenanceDt	string	the last maintenance date as retrieved by LoadCustomerContract() or blank when creating a new contract
MasterAccountNum ²	int	Master account number (without checkdigit) If the contract is dependent on a master contract, otherwise 0.
MasterContractId ²	int	Master contract id, if contract is a subordinate contract, otherwise 0
HasSlaves	bool	If True, indicates that the contract has subordinate contract(s). That is, the contract is a master contract to other contracts. *NOTE* read-only field.
UseMasterContractPrice	bool	If True, indicates that the product delivered to the account with the subordinate contract will be priced using the master contract price. Otherwise, product will be priced using the subordinate contract price. The value cannot be true if the MasterAccountNum equals the account number of the contract being modified and/or created.
SalesPerson	int	Salesperson value as retrieved by LoadSalesPersons() or 0
OpenEndedUnits ⁴	bool	If True, indicates the contract will be open-ended, i.e., it will not have a limit.

MinimumLoadQty	int	Enter a value in this field to indicate the minimum load the customer may be sold under this contract. Or 0. This field is for information only.
MaximumLoadQty	int	Enter a value in this field to indicate the maximum load the customer may be sold under this contract. Or 0. This field is for information only.
SplitLastLoad ³	bool	If True, allows the system to end the contract when the customer has used up all of the original units.
LastLoadUnlimited ³	bool	If True, allows the customer to overdraw unlimited units.
LastLoadUnitsLimit	int	If LastLoadUnlimited is false the value must be non zero, otherwise zero.
SupplierLocationCode	string	Supply Location as retrieved by LoadSupplyLocations () or blank *NOTE* only valid when Pricing Contract Type is 1: Supplier Cost Deviation
BaseSubLevel	int	The value represents the base sub-level code which works in conjunction with the base price code to get the proper price. *NOTE* only valid when Pricing Contract Type is 2:Dollar Deviation or 3:Percent Deviation
IncludeInQuoting	bool	If True, indicates that the contract should be included in ePrice Quote Processing.
QuoteCostPlus	bool	If True, indicates that the contract will be used as the default for customer / locations that have been assigned the cost-plus sales agreement type
Locations[]	short[]	An array of type short that includes all the locations the contract applies to.
Products[]	short[]	An array of type short that includes all the products covered by the contract. Null is contract is by WDMS part.
AddOnSuppliers	string[]	An array of type string that includes all contract suppliers (as retrieved by LoadSuppliers()) whose eligible add-ons will contribute to the cost basis when determining selling price. Or null. *NOTE* only valid when Pricing Contract Type is 1: Supplier Cost Deviation

CreateePriceQuote	bool	If True, an ePrice Quote will be issued for the contract price or for the supplier/supply location/product covered under the contract. Only Actual Price and Supplier Cost Deviation type contracts are eligible for ePrice Quoting. *NOTE* the customer must have an ePrice Quote Profile established in E3 ePrice Quote Parameter Maintenance. IncludeInQuoting must also be True.
EPriceQuoteEffectiveTime	string	the effective time of the quote when generating an ePrice Quote for the contract. Time must be using a 24-hour military format. Default is 00:01
UnitAllocations[]	int[]	An array of int that represents the original units spread out into 24 months. The total must equal the original units. Or NULL *NOTE* the units can only be spread out maximum 2 years even if the contract spans over 2 years.
TotalQuotesCreated	int	If CreateePriceQuote is True, this value will contain the total # of quotes created. *NOTE* meant as a read-only field therefore any value sent to the save will be ignored.
1		valid value combinations: blank/blank/blank mfg code/blank/blank mfg code/group code/blank mfg code/group code/stockid
2		either both are 0 or both have values
3		SplitLastLoad & LastLoadUnlimited cannot both be true
4		If OpenEndedUnits is true the following data fields will be initialized during the save: OriginalUnits RemainingUnits SplitPrice, LastLoadUnlimited & LastLoadUnitsLimit

IMPORTANT

When a master contract is assigned to a contract where the master account <> the account in question, the following data will be overwritten with the master contract during the save process.

Products[]
Original Units
RemainingUnits
MinimumLoad
MaximumLoad

When a master contract is assigned to a contract where the master account = the account in question, the following data will be overwritten with the master contract during the save process:

ContractDate
StartDate
StartDateEffectiveTime
EndDate
EndDateEffectiveTime
Products[]
Original Units
RemainingUnits
MinimumLoad
MaximumLoad

error code	description
525	Can't delete the entry(ies) at this time because the entry(ies) have related records that exist.
1512	Contract information cannot be saved. One or more posting codes are linked to another subordinate contract.
1575	A contract can cover specific posting code(s) or a WDMS product, but not both.
1576	At least one WDMS location must be provided.
1577	When selecting a WDMS product (Mfg Code, Group Code, Stock ID), selection can be either All/All/All, Mfg Code/All/All, Mfg Code/Group Code/All or Mfg Code/Group Code/Stock ID.
1578	Invalid Mfg Code, Group Code and/or Stock ID.
1579	A supplier must be provided.
1580	Invalid Supplier location.
1581	Invalid Supplier.
1582	Either 'Split price' or 'Use Contract Price for overdraw' can be selected, but not both.
1583	When 'Unlimited overdraw units' is not selected, a value is required in 'Maximum overdraw units'.
1584	The sum of the allocation units must total the original units.
1585	Master Account# and Master contract must either both be zero or each have a value.
1586	A contract cannot be its own Master contract.
1587	Invalid Master contract.
1588	Invalid Master account.
1589	A contract can only be a Master contract if coverage is by posting code(s).
1590	Remaining Units' must be less than or equal to 'Original Units'.
1591	Either 'Include in Quoting' or 'Quote Cost Plus' can be selected, but not both.
1592	Contract end date cannot be less than the start date.
1593	The contract date has exceeded the minimum/maximum allowable date as defined in the system.
1594	The start date has exceeded the minimum/maximum allowable date as defined in the system.
1595	The start date cannot be modified.
1596	The ending date exceeds the maximum allowable days from the starting date as defined in the system.
1597	No ePrice Quote Profile exists in ePrice Quote Parameter Maintenance for the customer and contract.
1598	Invalid Cost Type.
1599	Invalid Department Code.
1600	Invalid Sales Type.
1601	Invalid Salesperson Code.

Appendix A

Enterprise v. Web Service Overview

This section is dedicated to showing the differences in coding against smartconnect deployed as an enterprise service v. web service (All examples are in C#).

Declaration (Enterprise)

```
EnergyAPI.cCustBatch custBatchMaint = new EnergyAPI.cCustBatch();
```

Declaration (Web Service)

```
EnergyAPIWebService.cCustBatch custBatchMaint = new EnergyAPIWebService.cCustBatch();  
custBatchMaint.Url="http://some address" + "cCustBatch.asmx";
```

Method Call (Enterprise)

```
(eg: #1)  
EnergyAPI.cCustomerInfo customerInfo = new EnergyAPI.cCustomerInfo();  
string accountIn = "123";  
string fullAccount="";  
string sysUser="webuser";  
int accountNum =0;  
short companyID=2;  
int returnCode = customerInfo.ValidateAccount (accountIn,sysUser,ref fullAccount,ref  
accountNum, ref companyID);
```

Method Call (Web Service)

```
(eg: #1)  
EnergyAPIWebService.cCustomerInfo customerInfo = new  
EnergyAPIWebService.cCustomerInfo ();  
//  
customerInfo.Url="http://some address" + "cCustomerInfo.asmx";  
//  
string accountIn = "123";  
string fullAccount="";  
string sysUser="webuser";  
int accountNum =0;  
short companyID=2;  
int returnCode = customerInfo.ValidateAccount (accountIn,sysUser,ref fullAccount,ref  
accountNum, ref companyID);
```

Method Call (Enterprise)

```
(eg: #2)  
EnergyAPI.cCustBatch custBatchMaint = new EnergyAPI.cCustBatch();  
//
```

```

DataSet addCustomerDS = new DataSet ();
//
short pCompanyID =2;
string pSysUserid="webuser";
int errorCode =0;
addCustomerDS =custBatchMaint.CustBatchAdd(pCompanyID, pSysUserId, addCustomerDS, -
900, ref errorCode)

```

Method Call (Web Service)

(eg: #2)

```

EnergyAPIWebService.cCustBatch custBatchMaint = new EnergyAPIWebService.cCustBatch();
custBatchMaint.Url="http://some address" + "cCustBatch.asmx";
//
DataSet addCustomerDS = new DataSet ();
//
short pCompanyID =2;
string pSysUserid="webuser";
int errorCode =0;
//
addCustomerDS = GetDSfromXML(custBatchMaint.CustBatchAdd(pCompanyID, pSysUserId,
GetXmlFromDS(addCustomerDS), -900, ref errorCode));

```

*** NOTE *** : A Dataset must be serialized into XML before it can be consumed by web services. Conversely, the XML must be converted back to a dataset for use in the application.

GetDSfromXML/GetXmlFromDS are methods coded by the programmer and could look something like this:

```

public static DataSet GetDSfromXML (System.Xml.XmlNode pXmlDataDocument)
{
    DataSet sourceDS = new DataSet();
    System.IO.StringReader oRdr = null ;
    try
    {
        oRdr = new System.IO.StringReader(pXmlDataDocument.OuterXml);
        sourceDS.ReadXml(oRdr);
        //return sourceDS;
    }
    catch
    {
        sourceDS = null;
        oRdr = null;
    }
    return sourceDS;
}

```

```

public static string GetXmlFromDS(DataSet pSourceDS)
{
    try
    {
        return pSourceDS.GetXml();
    }
    catch
    {
        return "";
    }
}

```

The EnergyAPIWebService object can be created in one of two ways:

Using WSDL:

(example only; change elements (paths, folders etc) as required)

```

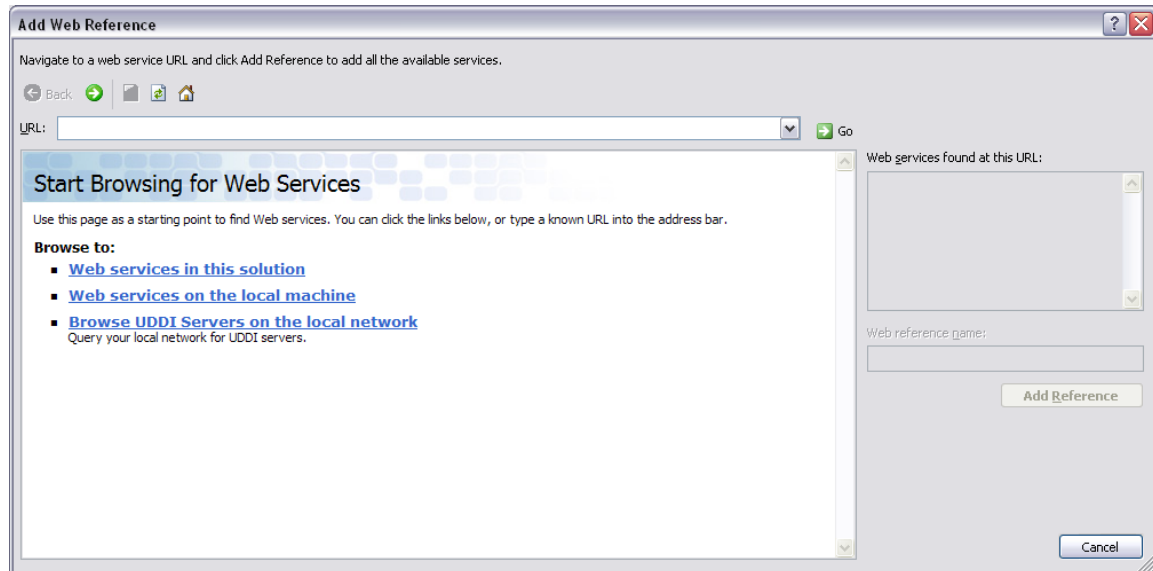
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cCustomerInfo.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/CCustomerLog.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cCustomerMaint.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cDelivery.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cLookUp.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cService.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/CTransaction.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/CUserProfile.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cUtility.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cPlanPrice.asmx
"C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\wsdl.exe" /language:CS
/n:EnergyAPIWebService http://localhost/EnergyAPIWebService/cCustBatch.asmx"

```

The above example displays the WSD Ling of all possible smartconnect branches. It is not necessary to WSDL all objects but rather only those that are required. These objects must also be added manually to your VS project.

