CONTENTS

About This Guide
Intended Audience .............................................................................................................. xxiii
Revision History ................................................................................................................ xxiii
Document Structure .......................................................................................................... xxvii
Documentation Set ........................................................................................................... xxvii
Typographical Conventions .............................................................................................. xxix
Contact Information ......................................................................................................... xxx

Chapter 1 Introduction
The cnpAPI Data Format ...................................................................................................... 2
Communications Protocols ................................................................................................. 2
General XML Coding Requirements .................................................................................. 3
Other XML Resources ...................................................................................................... 4
Batch Transaction Processing ............................................................................................. 5
Recommended Session File Size ...................................................................................... 5
Multiple Daily Delivery ..................................................................................................... 5
Payment Integration Platform (cnpAPI SDKs) ..................................................................... 7
Duplicate Transaction Detection ......................................................................................... 8
Batch Duplicate Checking .................................................................................................. 8
Batch Dupe Checking for Dynamic Payout Funding Instructions .................................... 9
Online Duplicate Checking ............................................................................................... 9
Coding for Report Groups ................................................................................................. 10
Additional/Alternate Methods of Tagging Transactions .................................................. 11
Recovery ............................................................................................................................ 12
Authorization/Sale Recycling ............................................................................................ 12
Recycling Engine .............................................................................................................. 12
Account Updater Service ................................................................................................... 14
Match Back ...................................................................................................................... 15
Merchant Requirements ................................................................................................... 16
Account Updater Features ................................................................................................. 17
Recurring Engine ............................................................................................................ 18
Payment Plans .................................................................................................................. 18
Subscriptions .................................................................................................................... 19
Add Ons and Discounts .................................................................................................... 21
Recurring Reports ........................................................................................................... 21
Transaction Types and Uses ............................................................................................. 22
Issuer Insights .................................................................................................................... 24
Issuer Insights Secure Scheduled Report ......................................................................... 24
Contents

eCommerce Solution for Apple Pay™ ................................................................. 50
  Overview of Apple Pay Operation ............................................................... 50
  Worldpay Decryption of Apple Pay PKPaymentToken .................................. 51
    Using the Browser JavaScript API for Apple Pay on the Web ................. 51
    Using the Worldpay Mobile API for Apple Pay ........................................ 53
  Submitting the Apple Pay PKPaymentToken in a cnpAPI Message ............. 54
  Merchant Decryption of Apple Pay PKPaymentToken .................................. 55
Recurring Payments with Apple Pay ............................................................. 56

eCommerce Solution for Google Pay™ .......................................................... 57
  Branding Requirements .................................................................................. 57
  Google Pay using eProtect .............................................................................. 57
  Merchant Decryption Method ........................................................................ 60
  Recurring Payments with Google Pay ........................................................ 62
Amazon Pay ..................................................................................................... 64

Healthcare Card Feature .................................................................................. 66

Supported Transaction Types .......................................................................... 68
  Authorization Transaction .............................................................................. 68
    AVS Only Transaction .................................................................................. 69
  Authorization Reversal Transactions ........................................................... 69
    Notes on the Use of Authorization Reversal Transactions ....................... 70
    Using Authorization Reversal to Halt Recycling Engine ......................... 70
  Activate Transaction ..................................................................................... 71
  Activate Reversal Transaction (Online Only) ................................................. 71
  Balance Inquiry Transaction .......................................................................... 71
  Cancel Subscription Transaction ................................................................. 71
  Capture Transaction ....................................................................................... 71
  Capture Given Auth Transaction ................................................................... 72
  Create Plan Transaction .............................................................................. 72
  Credit Transaction ......................................................................................... 72
  Deactivate Transaction .................................................................................. 73
  Deactivate Reversal Transaction (Online Only) ............................................ 73
  Deposit Reversal Transaction (Online Only) ................................................. 73
  eCheck Credit Transaction .......................................................................... 73
  eCheck Prenotification Credit Transaction .................................................. 73
  eCheck Prenotification Sale Transaction ...................................................... 73
  eCheck Redeposit Transaction ...................................................................... 74
  eCheck Sales Transaction ............................................................................. 74
  eCheck Verification Transaction ................................................................. 74
  eCheck Void Transaction (Online Only) ....................................................... 74
  Force Capture Transaction ........................................................................... 74
  Gift Card Auth Reversal .............................................................................. 74
  Gift Card Capture ......................................................................................... 75
  Gift Card Credit ........................................................................................... 75
## Contents

<table>
<thead>
<tr>
<th>Transaction Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load Transaction</td>
<td>75</td>
</tr>
<tr>
<td>Load Reversal Transaction (Online Only)</td>
<td>75</td>
</tr>
<tr>
<td>Refund Reversal Transaction (Online Only)</td>
<td>75</td>
</tr>
<tr>
<td>Register Token Transaction</td>
<td>75</td>
</tr>
<tr>
<td>Sale Transaction</td>
<td>75</td>
</tr>
<tr>
<td>Status Query Transaction</td>
<td>76</td>
</tr>
<tr>
<td>Unload Transaction</td>
<td>76</td>
</tr>
<tr>
<td>Unload Reversal Transaction (Online Only)</td>
<td>76</td>
</tr>
<tr>
<td>Update Card Validation Number Transaction</td>
<td>76</td>
</tr>
<tr>
<td>Update Plan Transaction</td>
<td>77</td>
</tr>
<tr>
<td>Update Subscription Transaction</td>
<td>77</td>
</tr>
<tr>
<td>Void Transaction (Online Only)</td>
<td>77</td>
</tr>
<tr>
<td>Using Void to Halt Recycling Engine</td>
<td>77</td>
</tr>
<tr>
<td>Instruction-Based Dynamic Payout Transactions</td>
<td>78</td>
</tr>
</tbody>
</table>

### Chapter 2 Testing Your cnpAPI Transactions

<table>
<thead>
<tr>
<th>Subject</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certification and Testing Environments</td>
<td>80</td>
</tr>
<tr>
<td>Sandbox Environment</td>
<td>80</td>
</tr>
<tr>
<td>Pre-Live Environment</td>
<td>80</td>
</tr>
<tr>
<td>Pre-Live Environment Limitations and Maintenance Schedules</td>
<td>80</td>
</tr>
<tr>
<td>Overview of Testing</td>
<td>82</td>
</tr>
<tr>
<td>Planning for Certification Testing</td>
<td>82</td>
</tr>
<tr>
<td>Required Certification Testing</td>
<td>83</td>
</tr>
<tr>
<td>Optional Testing</td>
<td>83</td>
</tr>
<tr>
<td>System Doctor</td>
<td>83</td>
</tr>
<tr>
<td>Transferring Files</td>
<td>86</td>
</tr>
<tr>
<td>Transferring Session Files</td>
<td>86</td>
</tr>
<tr>
<td>Submitting a Session File for Processing</td>
<td>86</td>
</tr>
<tr>
<td>Retrieving Processed Session Files</td>
<td>86</td>
</tr>
<tr>
<td>Transferring Online Files</td>
<td>87</td>
</tr>
<tr>
<td>ASP Programming Example</td>
<td>88</td>
</tr>
<tr>
<td>Java Programming Example</td>
<td>89</td>
</tr>
<tr>
<td>Notes on Timeout Settings</td>
<td>89</td>
</tr>
<tr>
<td>Notes on Persistent Connections</td>
<td>90</td>
</tr>
<tr>
<td>Helpful Web Sites</td>
<td>90</td>
</tr>
<tr>
<td>Performing the Required Certification Tests</td>
<td>91</td>
</tr>
<tr>
<td>Testing Authorization (including Indicators), AVS Only, Capture, Credit, Sale, and Void Transactions</td>
<td>91</td>
</tr>
<tr>
<td>Testing Authorization Reversal Transactions</td>
<td>104</td>
</tr>
<tr>
<td>Testing Direct Debit Transactions</td>
<td>108</td>
</tr>
<tr>
<td>Testing Token Transactions</td>
<td>115</td>
</tr>
<tr>
<td>Testing Query Transactions</td>
<td>119</td>
</tr>
<tr>
<td>Testing Stored Credentials Processing</td>
<td>120</td>
</tr>
</tbody>
</table>
# Contents