Adding a Web Reference (via VS IDE)

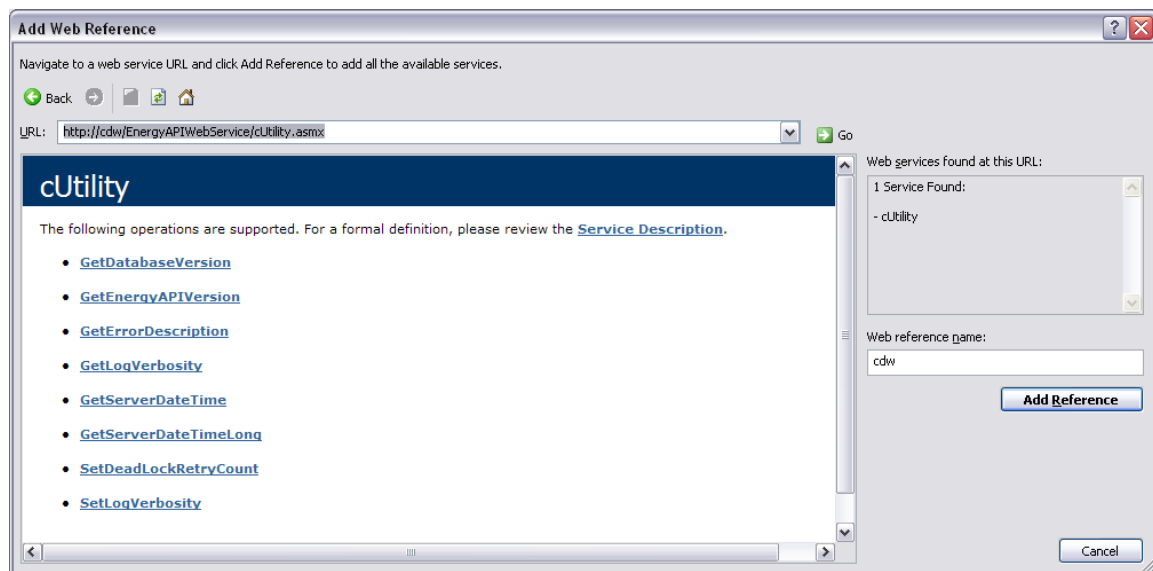
Right-click “References” and choose “Add Web Reference”. The following screen will be displayed:



URL: enter the http address of the web service. Click “Go”.
(eg: <http://ip address/EnergyAPIWebService/cUtility.asmx>)

Note: “EnergyAPIWebService” is a recommended name but it can be whatever is deemed appropriate (created during deployment of the smartconnect Web Service in IIS).

The following screen will be displayed



Click “Add Reference”.

This must be repeated for all smartconnect branches that are required for the project.

Appendix B

SmartConnect Templates

The following section lists the fields that are currently part of the SmartConnect Customer Maintenance templates. The fields are broken down by template type.

Master (template –900)					
Field Name (table)	Description	Data type	Min Value/Possible Values	Max Value/Stored Values	
acquisition_date	Acquisition Date	Date	12311960	06062049	
last_name	Last Name	String			
street1	Street 1	String			
street2	Street 2	String			
city	City	String			
state	State	String			
postal_code	Zip Code	String			
sort_code	Sort Code	String			
telephone	Telephone #	String			
general_text	General Text	String			
division	Division	Small Integer	1	9999	
type	Customer Type	Small Integer	1	7	
category ***	Category Code	Small Integer			
sales_person ***	Salesman	String			
keyoff_account	Open Item	String	Yes No	Y N	
credit_line	Credit Line Dollars	Long Integer	0	8350000	
credit_time_code	Credit Time-limit Code	Small Integer	0	9	
overage_dollar_amount	OverAge Limit Dollars	Long Integer	0	8350000	
collector_num ***	Collector	Small Integer			
finance_charge	Finance Charge	String	Yes No	Y N	
instant_invoice	Instant Invoice	String	Yes No	Y N	
instant_stat	Instant Statement	String	Yes No	Y N	
stat_day	Other Statement Days	Small Integer	0	160	
acquisition_reason_code	Acquisition Reason Code	Small Integer	0	255	
group_code_1	Group Code 1	Long Integer	0	999999	
company_form_code	Company Code For Forms	Small Integer	0	255	
tax_jurisdiction	Account Tax County	Small Integer	1	255	
budget_adjustment_dollars	Budget Adjustment Dollars	Currency	-99999.99	99999.99	
budget_start	Budget start month	Small Integer	1	12	
budget_payment	Budget payment amount	Currency	0	214700.00	
num_budget_payments	Number of budget payments	Small Integer	1	12	
statement_message_code	Statement Message Code	Small Integer	0	9999	
tax_1	Account tax 1	String	Yes No	Y N	
tax_2	Account tax 2	String	Yes No	Y N	
tax_3	Account tax 3	String	Yes No	Y N	
title	Title	String			
first_name	First Name	String			
name_suffix	Name Suffix	String			
service_in_budget	Service contract in budget	String	Yes No	Y N	
ivo_account	Invoice flag	String	Yes No	Y N	
cum_finance_charge	Cumulative finance charge	Currency	0	83500.00	
finance_charged_days	Finance charge days	Small Integer	0	155	
priority_account	Priority account	String	Yes No	Y N	

	budget_start_year	Budget Start Year	Small Integer	0	2049
	budget_dollars_billed	Budget Dollars Billed	Currency	-99999.99	4999999.99
1	balance	Account Balance	Currency	-9999999.99	99999999.99
1	data_conversion	Data Conversion	String	Yes No	Y N
	canadian_postal_code	Canadian Postal Code	String		
	credit_limit_code	Credit Dollar Limit code	Small integer		
	finance_charge_group	Finance Charge Group		0	52
	language	Language	String	English/French	E/F
	next_year_volume	Next Years Volume			
	no_dunning	Exempt from dunning	String	Yes/No	Y/N
	telephone	Telephone	String	9999999999	
	term_id	Term Code	Short Integer		
	ytd_units	Year to date gallons	Float		

***: These fields are "lookup" fields and should be presented to the end user as dropdowns. It is important to note that the list for these fields could vary (additions/deletions done through Energy client) and therefore it is highly recommended that the method `cLookup.GetLookupStoredValues` (page 97) be called for these fields.

1: balance/data_conversion – balance is a read-only field however if the 'data_conversion' field is set to 'Y', the balance field will be treated as non read-only and if applicable (old value <> new value or new customer being created) the value will be saved.

Delivery (template –901)				
Field Name (table)	Description	Data Type	Min Value/Possible Values	Max Value/Stored Values
delivery_address *1	Delivery Address	String		
delivery_instruction	Delivery Instructions	String		
zone	Zone	String		
fill_location	Fill Location	String		
product	Product Code	Small Integer		
base_price_code	Base Price Code	Small Integer	0	200
deviation	Deviation	Float	-9999.9999	9999.9999
deviation_date	Current Deviation Date	Date	12311960	06062049
size	Tank Size	Long Integer	0	999999
usable	Usable Gallons	Long Integer	0	999999
reserve	Reserve Gallons	Long Integer	0	32767
k_factor	K factor or Days between delivery or Route Table number	Float	0.0000	327.6700
hot_water_gallons	Hot water gallons/month	Long Integer	0	32767
separate_hw_heater	Separate hot water heater	String	Yes No	Y N
degree_day_next	Degree day next delivery	Long Integer	0	32767
commercial	Commercial account	String	Yes No	Y N
period_tank	Period account	String	Yes No	Y N
route_table_tank	Route table account	String	Yes No	Y N
will_call_tank	Will call	String	Yes No	Y N
tax_jurisdiction	Tank county code	Small Integer	1	255
tax_1	Sales tax	String	Yes No	Y N
tax_2	Tank tax 2	String	Yes No	Y N
tax_7	State road tax	String	Yes No	Y N
tax_8	Federal road tax	String	Yes No	Y N
tank_acquired	Tank acquired	Date	12311960	06062049
acquisition_reason ***	Acquisition reason	Small Integer		
price_sub_level	Base price sub-level code	Small Integer		
dd_table	Degree Day Table	Small Integer	0	255
tax_5	Tank tax 5	String	Yes No	Y N
tax_4	Tank tax 4	String	Yes No	Y N
tax_3	Tank tax 3	String	Yes No	Y N
zip	Delivery Zip Code			
tank_category_1	Category One			
delivery_notification	Delivery Notification	String	Yes No	Y N
dad_city ²	City	String		
dad_country ²	Country	Small Integer	USA Canada	0 1
dad_first_name ²	First Name	String		
dad_last_name ²	Last Name	String		
dad_latitude ²	Latitude	Float	-180	360
dad_longitude ²	Longitude	Float	-180	360
dad_middle_initial ²	Middle Initial	String		
dad_name_suffix ²	Name Suffix	String		

dad_postal_code²	Postal Code	String		
dad_state²	State	String		
dad_street1²	Delivery Street1	String		
dad_street2²	Delivery Street2	String		
dad_street3²	Delivery Street3	String		
dad_title²	Title	String		
dont_price_meter	Don't Auto-price ticket	String	Yes/No	
Last_delivery_date	Last delivery date	date		
tank_status	Status Flag	String*1	Active/Inactive	A/I
tax_11	Tank tax 11	String	Yes/No	Y/N
tax_12	Tank tax 12	String	Yes/No	Y/N
tax_13	Tank tax 13	String	Yes/No	Y/N
tax_9	Tank tax 9	String	Yes/No	Y/N
daily_usage_rate	Usage Rate per Day	Decimal	0	9999.99
Delivery_interval	Days Between Delivery	Integer	0	327
its_qoh	ITS Quantity on Hand	Short Integer	-32767	32767
date_last_inventoried	Last Inventory Date	Date		
its_product	ITS Product	Short Integer		
tank_category_2	Category 2	Short Integer		
category	Tank Category	Short Integer		
po_expiration	PO Expiration	Date		
po_required	PO Required	String	Yes/No	
po_number	PO Number	String*25		
dont_price_meter	Dont auto-price ticket	String	Yes/No	
elec_in_tank_units	Last Electronic in tank units	Float		
elec_pending_delivery	Electronic-Potential pending delivery flag	String	Yes/No	
elec_percent_reading	Last Electronic percent reading	Float		
elec_tank_read_date	Electronic tank reading date	Date		
exempt_from_mar	Exempt from Minimum Annual Requirement	String	Yes/No	
key_code	Key code	String		
key_required	Key required	String	Yes/No	
misc_charge_remote_invoice	Apply Misc. Charge on Remote Invoice	String	Yes/No	
sales_type	Sales Type	Small Integer		
snow_drop	Snow drop	String	Yes/No	
special_ticket_handling	Special Ticket Handling	String	Yes/No	

*** : These fields are "lookup" fields and should be presented to the end user as dropdowns. It is important to note that the list for these fields could vary (additions/deletions done through Energy client) and therefore it is highly recommended that the method `CLookup.GetLookupStoredValues` (page 97) be called for these fields.

*1 : delivery address field will be rendered "read-only" if any of the parsed address fields (all dad_* fields) contains data. This is to maintain the business rules currently in place in the Energy client application. Ideally the delivery address field should not be presented to the end user if using the parsed fields.

²: when updating an existing record, if any parsed field is updated, **ALL** parsed fields, whether updated or not **must be included** in the request/dataset to prevent loss of data.

Service (template –902)					
Field Name (table)	Description	Data Type	Min Value/Possible Values	Max Value/Stored Values	
service_address ^{*1}	Service address	String			
service_instruction	Service instructions	String			
zone	Service zone	String			
service_contract	Service contract type	String			
contract_billing_period	Service contract renewal period	String			
contract_renewal_date	Service renewal date	Date			
contract_deviation	Service contract deviation (+/- \$)	Float	-9999.9900	9999.9900	
tax_jurisdiction	Service county code	Small Integer	1	255	
labor_tax	Labor taxable	String	Yes No	Y N	
part_tax	Parts taxable	String	Yes No	Y N	
contract_tax	Contract taxable	String	Yes No	Y N	
contract_base_price	Contract Base price code	Small Integer	0	99	
contract_sub_level	Contract Sub-Level	Small Integer	0	255	
last_clean_date	Last cleaning date	Date			
tank_warr_amount	Tank warranty amount	Float	0	9999.9900	
tank_warr_date	Tank warranty renewal date	Date	12311960	06062079	
Category ^{***}	Service Category	Small Integer			
service_location	Service Location Definition				
sad_city ²	City				
sad_country ²	Country	Small Integer	USA Canada	0 1	
sad_first_name ²	First Name	String			
sad_last_name ²	Last Name	String			
sad_latitude ²	Latitude	Float	-180	360	
sad_longitude ²	Longitude	Float	-180	360	
sad_middle_initial ²	Middle Initial	String			
sad_name_suffix ²	Name Suffix	String			
sad_postal_code ²	Postal Code	String			
sad_state ²	State	String			
sad_street1 ²	Service Street1	String			
sad_street2 ²	Service Street2	String			
sad_street3 ²	Service Street3	String			
sad_title ²	Title	String			
part_base_price_code	Inventory base price	Short Integer			
department_code	Department Code	Short Integer			
acquired	Acquisition Date	Date			
contract_tax_2	Contract Tax 2	String	Yes No	Y N	
contract_tax_3	Contract Tax 3	String	Yes No	Y N	
contract_tax_4	Contract Tax 4	String	Yes No	Y N	
contract_tax_5	Contract Tax 5	String	Yes No	Y N	
last_contract_bill_date	Last Bill Date	Date			
initial_contract_date	Initial Contract Date	Date			
contract_price	Contract Price	Float			
contract_tax_amount	Contract Tax Amount	Float			
contract_total_price	Contract Total Price	Float			

***: These fields are "lookup" fields and should be presented to the end user as dropdowns. It is important to note that the list for these fields could vary (additions/deletions done through Energy client) and therefore it is highly recommended that the method cLookup.GetLookupStoredValues (page 97) be called for these fields.

*¹: service address field will be rendered "read-only" if any of the parsed address fields (all sad_* fields) contains data. This is to maintain the business rules currently in place in the Energy client application. Ideally the service address field should not be presented to the end user if using the parsed fields.

*²: when updating an existing record, if any parsed field is updated, **ALL** parsed fields, whether updated or not **must be included** in the request/dataset to prevent loss of data.

WDMS (template –903)					
Field Name (table)		Description	Data Type	Min Value/Possible Values	Max Value/Stored Values
last_name		Last name	String		
street1		Location street 1	String		
street2		Location street 2	String		
town		Location town	String		
state		Location state	String		
zip		Location zip code	String		
phone_number		Location phone #	String		
route***		Route Code	String		
base_price_code		Location price category	Small Integer	1	99
terms_rate		Loc prompt pay %/cents	Float	-99.00	99.00
terms_days		Location prompt pay days	Small Integer	0	99
terms_net_days		Location net days	Small Integer	-1	127
price_wo		Location price Bill of Lading	String	Yes No	Y N
default_whse***		Location default warehouse	String		
tax_jurisdiction***		Location tax county	Small Integer		
tax_1		Location sales tax	String	Yes No	Y N
tax_2		Location tax 2	String	Yes No	Y N
tax_7		Location state excise tax	String	Yes No	Y N
tax_8		Location federal excise tax	String	Yes No	Y N
invoice_special_handling		Special Invoice Handling	String	Yes No	Y N

***: These fields are "lookup" fields and should be presented to the end user as dropdowns. It is important to note that the list for these fields could vary (additions/deletions done through Energy client) and therefore it is highly recommended that the method `cLookup.GetLookupStoredValues` (page 97) be called for these fields.

Transaction Upload Data (DSARTransactions)

TRANS_MAIN Fields

Field	Type	Default Value	Required	Description
account_num	int		Y	Account number
accounting_date	smalldatetime		Y	Accounting date
acquisition_payment	ADD_YN			Future use
adjustment_code	smallint			System-defined code for the adjustment type: 1 = Reverse; 2 = ReverseNSF; 3 = Transfer; 4 = Rebill; 5 = Writeoff; 6 = Refund; 7 = Balance Transfer; 8 = Inventory Transfer
adjustment_flag	ADD_YN			Transactions posted in MCAP (Y/N)
batch_num	Smallint			Batch number
card_driver_id	varchar			Driver ID
card_hour	tinyint			Hour at which product was pumped
card_minute	tinyint			Minute at which product was pumped
card_num	int			Customer's card number
card_odometer	int			Odometer reading which may have been entered at pump
card_site_num	smallint			Card site number
card_trx_num	int			Card transaction number
card_veh_num	varchar			Customer's vehicle number
category	smallint			Customer category (CAT) at time of transaction
cc_authorization	varchar			Credit card authorization
cc_expiry_dt	smalldatetime			Credit card expiration date and time
cc_name_on_card	varchar			Name on credit card
cc_number	varchar			Credit card number
cc_type	smallint			Type of credit card (e.g., VISA, MasterCard)
closed_ko_trx_flag	ADD_YN			For Open Item account, is item closed (Y/N)
comment	varchar			100-character field for free form comments
comment_num	smallint			Internal number assigned to predefined comment
cost_type	smallint			Optional code available in Energy components Price Protection Plans and Customer Contracts. This code allows you to track different product costs at the transaction level.
created_by	varchar			Initial user to insert the row in the Sybase table

curr_mtr_rdg	money			Current meter reading
degree_day	int			Degree day of transaction
delivery_reason	smallint			Predefined delivery reason code set up in System Maintenance
delivery_type	tinyint			This value is calculated by assessing the tank setup at time of delivery. Delivery types are: 0 = automatic K-factor, 1 = automatic period, 5 = automatic route table, 2 = will call K-factor, 3 = will call period, 7 = will call route table
department_code	smallint			Optional code available in Energy components Price Protection Plans and Customer Contracts allowing you to categorize sales by departments
deviation	small			DEV – Price deviation maintained in Customer Maintenance
discount_amt	money			Prompt payment discount
division	smallint			DIV – Customer division (DIV) at time of transaction
driver_num	smallint			DRI – A system-defined number identifying the driver
due_by_date	smalldatetime			Pay by this date to receive discount
eft_due_date	smalldatetime			The date that the EFT payment is due to be drafted (for EFT customers only)
eft_trx_status	char			The status of the EFT transaction (for EFT customers only) N=Non-EFT; U = Undrafted; P = Pending; D = Drafted; E=EFT Payment
embedded	ADD_YN			ADD Systems internal use only
end_time	smallint			Future use
event_date	smalldatetime		Y	Event date and time
gasboy_trx_flag	ADD_YN			Is this a Gasboy transaction? (Y/N)
gl_num	int			GL number
gross_amount	money			Gross dollar amount (does not include taxes)
gross_plus_embedded	money			The gross dollar amount plus the embedded taxes
in_tank_units	money			Units in tank after delivery, partial fills only
invoice_comment	varchar			100-character field for invoice comments

invoice_date	smalldatetime			Invoice date and time (for CARD transactions only). Assigned by system when FEECARD is run.
invoice_num	int			Invoice number (for CARD transactions only). Assigned by system when FEECARD is run.
its_prod_num	smallint			ITP – Inventory Tracking System product number
k_factor	smallmoney			KFA – K-factor of delivery
keyoff_column	smallint			ADD Systems internal use only
keyoff_flag	ADD_YN			Indicates that this account is keyoff (open item) (Y/N)
keyoff_no_more_flag	tinyint			ADD Systems internal use only
keyoff_row	smallint			ADD Systems internal use only
ko_statement_date	smalldatetime			Keyoff (open item) statement date. Assigned by system when running STAT, but only if open item detail is shown on the statement
last_maintenance_dt	datetime			The last date and time the row was updated (NOTE: If no value is provided in this field the date time will be defaulted to the current system date time)
last_maintenance_userid	char			The user name last used to update this row. It may or may not identify the person involved, depending on how Client and Host user names are established on your system. (NOTE: If no value is provided in this field, 'acct_api' will be used)
last_on_stat_flag	ADD_YN			Indicates that this transaction and all 'older' transactions have appeared on a customer statement. Applies to balance forward customers only.
letter_num	smallint			The dunning letter number last sent. This number corresponds to the letter number in the Dunning Letter Descriptions component. It will be displayed in the Credit Code column in Enter Credit Actions and in the Credit Action column in Credit Display. (smallint – 0 to +/- 32.767)
location_num	smallint			WDMS location number
miles	smallint			Future use
minimum_charge	ADD_YN			Flag (Y/N) that indicates if sale was billed at minimum charge amount or not
mtr_rdg_est	ADD_YN			Estimated meter reading (Y/N)
mtr_type	tinyint			Meter type as defined in the Meter Type Conversion Table
n_option_flag	ADD_YN			Indicates that the user entered "N" on the posting line in MDEL/POST when the transaction was entered
net_amount	ADD_AMT_2			Net dollar amount (includes taxes)
odometer	int			The odometer reading

open_amt	ADD_AMT_2			Dollars open on keyoff transaction
part_fill_reason	smallint			The reason for a partial fill
partial_fill	ADD_YN			Partial fill (Y/N)
payment_method	smallint			Payment method (e.g., check, ATM card, cash, etc.)
payment_method_info	smallint			Financial institutions for "Payment Method" field
payment_type	smallint			Identifies whether payment is from a regular customer or an energy assistance agency
payment_type_class	smallint			Payment type class
plan_price_id	int			A value of "0" indicates that the customer's regular price was used to calculate this transaction. Any other value indicates that the customer's plan price was used to calculate this transaction. This field can be manually joined to the PLAN_PRICE table via transaction_key when reporting price protection plan detail.
percent_fill	tinyint			Flag indicating that the customer is percent fill
po_number	varchar			The purchase order number
posting_code	smallint			Posting code on the transaction
posting_date	smalldatetime			Posting date and time
ppg	ADD_4_AMT_4			Price per gallon/liter
ppu	ADD_4_AMT_4			PPU – Price per unit (e.g., per cylinder, per meter unit)
prem_bill_flag	ADD_YN			Future use
prev_mtr_rdg	ADD_AMT_2			Previous meter reading
prior_meter_rdg_date	smalldatetime			Prior meter reading date and time
publication_price	ADD_YN			PUB – Publication pricing set on customer maintenance template (Y/N)
raven_message_number	smallint			Message number on the Raven hand held unit
reference_num	int		Y (must be > 0)	Reference number (ticket, invoice, check number, etc.)
runout	ADD_YN			Was delivery a runout (Y/N)
runout_reason	smallint			One of a list of predefined reason codes for runout
salesperson	smallint			The salesperson on the account
sales_type	char			Optional code available in Energy components Price Protection Plans and Customer Contracts allowing you to identify transactions by sales type
separate_taxes_exist	ADD_YN			Are there separate tax transactions (Y/N)
serial_num_entered	ADD_YN			Was serial number for Equipment system entered (Y/N)
service_num	smallint			Service number
skip_forecast	ADD_YN			Skip forecast (Y/N). This flag can be set in Host program KFIX. When set to "Y", it causes the Host forecasting programs to skip this transaction. The default is "N".

start_time	smallint			Future use
statement_message_code	smallint			The statement message code for this customer
status	char			The row status; "A" = Active, "I" = Inactive
tank_num	smallint			Tank number
tax_jurisdiction	smallint			Tax county
tax_t4pct_10	smallint			ADD Systems internal use only
term_id	smallint			User-defined code set up in the Terms Code Table used to define the payment terms for an account
TRANS_MAIN_lock	tinyint			ADD Systems internal use only
transaction_inserted	timestamp			Date and time transaction physically posted
truck_num	smallint			Truck number
trx_inserted	datetime	Will be generated in Energy		The date and time that the transaction was inserted into the database
trx_type	char	Will be generated in Energy		D = debit transaction; C = credit transaction
trx_unique_key	int	Will be generated in Energy		ADD Systems internal use only
type	smallint			TYP– Customer type (1 – 7)
units	ADD_AMT_2			Number of units (gallons, liters, etc.) This field cannot be used in VCQ, but <u>must</u> be used by all other reporting mechanisms – e.g., MS Access, Excel, etc.)
units_returned	smallint			Units returned (<i>ITS only</i>)
units_used	ADD_AMT_2			Units sold to customer, e.g., ITS, cylinder or meter reading sales
x_option_flag	ADD_YN			Suppress invoice (Y/N)
data_conversion	ADD_YN	N	Y	Indicates that load is part of a conversion load process. Originally this field was related to data conversions however, with relation to the “UploadTransactions” method, if you want to, for whatever reason, update the “balance” field you must set this field to “Y”.
update_balance	ADD_YN	Y	Y	Determines if the load will update the account balance based on this transaction. Note: In Smartconnect V14.01.01 and higher, this field MUST BE set to “Y” if the data conversion is different from “Y”. In older versions of Smartconnect (V14.01.00 and older), It was not enough to set this field to “Y” to get the balance updated, you must also set “data_conversion” to “Y”.