Testing Online Duplicate Transaction Processing .............................................................. 123  
Testing Refund Authorization ............................................................................................ 124  
Performing the Optional Tests .......................................................................................... 125  
Testing AVS and Card Validation ....................................................................................... 126  
Testing Address Responses ............................................................................................... 127  
Testing Advanced AVS Response Codes ......................................................................... 128  
Testing Response Reason Codes and Messages .............................................................. 129  
Testing 3DS Responses .................................................................................................... 133  
Testing the Prepaid Filtering Feature ............................................................................. 135  
Testing the International Card Filter Feature ................................................................. 136  
Testing Security Code No-Match Filtering ....................................................................... 137  
Testing Advanced Fraud Tools ......................................................................................... 139  
Testing Account Updater .................................................................................................. 141  
  Testing Account Updater Extended Response Codes ..................................................... 142  
  Testing Account Updater for Tokenized Merchants ....................................................... 142  
Testing Tax Billing ............................................................................................................ 142  
Testing Convenience Fees ............................................................................................... 143  
Testing the Recycling Engine .......................................................................................... 145  
  Testing Recycling Engine Cancellation ........................................................................ 153  
Testing Recurring Engine Transactions .......................................................................... 155  
Testing Gift Card Transactions ....................................................................................... 166  
Testing MasterPass Transactions ..................................................................................... 175  
Testing Apple Pay Transaction Processing ..................................................................... 177  
  Testing the Submission of the Decrypted PKPaymentToken in cnpAPI ......................... 176  
  Testing the Submission of PKPaymentToken in cnpAPI .............................................. 177  
Testing Google Pay Transaction Processing ................................................................. 178  
  Testing Google Pay using eProtect ............................................................................. 178  
Testing Amazon Pay ......................................................................................................... 179  
Testing checkoutId ............................................................................................................ 182  
Testing Transaction Volume Capacity ............................................................................. 183  

## Chapter 3 cnpAPI Transaction Examples

Overview of Online and Batch Processing Formats ......................................................... 186  
  Batch Process Format .................................................................................................... 186  
  Supported Communication Protocols ........................................................................... 187  
  Batch Processing Request Format .................................................................................. 187  
  Batch Processing Response Format .............................................................................. 187  
Online Processing Format ............................................................................................... 188  
  Supported Communication Protocols ........................................................................... 189  
  Online Processing Request Format ................................................................................ 189  
  Online Processing Response Format .............................................................................. 189  
Transaction Types and Examples ..................................................................................... 190  
  Authorization Transactions ............................................................................................ 192
Authorization Request Structure ................................................................................... 192
Authorization Response Structure .............................................................................. 199
Authorization Reversal Transactions ......................................................................... 204
Authorization Reversal Requests ............................................................................. 204
Authorization Reversal Responses .......................................................................... 205
Activate Transactions .................................................................................................. 206
Activate Request ........................................................................................................ 206
Activate Response ...................................................................................................... 208
Activate Reversal Transactions (Online Only) ............................................................. 209
Activate Reversal Request ........................................................................................ 209
Activate Reversal Response ..................................................................................... 210
Balance Inquiry Transactions ...................................................................................... 211
Balance Inquiry Request ........................................................................................... 211
Balance Inquiry Response ....................................................................................... 212
Cancel Subscription Transactions ............................................................................. 213
Cancel Subscription Request .................................................................................... 213
Cancel Subscription Response .................................................................................. 213
Capture Transactions .................................................................................................. 214
Capture Request ........................................................................................................ 214
Capture Response ...................................................................................................... 221
Capture Given Auth Transactions ............................................................................... 222
Capture Given Auth Request ..................................................................................... 222
Capture Given Auth Response ................................................................................... 226
Create Plan Transactions ........................................................................................... 228
Create Plan Request .................................................................................................. 228
Create Plan Response .................................................................................................. 229
Credit Transactions ..................................................................................................... 229
Credit Request for a Worldpay Processed Transaction ............................................. 230
Credit Request for a Non-Worldpay Processed Transaction ....................................... 233
Credit Response ......................................................................................................... 237
Deactivate Transactions ............................................................................................. 239
Deactivate Request .................................................................................................... 239
Deactivate Response ................................................................................................. 240
Deactivate Reversal Transactions (Online Only) ......................................................... 240
Deactivate Reversal Request .................................................................................... 240
Deactivate Reversal Response .................................................................................. 242
Deposit Reversal Transactions (Online Only) .............................................................. 242
Deposit Reversal Request ......................................................................................... 242
Deposit Reversal Response ....................................................................................... 244
eCheck Credit Transactions ........................................................................................ 244
eCheck Credit Request Against a Worldpay Transaction ........................................ 245
eCheck Credit Request for a Non-Worldpay Processed Sale ..................................... 246
eCheck Credit Response ............................................................................................ 247
Contents

eCheck Prenotification Credit Transactions (Batch Only) ................................................... 247
  eCheck Prenotification Credit Request ........................................................................ 248
  eCheck Prenotification Credit Response .................................................................... 249
eCheck Prenotification Sale Transactions (Batch Only) ..................................................... 250
  eCheck Prenotification Sale Request ........................................................................ 250
  eCheck Prenotification Sale Response ...................................................................... 251
eCheck Redeposit Transactions .......................................................................................... 252
  eCheck Redeposit Request ....................................................................................... 252
  eCheck Redeposit Response ...................................................................................... 254
eCheck Sale Transactions .................................................................................................. 255
  eCheck Sale Request ................................................................................................... 255
  eCheck Sale Response ................................................................................................ 256
eCheck Verification Transactions ..................................................................................... 258
  eCheck Verification Request ..................................................................................... 258
  eCheck Verification Response .................................................................................... 260
eCheck Void Transactions (Online Only) .......................................................................... 262
  eCheck Void Request .................................................................................................. 262
  eCheck Void Response ................................................................................................ 262
Force Capture Transactions ............................................................................................... 263
  Force Capture Request ............................................................................................... 263
  Force Capture Response ............................................................................................. 266
Fraud Check Transaction ................................................................................................. 268
  Fraud Check Request .................................................................................................. 268
  Fraud Check Response ................................................................................................. 269
Gift Card Auth Reversal Transactions ................................................................................ 270
  Gift Card Auth Reversal Request ............................................................................... 270
  Gift Card Auth Reversal Response ............................................................................. 271
Gift Card Capture Transactions .......................................................................................... 272
  Gift Card Capture Request ....................................................................................... 272
  Gift Card Capture Response ....................................................................................... 273
Gift Card Credit Transactions ............................................................................................. 273
  Gift Card Credit Request ........................................................................................... 274
  Gift Card Credit Response .......................................................................................... 275
Load Transactions .............................................................................................................. 276
  Load Request ................................................................................................................ 276
  Load Response .............................................................................................................. 277
Load Reversal Transactions (Online Only) ......................................................................... 278
  Load Reversal Request ............................................................................................... 278
  Load Reversal Response .............................................................................................. 279
Status Query Transactions (Online Only) .......................................................................... 280
  Query Transaction Request ....................................................................................... 280
  Query Transaction Response ...................................................................................... 280
Refund Reversal Transactions (Online Only) ..................................................................... 283
Refund Reversal Request ............................................................................................. 284
Refund Reversal Response .......................................................................................... 285
Register Token Transactions ........................................................................................ 286
Register Token Request ............................................................................................... 286
Register Token Response ............................................................................................. 289
RFR Transactions (Batch Only) .................................................................................... 291
RFR Request ................................................................................................................ 291
RFR Response .............................................................................................................. 292
Sale Transactions ......................................................................................................... 292
Sale Request ................................................................................................................ 292
Sale Response .............................................................................................................. 295
Unload Transactions .................................................................................................... 300
Unload Request ........................................................................................................... 300
Unload Response .......................................................................................................... 301
Unload Reversal Transactions (Online Only) ................................................................. 301
Unload Reversal Request ............................................................................................ 302
Unload Reversal Response .......................................................................................... 303
Update Plan Transactions ............................................................................................ 303
Update Plan Request .................................................................................................... 304
Update Plan Response .................................................................................................. 304
Update Subscription Transactions .................................................................................. 305
Update Subscription Request ....................................................................................... 305
Update Subscription Response .................................................................................... 306
Update Card Validation Number Transactions ............................................................. 307
Update Card Validation Number Request .................................................................... 307
Update Card Validation Number Response .................................................................. 308
Void Transactions (Online Only) .................................................................................... 308
Void Request .............................................................................................................. 309
Void Response ............................................................................................................ 310