TRANS_DELIVERY Fields

Field	Type	Default Value	Required	Description
account_num	int			The account number
actual_seq	smallint			The delivery sequence for RAVEN
daily_usage_rate	ADD_4_AMT_2			The gallons per day if the customer is URD
day_of_week	smallint			The day of week schedule if the customer is DOW
dd_ahead_or_behind	smallint			The degree days ahead or behind the projected next degree day for KFA customers
degree_day_next	int			The system-calculated degree day next for KFA customers
delivery_interval	smallint			The days between delivery if this is a period customer
delivery_seq_number	smallint			The delivery sequence number from Select Tickets
delivery_type	tinyint			This value is calculated by assessing the tank setup at time of delivery. Delivery types are: 0 = automatic K-factor, 1 = automatic period, 5 = automatic route table, 2 = will call K-factor, 3 = will call period, 7 = will call route table
end_time	smallint			The delivery stop time for RAVEN
hot_water_gallons	int			The number of hot water gallons for this tank
k_factor	ADD_4_AMT_4			K-factor of delivery
k_factor_locked	ADD_YN			KLK flag value at the time of delivery (Y/N)
last_maintenance_dt	datetime			The last date and time the row was updated
last_maintenance_userid	char(20)			The user name last used to update this row. It may or may not identify the person involved, depending on how Client and Host user names are established on your system.
next_delivery_date	smalldatetime			The next date that the customer is due for delivery
phone_order	char(1)			Phone order (Y/N)
phone_order_units	ADD_AMT_2			Number of units ordered on the phone order
proj_days_in_reserve	smallint			The projected number of days the customer will have enough fuel until runout
proj_days_to_due	smallint			The number of days projected by the system until the customer will be due for delivery
proj_degree_day_runout	int			The degree day the customer is projected to runout

pproj_delivery_units	ADD_AMT_2			The number of units projected for the next delivery
proj_runout_date	smalldatetime			The date the system projects the customer will runout
reserve_gallons	int			The number of gallons in reserve for this customer
route_table	smallint			The number of the route table used for this customer, if the customer is a route customer
separate_hw_heater	ADD_YN			Customer has separate hot water heater (Y/N)
size	int			The tank size
start_geocode	decimal(9,6)			Future use
start_time	smallint			The delivery start time for RAVEN
status	char(1)			The row status; "A" = Active, "I" = Inactive
ticket_out	ADD_YN			Ticket out flag value at the time of delivery (Y/N)
ticket_seq_number	smallint			Ticket sequence number in Select Tickets
TRANS_DELIVERY_lock	tinyint			ADD Systems internal use only
trx_unique_key	int			ADD Systems internal use only
usable	int			The useable gallons
variable_hotwater_usage	ADD_YN			Variable hot water usage (Y/N)
zone	char(10)			The delivery zone code; alphanumeric, 10 chars

Appendix C

All Return Codes & Descriptions

- 52 No user found.
- 96 Unknown error occurred
- 101 Unable To Establish A Connection - Please Make Sure that the file IVRInterface.ini exists in the proper directory.
- 102 Invalid database connection requested
- 103 Unable to locate a unique account number. No account was found
- 104 Unable to create a log header
- 105 Unable to create a log detail
- 106 Unable to create a phone order
- 107 Phone order not found
- 108 Unable to reschedule phone order
- 109 A User Profile already exists for this email address.
- 110 A link already exists to this company/account number.
The answer to the Password Hint did not match. Unable to retrieve the
- 111 password.
- 112 Login failed. Invalid login credentials supplied.
- 113 Invalid expiration month
- 114 Invalid expiration year
- 115 Unable to get a new batch number
- 116 Unable to create a payment batch header
- 117 Invalid payment posting code
- 118 Unable to create the payment detail
- 119 Log Verbosity code should be a number between 0 and 4
- 120 Invalid Tank number/Product code
- 121 Invalid product code
- 122 Invalid input parameters
- 123 Unable to get next sequence number - Please call support
- 124 Password must be at least 6 characters long and at most 20 characters long.
- 125 Invalid Customer LOG Template ID. Please try again.
- 126 Invalid zip code entered. Please try again
- 127 Nothing to save. Please try again
- 128 The total length of the Name including the spaces and punctuation cannot exceed 40 characters
- 129 Unable to create the account location. Please try again
- 130 You've reached the maximum number of account locations. Please call your database administrator.
- 131 Unable to update the location text. Please try again.
- 132 The record has already been modified by another user. Please refresh the data and try again.

- 133 Unable to create the account. Please try again.
134 You have exceeded the maximum combined length of 87 characters for the fields Name, Street1, Street2, and/or Town, State. Try again.
135 Unable to update the general text record for the account. Please try again.
136 Unable to update the delivery address record for the account. Please try again.
137 Unable to update the delivery instruction record for the account. Please try again.
138 Unable to update the service address record for the account. Please try again.
139 Unable to update the service instruction record for the account. Please try again.
140 Unable to update the service location definition record for the account. Please try again.
141 The field is required. Please enter a new value and try again
142 The value is outside the range of allowed values. Please enter a valid field value.
143 Unable to calculate the estimated inventory/Days in reserve.
144 Unable to locate a unique account number. More than 1 account was found
145 ERROR: This Customer does not exist in Pegasus.
146 ERROR: Failed to update the Customer in Pegasus.
147 ERROR: Failed to create the Call Note in Pegasus.
148 ERROR: Failed to create the Service Remarks in Pegasus.
149 ERROR: Failed to create the Call Reason in Pegasus.
150 ERROR: Failed to create the Service Call in Pegasus.
151 ERROR: This call number already exists in Pegasus.
152 ERROR: Failed to create the Service Zone in Pegasus.
153 ERROR: The Customer table in Pegasus is full.
154 ERROR: Failed to create the Customer in Pegasus.
155 ERROR: The Call or Technician have been modified by another user. Please re-submit changes.
156 ERROR: Failed to update the Technician in Pegasus.
157 ERROR: Failed to remove the call from the Technician's Workload in Pegasus.
158 ERROR: Failed to update the Technician's Workload in Pegasus.
159 ERROR: Failed to update the Service Call in Pegasus.
160 ERROR: Failed to update the Call Reason in Pegasus.
161 ERROR: Failed to update the Call Note in Pegasus.
162 ERROR: Failed to cancel the Service Call in Pegasus.
163 ERROR: The Service Call has already been canceled in Pegasus.
164 ERROR: This Call does not exist in Pegasus.
165 ERROR: Cannot cancel - work has been done in Pegasus.
166 Warning: The ETA is outside the scheduled time.

- 167 Call could not be scheduled and/or re-scheduled in the Calendar - no Calendar was found.
- 168 Table SERVICE_DISPATCH is full
- 169 Parameter W/O number sent by caller is < 1
- 170 NT Calendar not available for specific company/call reason/zone.
- 171 All workorders are in use on the service record.
- 172 This call was not found on the service record. Another user may have modified it.
- 173 The From Date must be less than the To Date.
- 174 The From Date cannot be less than Today.
- 175 The call is already canceled and/or does not exist.
- 176 The call reason is invalid for the account/service location.
- 177 A call with "on-site"/"completed"/"Interrupted" status cannot be canceled/rescheduled.
- 178 Call reasons 1,2 and/or 3 must be unique.
- 179 Call Reason one must be greater than 0.
- 180 Cannot reschedule a canceled/invalid work order.
- 181 No scheduling slots are available for preferred date and time ranges.
- 182 Could not determine the scheduling method associated with company.
- 183 The From Date must be greater than Today.
- 184 Call Reason must be between 1 and 255.
- 185 Invalid "From Date" and/or "To Date" supplied. Values are outside acceptable range as defined in Energy. Please verify.
- 186 The length of the service note exceeds 255 characters. Please try again.
- 187 The length of the zone cannot be less than 1 or exceed 10 characters. Please try again.
- 188 Invalid Zone supplied. Please try again.
- 189 Unable to update Service Wait Time information.
- 190 Unable to insert Service Wait Time information.
- 191 Invalid Log File Path, the length should be between 1 and 100 characters
- 192 Invalid Log File Name, the length should be between 1 and 100 characters
- 193 Invalid row status entered, should be A or I.
- 194 You have exceeded your ADD Energy license count. Please try again later.
- 195 Cannot retrieve the application license information. Please call support.
- 196 Cannot update the license count. Please call support.
- 197 Invalid License key detected. Please call support.
- 198 The selected feature requires a valid license.
- 199 You have exceeded the maximum number of items for the current feature. Closing the window without deleting some items will disable the component. Are you sure that you want to close the window?

- 200 You have exceeded the maximum number of items for the selected feature. The feature will be disabled until the number of items is within the licensed number of items. Please call support.
- 201 The starting and ending dates must be within the minimum and maximum dates on the plan.
- 202 The price must be between the minimum and maximum price on the plan.
- 203 A valid master account number is required for a master plan.
- 204 The selected plan type is not supported by the selected plan code.
- 205 The starting values must be greater than or equal to their corresponding remaining values (dollars, units, number of deliveries, ...). The end date must be greater than the starting date.
- 206 The master account number cannot be the same as the account number.
- 207 A master account number is not permitted for a non-master plan.
- 208 Invalid customer number.
- 209 The alternate product cannot be the same as the product/ if you choose all products, you must choose no alternate products.
- 210 You cannot choose all products and all tanks.
- 211 This functionality is not available at this release.
- 212 Invalid event date.
- 213 Invalid accounting date.
- 214 Invalid posting date.
- 215 Invalid posting code.
- 216 Invalid reference number.
- 217 The event date and the accounting date must be within 125 days of each other.
- 218 The posting date and the event date must be within 127 days of each other.
- 219 Unable to update account record -- transaction skipped.
- 220 Unable to create transaction record -- transaction skipped.
- 221 Unable to create trans delivery record -- transaction skipped.
- 222 Unable to get new transaction key -- transaction skipped.
- 223 Invalid division.
- 224 Invalid type.
- 225 Invalid transaction reference number, the valid range is from 1 to 8388350
- 226 Invalid last 4 digits for Credit Card.
- 227 Invalid number of digits in Credit Card number.
- 228 Invalid Customer Vault Id.
- 229 Unable to unlink this company/account number combination.
- 337 Another user has already modified the selected information. Press OK to refresh this information then redo the action you just performed.
- 504 No data was found. Change the selection criteria and try again.

- 521 This data has been changed by another user. Click View/Refresh to reset the data, then re-enter your changes.
- 565 This data has been changed by another user, please refresh the data and try again.
- 578 An entry already exists, please enter a unique combination of values.
- 899 For each contact type, only one entry can be designated as the primary contact at the customer or location level. Please try again.
- 605 The Accounting date must be greater than the Cutoff date.
- 606 The Accounting date must be less than or equal to the Posting date.
- 607 The Event date must be greater than the Accounting minus the maximum day limit established on this database
- 608 The Event date must be less than the Accounting date plus the maximum day limit established on this database
- 788 There is a problem inserting data into a table. The process did not complete. Call your System Administrator.
- 831 Customer is terminated.
- 949 Energy was unable to process your request because the Reinstate Reason code is invalid. Please contact ADD Systems Customer Support.
- 980 The Delivery date cannot be more than 30 days prior to Today.
- 981 The value of delivery units exceeds the Usable.
- 984 The date cannot be earlier than today.
- 985 There is currently a ticket out. Do you want to cancel it?
- 1011 There is a problem updating data in a table. The process did not complete. Call your System Administrator.
- 1023 Automatic forecasting unavailable - No previous delivery records on file.
- 1025 Cannot reforecast - next delivery date less than or equal to previous delivery.
- 1073 This customer's Must Charge Credit Card flag is set to Yes. You must pre-authorize the charge for this phone order
- 1091 Electronic Tank Reading not allowed on this tank.
- 1092 Electronic Tank Reading Date cannot be greater than the current date.
- 1093 Invalid Electronic Tank Reading Date.
- 1094 Invalid Electronic Tank Reading Value.
- 1135 An entry already exists with this effective date and time.
- 1300 Customer number does not exist, please review.
- 1301 Customer tax jurisdiction does not exist.
- 1302 Cost center is required.
- 1303 Raven SLPP dollars differs from the current SLPP dollars. The current SLPP dollars will be used.
- 1304 Input product 1 does not exist.
- 1305 Input product 2 does not exist.
- 1306 Input product 3 does not exist.
- 1307 Input product 4 does not exist.

- 1308 Input product 5 does not exist.
- 1309 Input product 6 does not exist.
- 1310 Input volume 1 is not within the valid range.
- 1311 Input volume 2 is not within the valid range.
- 1312 Input volume 3 is not within the valid range.
- 1313 Input volume 4 is not within the valid range.
- 1314 Input volume 5 is not within the valid range.
- 1315 Input volume 6 is not within the valid range.
- 1316 Invalid cost center.
- 1317 Invalid delivery type.
- 1318 Invalid meter product.
- 1319 Invalid metered tank.
- 1320 Invalid pre-defined comment.
- 1321 Invalid serial number.
- 1322 Invalid volume delivered to non-inventory location.
- 1324 Inventory location does not exist.
- 1325 Method info does not exist.
- 1326 Non-Productive hours are required and no value is entered.
- 1327 Non Productive Reason is required and no value is entered.
- 1328 Output product does not exist.
- 1329 Output volume is not within the min/max range.
- 1332 Prime Starts is required and no value is entered.
- 1333 Product does not exist.
- 1334 Productive hours is required and no value is entered.
- 1336 Reference number was not entered and is required.
- 1337 Selected truck does not exist.
- 1338 Selected truck has inactive status.
- 1339 Selected warehouse has inactive status.
- 1340 Supplier code does not exist.
- 1341 Tank number not found for this customer.
- 1342 Tax jurisdiction does not exist.
- 1343 The position entered doesn't exist for the warehouse.
- 1344 Ticket Number does not match (any) ticket number(s) for the Customer/Delivery Location.
- 1345 Ticket number required for keyoff customer.
- 1346 Totalizer # has not been entered and is required.
- 1347 Totalizer is not within the min/max range.
- 1348 Unit of measure does not exist.
- 1349 Volume delivered is not within the min/max range.
- 1350 Volume has not been entered and is required.
- 1351 Volume is not within the min/max range.
- 1352 Warehouse code does not exist.
- 1353 End Odometer is required and has not been entered.
- 1354 End Odometer must be equal to or greater than Begin Odometer.
- 1355 End Totalizer 1 must be equal to or greater than Begin Totalizer 1.
- 1356 End Totalizer 2 must be equal to or greater than Begin Totalizer 2.

- 1357 End Totalizer 3 must be equal to or greater than Begin Totalizer 3.
- 1358 End Totalizer 4 must be equal to or greater than Begin Totalizer 4.
- 1359 Fuel Consumption is required and has not been entered.
- 1360 Odometer difference doesn't match Odometer Begin and Odometer End.
- 1362 Totalizer 1 difference doesn't match Totalizer 1 Begin and Totalizer 1 End.
- 1363 Totalizer 1 End is required and has not been entered.
- 1364 Totalizer 2 difference doesn't match Totalizer 2 Begin and Totalizer 2 End.
- 1365 Totalizer 2 End is required and has not been entered.
- 1366 Totalizer 3 Difference doesn't match Totalizer 3 Begin and Totalizer 3 End.
- 1367 Totalizer 4 Difference doesn't match Totalizer 4 Begin and Totalizer 4 End.
- 1368 Totalizer 4 End is required and has not been entered.
- 1369 Totalizer 3 End is required and has not been entered.
- 1370 Charge Amount is required and has not been supplied.
- 1373 A position has been specified without a warehouse.
- 1374 The quantity delivered must not exceed the tank size.
- 1375 Raven unit price differs from the current tank unit price. The Raven price will be used to price this delivery.
- 1376 The parameters are not valid for the plan type. The starting / remaining values for the units, dollars, or number of deliveries options must correspond to the type of plan being added.
- 1377 Cannot save changes! One or more Host user IDs is the same as the Client user ID, including matching upper case values. Please make all Host user IDs unique.
- 1378 The combination of hauler, product classification, vehicle profile, origin and destination already exists.
- 1379 There is a unit price of 0 in your order. Do you wish to continue?
- 1380 System low in resources. Please close some application windows.
- 1381 System critically low in resources. Opening a new application window may cause the application to become unstable. Please close some application windows and try again.
- 1382 System critically low in resources. The application could become unstable. Please close some application windows.
- 1383 There are no stops with Geo-Coded locations to show on the map.
- 1384 Unable to create a Direct-to-CRED order because your computer date does not match the master file date.
- 1385 A deletion of Load(s) or Distribution(s) is not allowed in a Splash Blend event.
- 1386 Loads/Distributions must be pre-selected when maintaining a Splash Blend event. Maintenance for those Events has been unselected.
- 1387 You can only select one delivery center when showing unassigned deliveries.

- 1388 The selected location is not active.
- 1389 Cannot log Scheduled Product phone order. There is already an existing ticket / order out.
- 1390 Delivery is currently suspended.
- 1391 The offset posting code is invalid for the selected product code.
- 1392 Outsourced document not currently available for viewing.
- 1393 Warning: Could not detect eServer Premium license, all Premium Jobs have been removed/disabled. Please contact ADD Systems to have the correct license(s) configured/installed.
- 1394 Unable to decrease maximum number of characters per line as it would truncate existing text.
- 1395 A flat rate part category has not been configured. Please contact ADD Systems Customer Support for assistance.
- 1396 The highlighted fields are normally checked in conjunction with Update A/R. Please review and adjust, if appropriate.
- 1397 Only ONE of the following options may be set to 'Yes': 'Hold for Direct Ship Type', 'Hold for Charge Delivery Type', 'Hold for Deposit Type', 'Hold for Special Order Type' or 'Hold for Lost Sale Type'.
- 1398 Equipment maintenance required.
- 1399 Equipment movement required.
- 1400 Gas check required.
- 1401 At least one technician must be assigned to the primary function code.
- 1402 This batch is a pre 13.01.03 batch and cannot be opened. Please cancel and reenter.
- 1403 Warning: This selection may generate too much data that may cause the system to run out of memory. Do you still want to proceed?
- 1404 Warning: Another transaction for the same \$ amount has been found on the same credit card within the timeframe of the duplicate check window. Do you want to continue processing this transaction?
- 1405 Another transaction for the same \$ amount might have been sent to the CC processing gateway within the timeframe of the duplicate check window. Please check transactions on the CC processing gateway and reconcile with E3 as needed.
- 1406 A service part that is designated with a posting code source of part must be assigned a non-liquid posting code. You may change the posting code to a non-liquid debit, or change the source to function code.
- 1407 You are attempting to use an invalid posting code in this call. Only non-liquid debit posting codes are permitted in Service Billing.
- 1408 This formula is being used in Maintain Products and cannot be deleted or deactivated.
- 1409 Warning: This order has at least one detail row with a quantity of zero. Do you wish to continue?
- 1410 The Quantity on Hand for the component product is not sufficient for production. Do you wish to proceed?

- 1411 Order date cannot be less / greater than 120 days from the original event date.
- 1412 Order date is before the Cutoff Date. Do you wish to continue?
- 1413 The order number you have entered is already in use in an order billing batch.
- 1414 Only open encumbered orders are eligible for billing.
- 1415 Orders in the dispatch system are not eligible for billing.
- 1416 Do you want to finalize the order?
- 1417 Order has no special name and address. Do you wish to create a special name and address?
- 1418 Do you want to update the unit prices using the new BPC?
- 1419 Order Modification is not allowed due to stage rules.
- 1420 The order number you have entered has already been billed.
- 1421 All printing must be completed prior to exiting the component.
- 1422 Do you want to delete all of the order lines?
- 1423 Order maintenance is not allowed for cash sales.
- 1425 Transaction is currently in an EFT batch and cannot be maintained.
- 1426 Transaction is currently waiting for adjustment approval and cannot be maintained.
- 1427 Batch cannot have multiple ship dates when driver performance totals have been entered.
- 1428 Please accept or reject the order being edited in Order Entry / Order Maintenance before deleting or exiting the Load Track event.
- 1429 Transactions exist with accounting dates between %% and %. Press <Cancel> and re-run JRNL/SJRNL to send these to the G/L system or press <OK> to continue with the Cutoff.
- 1430 Reversal transactions exist with accounting dates between %% and %. Press <Cancel> and re-run JRNL/SJRNL to send these to the G/L system or press <OK> to continue with the Cutoff.
- 1431 Warning, there is no A/P supplier cross reference record for the current receipt.
- 1432 There is no A/P supplier cross reference record for the current receipt. You must set one up in order to post this receipt.
- 1433 Another order is being created or edited. Please close the Order Entry or Order Maintenance screen and try again.
- 1434 The selected hauler office is not associated with the selected hauler. Please choose a valid hauler office.
- 1435 The selected supply location is not associated with the selected supplier. Please choose a valid supply location.
- 1436 Supply location and/or hauler office does not exist in the Pricing Market. Please select a valid supply location and/or hauler office for the Pricing Market.
- 1437 Do you wish to clear the service contract information?
- 1438 Do you wish to save and create an order with no order lines?
- 1439 There are pending deliveries / services marked to be charged to this credit card. Do you wish to continue?

- 1440 You are about to create a duplicate of this location. Are you sure you want to continue?
- 1441 A folder and/or function can no longer be created at this level.
- 1442 A folder cannot be created at this level.
- 1443 The value for 'parent folder' is not valid. Please provide a valid value.
- 1444 Raven unit price differs from the current tank unit price. The tank price will be used to price this delivery.
- 1445 The customer's source of charge is not setup for this type of transaction.
- 1446 PO Line Warehouse must be same as Ship To Warehouse for EDI Pos.
- 1447 %% must be between %% and %%.
- 1448 This Clerk Code is not eligible to release the Order Entry Special Order associated with this PO.
- 1449 Cannot delete PO lines with units received.
- 1450 This PO is not closed and cannot be deleted.
- 1451 This PO is closed and cannot be modified.
- 1452 The PO cannot be closed until all PO lines are received or closed.
- 1453 Unit Cost cannot be zero for inventory products.
- 1454 Invoice Header for Special Order %% not found.
- 1455 Warning: Order line %% for special order number %% does not exist
- 1456 Some rows in this range have units received and will not be deleted. Continue?
- 1457 Invalid network/site number.
- 1458 Invalid transaction type. Valid types are: R=Remote, D=Domestic, F=Foreign.
- 1459 Invalid card number.
- 1460 Please select a valid Site / Product row..
- 1461 The data has been finalized and can no longer be maintained. Continue in view-only mode?
- 1462 This Account/Tank combination has an Unresolved Exception for a delivery made on %% .
- 1463 The customer has an active EFT setup and/or a pending EFT.
- 1464 The customer has an active Credit Card setup and/or a pending Credit Card charge.
- 1465 The combination of Contract type, Contract base price code and Contract sub-level you have selected is not active. Please select an active combination of Contract type, Contract base price code and Contract sub-level.
- 1467 Service call scheduled for today. Use Pegasus or Service call screen to reschedule the call.
- 1468 You have selected to sort fields from multiple groups. This may lead to confusing results.
- 1469 This tank has an accepted but unprinted ticket. Use Select Tickets/Previously Accepted to find and remove this potential delivery.