Chapter 4 cnpAPI Elements

accNum .......................................................................................................................... 312
accountInfo .................................................................................................................... 313
accountInformation ....................................................................................................... 314
accountNumber ............................................................................................................. 315
accountNumberLength .................................................................................................... 316
accountUpdate ............................................................................................................... 317
accountUpdateFileRequestData .................................................................................... 318
accountUpdater ............................................................................................................. 319
accountUpdateResponse ............................................................................................... 323
accType .......................................................................................................................... 324
actionReason .................................................................................................................. 325
activate .......................................................................................................................... 326
activateResponse .................................................................................................................. 327
activateReversal .................................................................................................................... 328
activateReversalResponse ...................................................................................................... 329
active ...................................................................................................................................... 330
addOnCode ............................................................................................................................... 331
addressIndicator ..................................................................................................................... 332
addressLine1, addressLine2, addressLine3 ............................................................................. 333
advancedAVSResult .................................................................................................................. 334
advancedFraudChecks ............................................................................................................. 335
advancedFraudResults ............................................................................................................. 336
affiliate .................................................................................................................................... 337
affluence ................................................................................................................................. 338
allowPartialAuth ...................................................................................................................... 339
amexAggregatorData ............................................................................................................... 340
amount .................................................................................................................................... 341
androidpayResponse ............................................................................................................. 342
applepay ................................................................................................................................. 343
applepayResponse ................................................................................................................... 344
applicationData ....................................................................................................................... 345
applicationExpirationDate ...................................................................................................... 346
applicationPrimaryAccountNumber ....................................................................................... 347
approvedAmount .................................................................................................................... 348
authAmount ............................................................................................................................. 349
authCode ................................................................................................................................. 350
authDate .................................................................................................................................. 351
authenticatedByMerchant ........................................................................................................ 352
authentication ......................................................................................................................... 353
authenticationResult ................................................................................................................ 354
authenticationTransactionId ..................................................................................................... 355
authenticationValue .................................................................................................................. 356
authInformation ....................................................................................................................... 357