- 1470 Record cannot be saved. There is a discrepancy (invalid field(s)) between the provided template and what is defined in the system.
- 1471 The record could not be saved because the supplied field is not part of the standard SmartConnect template.
- 1472 Contract linked to selected quote has expired. Use contract price anyway?
- 1473 Driver's full route needs to be loaded prior to resequencing.
- 1474 Action cannot be performed due to unsaved data.
- 1475 Invalid route. This route will not be shown on the map. Driver: %, Shift Date: %.
- 1476 Map session generation timed out.
- 1477 Do you want to move 'Mark as Read' items to history?
- 1478 This tank has unprinted selected order(s). The entrie(s) will be deleted from the selection table when the tank is put on delivery stop.
- 1479 This user is not allowed to perform a delivery stop action.
- 1480 Invalid company/database name provided.
- 1481 Transferring product from a warehouse/position to the same warehouse/position is not permitted.
- 1482 The selected technician is not active.
- 1483 The Credit Card record specified is either Invalid, Inactive or Expired.
- 1484 The full path (including file name and extension) must be provided.
- 1485 The file extension must be provided.
- 1486 The detail field is read only and cannot be modified.
- 1487 Invalid date or time provided.
- 1488 The maximum allowable length has been exceeded.
- 1489 The value must be either Y or N.
- 1490 Follow up flag is required.
- 1491 Follow up resource is required.
- 1492 Alerts cannot be set for this log type.
- 1493 Alert flag is required for this log type.
- 1494 Location Number is required for this log type.
- 1495 The file type is not permitted for this log type and/or field.
- 1496 File attachments are not permitted for this log type.
- 1497 Invalid location number provided.
- 1498 The modification of the log note and/or resolved note is not permitted for this log type.
- 1499 The modification of a system generated log is not permitted.
- 1501 You have selected a serial number already configured in the Equipment system. Do you wish to link to that serial number?
- 1502 You have selected a serial number that is configured for another customer/service location in the EQ system. Do you wish to continue?
- 1503 Cannot save inspection without setting up at least one appliance/container/regulator.
- 1504 Container is linked to an equipment on a different account/service location.

- 1505 One or more fields has an inactive value. Do you wish to continue?
- 1506 Selected asset does not exist.
- 1507 Selected asset has inactive status.
- 1508 No Financial Institutions are set up for EFT
- 1509 The address being returned from Bing maps does not match the address entered. Would you still like to use it?
- 1510 Dollars GL number required.
- 1511 Units GL number required.
- 1512 Contract information cannot be saved. One or more posting codes are linked to another subordinate contract.
- 1513 Unable to proceed as new account balance will be out of range. Valid account balance range: -21000000 to 21000000.
- 1514 There are transactions that are unprocessed that use this EFT record.
- 1515 There are transactions related to this Customer EFT record that are in a pending EFT batch. These will be processed when the batch is updated.
- 1516 This order will cause the warehouse inventory to go negative. Do you wish to proceed?
- 1517 Document ID %, is not available for viewing at this time.
- 1518 Special access privilege is required to delete or update this record.
- 1519 Card type entered did not match the card type associated with the card number. Card type has been corrected.
- 1520 The card type associated with the card number entered is not set up for credit card processing.
- 1521 Only posting codes linked to the master contract are valid.
- 1522 The cost center entered does not match the truck cost center.
- 1523 The component units in this formula of liquid do not equal the number of finished product units.
- 1524 Negative receipts have been detected in the accounting period about to be closed. These receipts will not be adjusted when inventory is replenished. Do you wish to continue?
- 1525 Account period cannot be closed as there are negative receipts detected.
- 1526 This transfer will cause the warehouse inventory to go negative. Do you wish to proceed?
- 1527 Newly calculated degree day next has exceeded the maximum limit - setting to max value 32767.
- 1528 Newly calculated degree day next is less than the minimum limit - setting to min value 1.
- 1529 This tank has been canceled. Do you want to continue?
- 1530 Click the OK button to clear all physical inventory counts.
WARNING: Any unsaved changes may be lost.
- 1531 The unit cost for overage(s) is equal to zero. Do you still want to perform the update?

- 1532 The system was unable to update inventory for the entire worksheet due to one or more errors. The failed rows have been moved to a held worksheet. You may open the worksheet to get the details and try again.
- 1533 An entry already exists with this Product/Warehouse/Position/Date combination in worksheet %%.
- 1534 An inventory adjustment has already been made for a future date.
- 1535 The action date and/or termination date cannot be greater than the current master file date.
- 1536 The acquisition reason code provided is invalid.
- 1537 The reminder date cannot exceed the maximum allowed date as defined by the E3 system option 'Credit - Reminder Max. Days'
- 1538 The release until date cannot exceed the maximum allowed date as defined by the E3 system option 'Credit - Release Until Max. Days'
- 1539 The credit action code, termination code and/or posting code is invalid.
- 1540 A credit action cannot be entered on an account that is terminated. Please reinstate the account first and try again.
- 1541 Warning: Technician has outstanding work orders assigned that may require follow-up. Do you want to continue?
- 1542 Include rows for which Physical Inventory Counts have already been entered?
- 1543 Energy was unable to process your request because the Cancel Reason code is invalid. Please contact ADD Systems Customer Support.
- 1546 Invalid Geotag Code entered. Change the code and try again.
- 1548 Masterfile date cannot be more than 125 days prior to the current posting date.
- 1549 Validation of the Geotag was successful.
- 1550 The posting date is future-dated. Are you sure you want to continue?
- 1551 Are you sure you want to proceed and lose changes for the current table?
- 1552 Warning: The Font Style and Size settings are unavailable because the database option for Laser Printing is "on."
- 1553 There is no SMS provider defined in the system, SMS cannot be enabled.
- 1554 Follow up required, due date and/or resolved date can only be set/modified when the Customer Log template's 'utilize follow-up flag' is checked.
- 1555 Due date cannot be less than the current date.
- 1556 Resolved date cannot be greater than the current date.
- 1557 The selected delivery reason is not valid for cylinder exchange.
- 1558 The selected delivery reason is not valid for metered products.
- 1559 The selected delivery reason is not valid for bulk products.
- 1560 Customer contact not found for the generated document. Please create or update customer/location contacts and try again.

- 1561 Imported record will not be saved as the source record cannot be found.
- 1562 No updates allowed. The record has already been processed.
- 1563 Invalid plan type.
- 1564 Tank# cannot be 0 when setting up a variable plan.
- 1565 Phone # is required. Please provide a phone #.
- 1566 The Gross/Net / Update Fees / Calculate fees combination is not valid for the selected transaction type. Please enter a valid combination.
- 1567 There are tickets with duplicate scheduled stop order numbers. Do you want to continue?
- 1568 There is a ticket with a scheduled stop order number that is prior to or the same as an actual stop order number. Do you want to continue?
- 1569 At least one Day of Week Schedule must be entered for DOW Tanks.
- 1570 The provided template record type and/or maintenance mode is invalid for this action.
- 1571 Invalid account and/or transaction information provided.
- 1572 Municipality/County Code State does not match Address State
- 1573 Newly calculated degree day next has exceeded the maximum limit - setting to max value 99999.
- 1574 Warning! Starting meter reading does not match current tank meter reading.
- 1575 A contract can cover specific posting code(s) or a WDMS product, but not both.
- 1576 At least one WDMS location must be provided.
- 1577 When selecting a WDMS product (Mfg Code, Group Code, Stock ID), selection can be either All/All/All, Mfg Code/All/All, Mfg Code/Group Code/All or Mfg Code/Group Code/Stock ID.
- 1578 Invalid Mfg Code, Group Code and/or Stock ID.
- 1579 A supplier must be provided.
- 1580 Invalid Supplier location.
- 1581 Invalid Supplier.
- 1582 Either 'Split price' or 'Use Contract Price for overdraw' can be selected, but not both.
- 1583 When 'Unlimited overdraw units' is not selected, a value is required in 'Maximum overdraw units'.
- 1584 The sum of the allocation units must total the original units.
- 1585 Master Account# and Master contract must either both be zero or each have a value.
- 1586 A contract cannot be its own Master contract.
- 1587 Invalid Master contract.
- 1588 Invalid Master account.
- 1589 A contract can only be a Master contract if coverage is by posting code(s).
- 1590 Remaining Units' must be less than or equal to 'Original Units'.

- 1591 Either 'Include in Quoting' or 'Quote Cost Plus' can be selected, but not both.
- 1592 Contract end date cannot be less than the start date.
- 1593 The contract date has exceeded the minimum/maximum allowable date as defined in the system.
- 1594 The start date has exceeded the minimum/maximum allowable date as defined in the system.
- 1595 The start date cannot be modified.
- 1596 The ending date exceeds the maximum allowable days from the starting date as defined in the system.
- 1597 No ePrice Quote Profile exists in ePrice Quote Parameter Maintenance for the customer and contract.
- 1598 Invalid Cost Type.
- 1599 Invalid Department Code.
- 1600 Invalid Sales Type.
- 1601 Invalid Salesperson Code.
- 1602 Non-zero gross units are not allowed if the product (or supplier/warehouse) are not set for gross / net.
- 1603 There is a mismatch between the provided account and contract.
- 1604 Supplier Code and Supplier Location must either both be blank or each have a value.
- 1605 Error: Unable to update Purchase Position.
- 1606 This Location Transfer flag (LTR) cannot be set on this location because it has been cross referenced to a WDMS Warehouse Location.
- 1607 This location may not be linked to a WDMS Warehouse Location because it has been configured as a location transfer (LTR) location.
- 1609 This action will replace manual overrides for Tariff In and Tariff Out with the original values. Continue?
- 10018 No account/tank data for this account.

Appendix D

Getting Started

Please contact the E3 administrator to obtain the url of the SmartConnect Web Service and an Energy Security identity. Eg “WebUser”. This will be used in all SmartConnect API calls to pass the “SysUser” id.

Web Service URL: _____

Security id: _____

ADD Systems provides support and resources to assist you. Please contact our support department if you need help with the SmasrtConnect API.

Code Snippet Examples

(Assumption is that all prior steps were completed)

References to the service(s) were added via the IDE as follows:

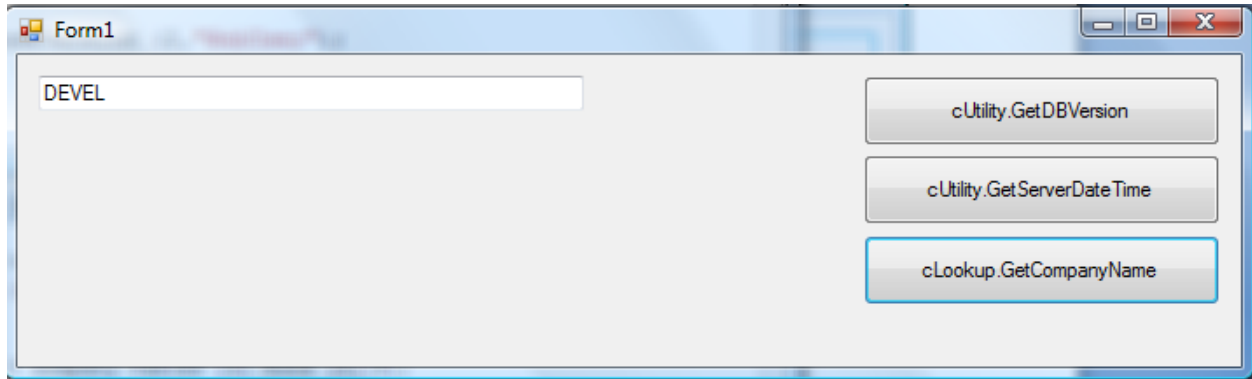
[Refer to the section “Adding a Web Reference (via IDE) for steps]

- Visual Studio 2005/References/Add Web Reference
- Visual Studio 2008/References/Add Service Reference
- Visual Web Developer 2010 Express/References/Add Web Reference

Using Visual Studio 2005

- Open new project
- Add form, textbox, buttons
- Add necessary Web Reference(s); Right-click on the project in the Solution Explorer and select “Add Web Reference...”.