authorization ............................................................................................................................ 358
authorizationResponse ........................................................................................................... 359
authorizationSourcePlatform ................................................................................................. 360
authReversal ............................................................................................................................ 361
authReversalResponse ............................................................................................................ 362
availableBalance .................................................................................................................... 363
avsResult ................................................................................................................................. 364
balanceInquiry ........................................................................................................................ 365
balanceInquiryResponse ......................................................................................................... 366
batchRequest ........................................................................................................................... 367
batchResponse ........................................................................................................................ 375
beginningBalance .................................................................................................................... 376
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>billingDate</td>
<td>377</td>
</tr>
<tr>
<td>billMeLaterRequest</td>
<td>378</td>
</tr>
<tr>
<td>billMeLaterResponseData</td>
<td>380</td>
</tr>
<tr>
<td>billToAddress</td>
<td>381</td>
</tr>
<tr>
<td>bin</td>
<td>383</td>
</tr>
<tr>
<td>bmiMerchantId</td>
<td>384</td>
</tr>
<tr>
<td>bmiProductType</td>
<td>385</td>
</tr>
<tr>
<td>bypassVelocityCheck</td>
<td>386</td>
</tr>
<tr>
<td>campaign</td>
<td>387</td>
</tr>
<tr>
<td>cancelSubscription</td>
<td>388</td>
</tr>
<tr>
<td>cancelSubscriptionResponse</td>
<td>389</td>
</tr>
<tr>
<td>capability</td>
<td>390</td>
</tr>
<tr>
<td>capture</td>
<td>391</td>
</tr>
<tr>
<td>captureAmount</td>
<td>392</td>
</tr>
<tr>
<td>captureGivenAuth</td>
<td>393</td>
</tr>
<tr>
<td>captureGivenAuthResponse</td>
<td>394</td>
</tr>
<tr>
<td>captureResponse</td>
<td>395</td>
</tr>
<tr>
<td>card</td>
<td>396</td>
</tr>
<tr>
<td>cardAcceptorTaxId</td>
<td>398</td>
</tr>
<tr>
<td>cardholderAuthentication</td>
<td>399</td>
</tr>
<tr>
<td>cardholderId</td>
<td>400</td>
</tr>
<tr>
<td>cardholderName</td>
<td>401</td>
</tr>
<tr>
<td>cardOrToken</td>
<td>402</td>
</tr>
<tr>
<td>cardProductType</td>
<td>403</td>
</tr>
<tr>
<td>cardSuffix</td>
<td>404</td>
</tr>
<tr>
<td>cardValidationNum</td>
<td>405</td>
</tr>
<tr>
<td>cardValidationResult</td>
<td>406</td>
</tr>
<tr>
<td>cashBackAmount</td>
<td>407</td>
</tr>
<tr>
<td>catLevel</td>
<td>408</td>
</tr>
<tr>
<td>ccdPaymentInformation</td>
<td>409</td>
</tr>
<tr>
<td>chargeback</td>
<td>410</td>
</tr>
<tr>
<td>checkNum</td>
<td>411</td>
</tr>
<tr>
<td>checkoutId</td>
<td>412</td>
</tr>
<tr>
<td>city</td>
<td>413</td>
</tr>
<tr>
<td>clinicOtherAmount</td>
<td>414</td>
</tr>
<tr>
<td>code</td>
<td>415</td>
</tr>
<tr>
<td>commodityCode</td>
<td>416</td>
</tr>
<tr>
<td>companyName</td>
<td>417</td>
</tr>
<tr>
<td>country</td>
<td>418</td>
</tr>
<tr>
<td>createAddOn</td>
<td>419</td>
</tr>
<tr>
<td>createDiscount</td>
<td>420</td>
</tr>
<tr>
<td>createPlan</td>
<td>421</td>
</tr>
<tr>
<td>createPlanResponse</td>
<td>422</td>
</tr>
<tr>
<td>Identifier</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>fraudResult</td>
<td>514</td>
</tr>
<tr>
<td>fundingInstructionVoid</td>
<td>515</td>
</tr>
<tr>
<td>fundingInstructionVoidResponse</td>
<td>516</td>
</tr>
<tr>
<td>fundingSource</td>
<td>517</td>
</tr>
<tr>
<td>fundingSubmerchantId</td>
<td>518</td>
</tr>
<tr>
<td>fundsTransferId</td>
<td>519</td>
</tr>
<tr>
<td>giftCardAuthReversal</td>
<td>520</td>
</tr>
<tr>
<td>giftCardAuthReversalResponse</td>
<td>521</td>
</tr>
<tr>
<td>giftCardBin</td>
<td>522</td>
</tr>
<tr>
<td>giftCardCapture</td>
<td>523</td>
</tr>
<tr>
<td>giftCardCaptureResponse</td>
<td>524</td>
</tr>
<tr>
<td>giftCardCredit</td>
<td>525</td>
</tr>
<tr>
<td>giftCardCreditResponse</td>
<td>526</td>
</tr>
<tr>
<td>giftCardResponse</td>
<td>527</td>
</tr>
<tr>
<td>giropay</td>
<td>528</td>
</tr>
<tr>
<td>giropayResponse</td>
<td>529</td>
</tr>
<tr>
<td>header</td>
<td>530</td>
</tr>
<tr>
<td>healthcareAmounts</td>
<td>531</td>
</tr>
<tr>
<td>healthcareIIAS</td>
<td>532</td>
</tr>
<tr>
<td>iban</td>
<td>533</td>
</tr>
<tr>
<td>ideal</td>
<td>534</td>
</tr>
<tr>
<td>idealResponse</td>
<td>535</td>
</tr>
<tr>
<td>IIASFlag</td>
<td>536</td>
</tr>
<tr>
<td>incomeAmount</td>
<td>537</td>
</tr>
<tr>
<td>incomeCurrency</td>
<td>538</td>
</tr>
<tr>
<td>international</td>
<td>540</td>
</tr>
<tr>
<td>intervalType</td>
<td>541</td>
</tr>
<tr>
<td>invoiceReferenceNumber</td>
<td>542</td>
</tr>
<tr>
<td>issuerCountry</td>
<td>543</td>
</tr>
<tr>
<td>itemCategoryCode</td>
<td>544</td>
</tr>
<tr>
<td>itemDescription</td>
<td>545</td>
</tr>
<tr>
<td>itemDiscountAmount</td>
<td>546</td>
</tr>
<tr>
<td>itemSequenceNumber</td>
<td>547</td>
</tr>
<tr>
<td>ksn</td>
<td>548</td>
</tr>
<tr>
<td>lastName</td>
<td>549</td>
</tr>
<tr>
<td>lineItemData</td>
<td>550</td>
</tr>
<tr>
<td>lineItemTotal</td>
<td>552</td>
</tr>
<tr>
<td>lineItemTotalWithTax</td>
<td>553</td>
</tr>
<tr>
<td>littleInternalRecurringRequest</td>
<td>554</td>
</tr>
<tr>
<td>littleOnlineRequest</td>
<td>555</td>
</tr>
<tr>
<td>littleOnlineResponse</td>
<td>556</td>
</tr>
<tr>
<td>littleRequest</td>
<td>558</td>
</tr>
<tr>
<td>littleResponse</td>
<td>559</td>
</tr>
</tbody>
</table>
Contents