Example (Form):



“clookup.GetCompanyName” Code behind:

```
private void button3_Click(object sender, EventArgs e)
{
    WSSmartConnectLookup.cLookup lookup = new WSSmartConnectLookup.cLookup ();
    int errorCode = 0;
    short companyID = 2;
    WSSmartConnectLookup.DSCompanyName company =
lookup.GetCompanyName(companyID, "WebUser", ref errorCode);
    if (errorCode == 0)
    {
        txtDBVersion.Text = (string)company.Tables[0].Rows[0][1];
    }
    else
    {
        txtDBVersion.Text = "ERROR RETRIEVING COMPANY NAME.";
    }

    lookup.Dispose();
    lookup = null;
}
```

Note: **WSSmartConnectLookup** is the Reference name assigned when adding the Web Service reference.

Using Visual Studio 2008

- Open new project
- Add form, textbox, buttons
- Add necessary Service Reference(s); Right-click on the project in the Solution Explorer and select “Add Service Reference...”.

“**cllookup.GetCompanyName**” *Code behind:*

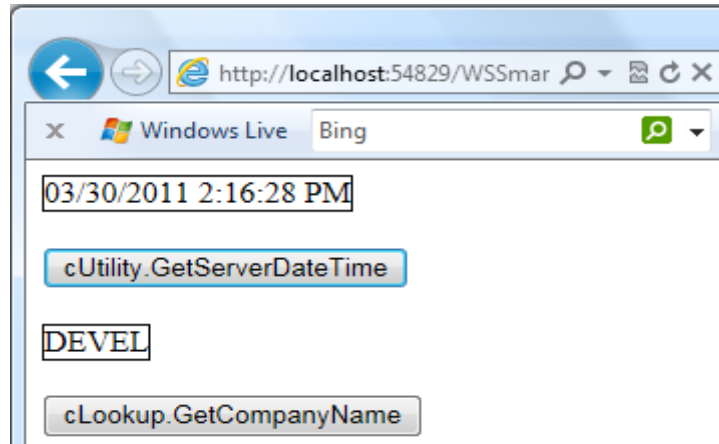
```
private void button2_Click(object sender, EventArgs e)
{
    SmartConnectLookup.cLookupSoapClient dd = new
SmartConnectLookup.cLookupSoapClient();
    int errorCode = 0;
    short companyID = 2;
    SmartConnectLookup.DSCompanyName companyNameDS =
dd.GetCompanyName(companyID, "WebUser", ref errorCode);
    if (errorCode == 0)
    {
        txtResult.Text = (string)companyNameDS.Tables[0].Rows[0][1];
    }
    else
    {
        txtResult.Text = "UNABLE TO RETRIEVE COMPANY NAME. ";
    }

    dd = null;
}
```

Note: **SmartConnectLookup** is the Reference name assigned when adding the Service reference.

Using Visual Web Developer 2010 Express

- Open New WebSite
- Add Web Form
- Add ASP Texbox, ASP Button.
- Add necessary Web Reference(s); Right-click on the project in the Solution Explorer and select “Add Web Reference...”.



“clookup.GetCompanyName” Code behind:

```
protected void GetCompanyName(object sender, EventArgs e)
{
    WSSmartConnectLookup.cLookUp dd = new WSSmartConnectLookup.cLookUp();
    int errorCode = 0;
    short companyID = 2;
    WSSmartConnectLookup.DSCompanyName companyInfo = dd.GetCompanyName(companyID,
    "WebUser", ref errorCode);
    //
    if (errorCode == 0)
    {
        lblCompany.Text = (string)companyInfo.Tables[0].Rows[0][1];
    }
    else
    {
        lblCompany.Text = "UNABLE TO RETRIEVE COMPANY NAME.";
    }

    dd.Dispose();
    dd = null;
}
```

Note: **WSSmartConnectLookup** is the Reference name assigned when adding the Web Service reference.

How to test your request

There is a tool available on the internet called SoapUI . You can download it from www.eviware.com

This tool will connect to the Smart Connect Web Service and formulate the xml requests. Once you fill in the parameters and send the request, SoapUI will display the xml response from the web service.

This will help you to debug problems with SmartConnect.

Appendix E

cOrder schema

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/" xmlns:tns="http://www.addsys.com/WDMS/POS"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://www.addsys.com/WDMS/POS" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified" targetNamespace="http://www.addsys.com/WDMS/POS">
      <s:element name="SaveOrders">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="orders_root" type="tns:Orders" />
            <s:element minOccurs="1" maxOccurs="1" name="p_iCompanyID" type="s:short" />
            <s:element minOccurs="0" maxOccurs="1" name="p_sSysUser" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="Orders">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="orders" type="tns:ArrayOfOrder" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfOrder">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="order" nillable="true"
type="tns:Order" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="Order">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="order_sequence_number" type="s:int" />
          <s:element minOccurs="0" maxOccurs="1" name="order_type">
            <s:simpleType>
              <s:restriction base="s:string" >
                <s:maxLength value="3"/>
              </s:restriction>
            </s:simpleType>
          </s:element>
          <s:element name="clerk" minOccurs="0" maxOccurs="1">
            <s:simpleType>
```

```

        <s:restriction base="s:string" >
            <s:maxLength value = "5"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="1" maxOccurs="1" name="cost_center">
    <s:simpleType>
        <s:restriction base="s:short" >
            <s:minInclusive value="0"/>
            <s:maxInclusive value="32766"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="1" maxOccurs="1" name="account_num">
    <s:simpleType>
        <s:restriction base="s:int" >
            <s:minInclusive value="1"/>
            <s:maxInclusive value="999999"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="1" maxOccurs="1" name="location_num">
    <s:simpleType>
        <s:restriction base="s:short" >
            <s:minInclusive value="1"/>
            <s:maxInclusive value="32766"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="customer_po_number">
    <s:simpleType>
        <s:restriction base="s:string" >
            <s:maxLength value = "25"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="text">
    <s:simpleType>
        <s:restriction base="s:string" >
            <s:maxLength value = "50"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" default="0" name="base_price_code"
type="s:unsignedByte" />
<s:element minOccurs="0" maxOccurs="1" name="ship_via">
    <s:simpleType>
        <s:restriction base="s:string" >
            <s:maxLength value = "10"/>
        </s:restriction>
    </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" default="0" name="freight_charge"
type="s:decimal" />
<s:element minOccurs="0" maxOccurs="1" name="sales_person">
    <s:simpleType>
        <s:restriction base="s:string" >
            <s:maxLength value = "3"/>
        </s:restriction>
    </s:simpleType>

```

```

        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="reference_num">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "15"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="priority_order">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value="1"/>
            <s:pattern value = "[Y,N]*"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="1" maxOccurs="1" name="requested_delivery_dt" type="s:dateTime"
/>

      <s:element minOccurs="0" maxOccurs="1" default="0" name="hauler_office_id">
        <s:simpleType>
          <s:restriction base="s:int" >
            <s:minInclusive value="0"/>
            <s:maxInclusive value="999999"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="1" maxOccurs="1" name="modified_order_num">
        <s:simpleType>
          <s:restriction base="s:int" >
            <s:minInclusive value="0"/>
            <s:maxInclusive value="999999"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="1" maxOccurs="1" name="flag_for_dispatch" type="s:boolean" />
      <s:element minOccurs="0" maxOccurs="1" name="address_override"
type="tns:AddressOverride" />
      <s:element minOccurs="0" maxOccurs="1" name="order_lines" type="tns:ArrayOfOrderLine"
/>

    </s:sequence>
  </s:complexType>
  <s:complexType name="AddressOverride">
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="telephone">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "10"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="name">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "40"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="street_1">

```

```

        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "40"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="street_2">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "40"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="city">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "40"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="state">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "4"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="1" maxOccurs="1" name="country">
        <s:simpleType>
          <s:restriction base="s:unsignedByte" >
            <s:minInclusive value="0"/>
            <s:maxInclusive value="1"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
      <s:element minOccurs="0" maxOccurs="1" name="postal_code">
        <s:simpleType>
          <s:restriction base="s:string" >
            <s:maxLength value = "10"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
    </s:sequence>
  </s:complexType>
  <s:complexType name="ArrayOfOrderLine">
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="unbounded" name="order_line" nillable="true"
type="tns:OrderLine" />
    </s:sequence>
  </s:complexType>
  <s:complexType name="OrderLine">
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" default="0" name="order_line">
        <s:simpleType>
          <s:restriction base="s:short" >
            <s:minInclusive value="0"/>
            <s:maxInclusive value="32766"/>
          </s:restriction>
        </s:simpleType>
      </s:element>
    </s:sequence>
  </s:complexType>

```

```
</s:element>
<s:element minOccurs="1" maxOccurs="1" name="action" type="tns:Action" />
<s:element minOccurs="0" maxOccurs="1" name="mfg_code">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "4"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="group_code">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "6"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="stock_id">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "12"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="unit_name">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "4"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" default="0" name="order_qty">
  <s:simpleType>
    <s:restriction base="s:decimal" >
      <s:minInclusive value="0"/>
      <s:maxInclusive value="99999999.999"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="1" maxOccurs="1" name="unit_price">
  <s:simpleType>
    <s:restriction base="s:decimal" >
      <s:minInclusive value="0"/>
      <s:maxInclusive value="99999.9999"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="whse_code">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "3"/>
    </s:restriction>
  </s:simpleType>
</s:element>
<s:element minOccurs="0" maxOccurs="1" name="bin_code">
  <s:simpleType>
    <s:restriction base="s:string" >
      <s:maxLength value = "4"/>
    </s:restriction>
  </s:simpleType>
</s:element>
```



```

    </s:element>
    <s:element minOccurs="0" maxOccurs="1" name="location_code">
      <s:simpleType>
        <s:restriction base="s:string" >
          <s:maxLength value = "10"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
    <s:element minOccurs="0" maxOccurs="1" name="ms_code">
      <s:simpleType>
        <s:restriction base="s:string" >
          <s:maxLength value = "4"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
    <s:element minOccurs="0" maxOccurs="1" default="" name="loading_auth_num">
      <s:simpleType>
        <s:restriction base="s:string" >
          <s:maxLength value = "20"/>
        </s:restriction>
      </s:simpleType>
    </s:element>
  </s:sequence>
</s:complexType>
<s:simpleType name="Action">
  <s:restriction base="s:string">
    <s:enumeration value="insert" />
    <s:enumeration value="update" />
    <s:enumeration value="delete" />
  </s:restriction>
</s:simpleType>
<s:element name="SaveOrdersResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="results_root"
type="tns:SaveOrdersResults" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="SaveOrdersResults">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="results"
type="tns:ArrayOfSaveOrderResult" />
    <s:element minOccurs="0" maxOccurs="1" name="exception" type="tns:SaveOrderException"
/>
  </s:sequence>
</s:complexType>
<s:complexType name="ArrayOfSaveOrderResult">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="result" nillable="true"
type="tns:SaveOrderResult" />
  </s:sequence>
</s:complexType>
<s:complexType name="SaveOrderResult">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="order_sequence_number" type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="order_number" type="s:int" />
    <s:element minOccurs="0" maxOccurs="1" name="exception" type="tns:SaveOrderException"
/>
  </s:sequence>
</s:complexType>

```