originalRefCode .......................................................................................................................... 607
originalSequenceNumber ............................................................................................................. 608
originalSystemTraceId ................................................................................................................. 609
originalToken ............................................................................................................................... 610
originalTokenInfo ......................................................................................................................... 611
originalTransactionAmount .......................................................................................................... 612
originalTxnTime ............................................................................................................................ 613
origLittleTxnId ............................................................................................................................... 614
origOrderId ..................................................................................................................................... 615
cannel as a child of billToAddress and shipToAddress ....................................................... 629
password .................................................................................................................................. 616
payerId ....................................................................................................................................... 617
payFacCredit .............................................................................................................................. 618
payFacCreditResponse .................................................................................................................. 619
payFacDebit ................................................................................................................................. 620
payFacDebitResponse .................................................................................................................... 621
paymentDataType ....................................................................................................................... 622
paymentPurpose .......................................................................................................................... 623
paypage .......................................................... 624
paypageRegistrationId .................................................................................................................. 625
paypal ....................................................................................................................................... 626
payPalNotes ................................................................................................................................. 627
payPalOrderComplete .......................................................... 628
phone .................................................................................................................................. 629
    phone as a child of billToAddress and shipToAddress ....................................................... 629
    phone as a child of customBilling ........................................................................................... 629
physicalCheckCredit .................................................................................................................. 630
physicalCheckCreditResponse .................................................................................................... 631
physicalCheckDebit ..................................................................................................................... 632
physicalCheckDebitResponse ....................................................................................................... 633
pin ............................................................................................................................................ 634
planCode .................................................................................................................................. 635
pos ....................................................................................................................................... 636
postDate ................................................................................................................................. 637
postDay .................................................................................................................................... 638
preapprovalNumber ..................................................................................................................... 639
preferredLanguage .................................................................................................................... 640
prepaid ................................................................................................................................. 641
prepaidCardType ....................................................................................................................... 642
processingInstructions ................................................................................................................. 643
processingType .......................................................................................................................... 644
productCode ............................................................................................................................... 645
publicKeyHash .......................................................................................................................... 646
quantity ................................................................................................................................. 647
sepaDirectDebit .................................................................................................................. 692
sepaDirectDebitResponse ................................................................................................. 693
sequenceNumber ............................................................................................................. 694
sequenceType ................................................................................................................... 695
shipFromPostalCode .......................................................................................................... 696
shippingAmount ................................................................................................................ 697
shipToAddress ................................................................................................................... 698
signature ............................................................................................................................. 699
sofort ................................................................................................................................. 700
sofortResponse ................................................................................................................. 701
ssn ....................................................................................................................................... 702
startDate ............................................................................................................................ 703
state ...................................................................................................................................... 704
submerchantCredit ......................................................................................................... 705
submerchantCreditResponse ........................................................................................... 706
submerchantDebit ............................................................................................................. 707
submerchantDebitResponse ............................................................................................ 708
submerchantName .......................................................................................................... 709
subscription ...................................................................................................................... 710
subscriptionId .................................................................................................................... 711
surchargeAmount .......................................................................................................... 712
systemTraceId .................................................................................................................. 713
taxAmount ....................................................................................................................... 714
taxExempt ......................................................................................................................... 715
taxIncludedInTotal ........................................................................................................... 716
taxRate ............................................................................................................................... 717
taxType .............................................................................................................................. 718
taxTypeIdentifier ............................................................................................................ 719
terminalId ........................................................................................................................ 720
termsAndConditions ....................................................................................................... 721
threatMetrixSessionId ..................................................................................................... 722
token ................................................................................................................................. 723
token (Worldpay generated card number replacement) .................................................... 724
token (PayPal generated) .................................................................................................. 725
tokenMessage ................................................................................................................... 726
tokenResponse ............................................................................................................... 727
tokenResponseCode ......................................................................................................... 728
totalHealthcareAmount .................................................................................................. 729
track ................................................................................................................................... 730
track1Status ..................................................................................................................... 731
track2Status ..................................................................................................................... 732
transactionAmount .......................................................................................................... 733
transactionId ..................................................................................................................... 734
Contents