```
</s:sequence>
</s:complexType>
<s:complexType name="SaveOrderException">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="exception_object_type"
type="tns:ExceptionObjectType" />
    <s:element minOccurs="1" maxOccurs="1" name="error_object_sequence_number"
type="s:int" />
    <s:element minOccurs="1" maxOccurs="1" name="error_number" type="s:int" />
    <s:element minOccurs="0" maxOccurs="1" name="error_message" type="s:string" />
  </s:sequence>
</s:complexType>
<s:simpleType name="ExceptionObjectType">
  <s:restriction base="s:string">
    <s:enumeration value="none" />
    <s:enumeration value="order" />
    <s:enumeration value="order_line" />
  </s:restriction>
</s:simpleType>
</s:schema>
</wsdl:types>
</wsdl:definitions>
```

Appendix F

EnergyAPI.CustomerLogSelectionCriteria		
Field name	Field type	Description
TemplatesToSearch	Integer ()	Templates to include in search. Default: empty array (search all templates) *Refer to LoadCustomerLogTemplates()* <i>DSNumCodeDescStatus.code</i>
** Atleast one of the following 4 properties must be set to true **		
IncludeGeneralCustomer	Boolean	Limits the search to log entries linked to account master records. Default: "True"
IncludeDeliveryLocations	Boolean	Limits the search to log entries linked with tank record(s). Default: "False"
IncludeServiceLocations	Boolean	Limits the search to log entries linked with service record(s). Default: "False"
IncludeWDMSLocations	Boolean	Limits the search to log entries linked with WDMS record(s). Default: "False"
** TO SEARCH ALL LOG ENTRIES LEAVE THE FOLLOWING 4 FIELDS WITH THEIR DEFAULT VALUE **		
IF LogsCreatedOrModifiedFromDate / LogsCreatedOrModifiedToDate are entered <i>LogsCreatedOrModifiedPrevMonths and LogsCreatedOrModifiedPrevDays are ignored.</i> If LogsCreatedOrModifiedPrevMonths is entered, LogsCreatedOrModifiedFromDate / <i>LogsCreatedOrModifiedToDate and LogsCreatedOrModifiedPrevDays are ignored.</i> If LogsCreatedOrModifiedPrevDays is entered, LogsCreatedOrModifiedFromDate / <i>LogsCreatedOrModifiedToDate and LogsCreatedOrModifiedPrevMonths are ignored.</i>		
The last field to be updated rules		
LogsCreatedOrModifiedFromDate	Date	From Date: Finds log entries created or modified within a specific date range. Default: 01/01/1900
LogsCreatedOrModifiedToDate	Date	To Date: Finds log entries created or modified within a specific date range. Default: 01/01/1900
LogsCreatedOrModifiedPrevMonths	Short (16bit integer)	Allows you to select the number of months prior to the current date within which the component will search for log entries. The maximum number of months is 24. Any value > 24 will be set to 24. Default: 0
LogsCreatedOrModifiedPrevDays	Short (16bit integer)	Allows you to select the number of days prior to the current date within which the component will search for log entries. The maximum number of days is 60. Any value > 60 will be set to 60. Default: 0
Contains selection criteria that limit or combine types of customer log entries for review.		

Any option or combination of options may be selected. All options are selected by default.		
FollowupStatusNotRequired	Boolean	This will limit the search to log entries for which follow-up is not required. Default: "True"
FollowupStatusRequired	Boolean	This will limit the search to log entries for which follow-up is required. Default: "True"
FollowupStatusRequiredDate	Date	This field will only be valid if you have checked "Required". The component will search for log entries with follow-up required dates on or before the date entered here. Default: Today + 5 years <i>*NOTE*</i> Any date where > 5 years from "today" will be reset by the API to "today"+ 5 years.
FollowupStatusResolved	Boolean	Select this field only if you want to search for log entries marked as resolved. Default: "True"
FollowupResources	Integer()	An individual or group that is selected to perform follow-up activities. Default: Empty array *Refer to LoadCustomerLogFollowupResources()* <i>DSLLoadResources.resource_id</i>
** ONLY ONE OF THE FOLLOWING 3 FIELDS CAN BE TRUE AT A GIVEN TIME; LAST ONE TO GET UPDATED RULES **		
IncludeAllAlerts	Boolean	Include both alerted and not alerted in search. Default: "True"
IncludeAlertedOnly	Boolean	This option lets you search for log entries with alerts. Default: "False"
IncludeNotAlertedOnly	Boolean	This option lets you search for log entries without alerts. Default: "False"
FullAccount	String	Full/ validated account # to search logs for specific customer. Default: blank (to search all customers)
DivisionsToInclude	Integer() Array	Divisions to include in search. Default: empty array (include all divisions) *Refer to LoadCustomerLogDivisions()* <i>DSDivisionName.division</i>
CollectorsToInclude	Integer()	Collectors to include in search. Default: empty array (include all collectors) *Refer to LoadCustomerLogCollectors()* <i>DSNumCodeDescStatus.code</i>

Customerlog.CustomerLogHeader		
Field name	Field type	Description
AccountNum	Integer	Account # without the checkdigit that is used to interact with the database.*Should not be displayed*
FullAccount	String	Account number with the checkdigit. *To be displayed*
LogHeaderID	Integer	Along with the AccountNum , uniquely identifies a log header record.
LogTemplateHeaderID	Integer	Required to identify the correct Log Template. A value < 0 signifies that the Log Template is system defined and updates to these log types are not permitted by the end user but only the system.
LocationType	Short (16bit integer)	0 – Customer Level, 1 – Tank, 2 – Service, 3 - WDMS
LocationNum	Integer	Represents a location # for any LocationType that is <> 0
FollowupRequired	String	Y: follow up is required N: follow up is not required
DueDate	Date	Due date for any log where follow up is required. *Note* this data is saved as a date only and 12:00AM (no time portion). To remove a due date, set the value to 01/01/1900
FollowupResource	Integer	The resource that will perform the follow up for any log where follow up is required. The corresponding description must be found by iterating through the list provided by: LoadCustomerLogFollowupResources()
ResolvedDate	Date	Date when log is resolved. *Note* This date can/will be saved with a time portion as well. To remove a resolved date, set the value to 01/01/1900
AlertFlag	Boolean	True: there is an alert for the log False: there is no alert for the log
AllowModificationToExistingNotes	Short (16bit integer)	0: modification to existing note is not permitted 1: modification to existing note is permitted 2: Only appending to existing note is permitted (Applies to both Note and ResolvedNote)
Note	String*1020	Comments that are attached to the Log record. The value cannot exceed 1020 characters or <u>truncation will occur</u> .
ResolvedNote	String*510	
		Notes that can be added when log is resolved. The value cannot exceed 510 characters or <u>truncation will occur</u> .

LastMaintenanceUserID	String	The last user to modify the log header record
LastMaintenanceDate	DateTime	The last date/time the log header record was modified
AllowAttachments	Boolean	If "False" then file attachments will not be permitted. As well there is no need to call LoadCustomerLogAttachments() as there will be no attachments to load.
AttachmentFileType	String	If not blank it represents the file extension that is permitted (eg xls, pdf, doc, etc). This governs both Note and Resolved Note attachment type.
NoteAttachmentSaveAsShortCut	Boolean	When set to "True" NoteAttachmentFullPath (if applicable) will serve as the path to the attachment location. Default: "False"
NoteAttachmentBinaryFormat	String	When saving an attachment to the log header record as binary this field must contain the file extension (eg: xls, doc, pdf, jpg, etc.). * ONLY update this field when adding and/or modifying an attachment * *IMPORTANT* When AttachmentSaveAsShortCut is set to "True" this field will be ignored.
NoteAttachmentFullPath	String	When saving a Note/Comment attachment to the log header record, this field must always contain the full path* (including the filename with the extension). *full path: not mapped but full UNC format* * ONLY update this field when adding and/or modifying an attachment *
NoteAttachmentBinary	Byte()	Note/Comment file attachment as binary. Use when access to a file location is limited. * ONLY update this field when adding and/or modifying an attachment * *IMPORTANT* When AttachmentSaveAsShortCut is "True" this field will be ignored.
NoteAttachmentRemove	Boolean	Set to "True" to remove the note attachment from the log header record. *IMPORTANT* When "True" all other NoteAttachment* fields will be ignored. Setting the value to "True" will not delete the actual file. Default: "False"
ResolvedAttachmentSaveAsShortCut	Boolean	When set to "True" ResolvedNoteAttachmentFullPath (if applicable) will serve as the path to the attachment location. Default: "False"
ResolvedAttachmentBinaryFormat	String	*refer to " NoteAttachmentBinaryFormat "*
ResolvedAttachmentFullPath	String	*refer to " NoteAttachmentFullPath "*
ResolvedAttachmentBinary	Byte()	*refer to " NoteAttachmentBinary "*
ResolvedAttachmentRemove	Boolean	Set to "True" to remove the resolved note

		attachment from the log header record. *IMPORTANT* When "True" all other <i>ResolvedAttachment*</i> fields will be ignored. Setting the value to "True" will not delete the actual file. Default: "False"
CreatedBy	String	The user id that created the log header record
CreatedDate	DateTime	The date/time when log header record was created
HeaderRecordUpdated	Boolean	If any change is made to any field in the header record , set this field to "True". In the case where only detail(s) records are being updated there is no need to set this property to "True"
SystemDefined	Boolean	"True": log is system created and cannot be modified "False": log is user created and can be modified *Goes hand in hand with <i>LogTemplateHeaderID*</i>
ErrorCode	Integer	0: no error occurred when updating the log header record. < > 0: an error occurred during the updating of the log header record. *Call <i>Utility.GetErrorDescription()</i> to retrieve the description.*

DCEnergy.Customerlog.CustomerLogDetail

Field name	Field type	Description
SequenceNo	Short (16bit integer)	Field order for log details
RepetitionNo	Short (16bit integer)	Non zero when there are repetitive details for each field. *should never be modified*
Label	String	Description of field
DataTypeID	Short (16bit integer)	Data type: 3: integer 5: decimal 6: currency 7: date (MM/dd/yy) 8: text 9: list 11: Yes/No 17: file attachment 70: time (HH:mm:ss)
Length	Short (16bit integer)	Maximum length when DataTypeID is 8
Minimum	String	When not blank: minimum value when DataTypeID in 3,5,6
Maximum	String	When not blank: maximum value when DataTypeID in 3,5,6
Required	Boolean	NewValue cannot be blank.
ModifyBy	Short (16bit integer)	*internal use* should not be modified. When value =2 field is read-only. The system will not allow modification to the field even if attempt is made.
FieldListDisplay	String	When DataTypeID is 9: Comma delimited list that must presented as a single selection drop down list.
FieldList	String	When DataTypeID is 9: Comma delimited list that corresponds to the value that will be saved for FieldListDisplay selection.
** THE FOLLOWING 3 FIELDS ARE VALID ONLY WHEN DataTypeID is 17 (File attachment)**		
FileAttachmentBinaryFormat	String	When saving a file attachment this property must contain the file extension (eg: xls, doc, pdf...)
FileAttachmentBinary	Byte()	File attachment in binary Default: nothing/null * ONLY update this field when adding and/or modifying an attachment *
FileAttachmentRemove	Boolean	Set to "True" to remove the file attachment from detail field. When "True" all other <i>FileAttachment*</i> fields will be ignored. Setting the value to "True" will not delete the actual file. Default: "False"

** FOLLOWING 2 FIELDS ARE VALID WHEN DataTypeID <> 17 **		
OldValue	String	Old value or Not applicable * *IMPORTANT* should never be updated.
NewValue	String	Will be used to update the database
ErrorCode	Integer	0: validation passed and update successful. <>0: validation failed and/or update not successful.

OldValue*: The **OldValue** is only applicable for templates that are SystemDefined (refer to CustomerLogHeader).