transactionId as a Child of the paypal element ................................................................. 734
transactionId as a Child of the header element ............................................................... 734
trialIntervalType .................................................................................................................. 735
trialNumberOfIntervals ....................................................................................................... 736
triggeredRule ........................................................................................................................ 737
txnTime ................................................................................................................................. 738
type ..................................................................................................................................... 739

type Element as a Child of the parent elements listed below ........................................ 739
type Element as a Child of fundingSource ....................................................................... 740
unitCost ................................................................................................................................. 741
unitOfMeasure ...................................................................................................................... 742
unload ................................................................................................................................. 743
unloadResponse .................................................................................................................. 744
unloadReversal .................................................................................................................... 745
unloadReversalResponse ................................................................................................. 746
updateAddOn ........................................................................................................................ 747
updatedCard ......................................................................................................................... 748
updateCardValidationNumOnToken .................................................................................. 749
updateCardValidationNumOnTokenResponse .................................................................. 750
updateDiscount .................................................................................................................... 751
updatePlan .......................................................................................................................... 752
updatePlanResponse ........................................................................................................... 753
updateSubscription .......................................................................................................... 754
updateSubscriptionResponse ............................................................................................ 758
updatedToken ...................................................................................................................... 759
url ....................................................................................................................................... 760
user .................................................................................................................................... 761
vendorCredit ......................................................................................................................... 762
vendorCreditResponse ........................................................................................................ 763
vendorDebit .......................................................................................................................... 764
vendorDebitResponse ........................................................................................................ 765
vendorName ....................................................................................................................... 766
verificationCode .................................................................................................................. 767
verify ................................................................................................................................ 768
version ................................................................................................................................. 769
virtualAccountNumber ....................................................................................................... 770
virtualAuthenticationKeyData ........................................................................................... 771
virtualAuthenticationKeyPresenceIndicator ...................................................................... 772
virtualGiftCard ..................................................................................................................... 773
virtualGiftCardBin ................................................................................................................. 774
virtualGiftCardResponse .................................................................................................... 775
visionAmount ...................................................................................................................... 776
void ................................................................................................................................... 777
voidResponse ........................................................................................................................... 778
wallet ....................................................................................................................................... 779
walletSourceType .................................................................................................................... 780
walletSourceTypeId .................................................................................................................. 781
yearsAtEmployer .................................................................................................................... 782
yearsAtResidence ................................................................................................................... 783
zip ............................................................................................................................................ 784

Appendix A Payment Transaction Response Codes

Payment Transaction Response Codes ................................................................................... 786
3DS Authentication Result Codes .......................................................................................... 806
AVS Response Codes ............................................................................................................. 808
AAVS Response Codes ......................................................................................................... 809
Card Validation Response Codes .......................................................................................... 812
Advanced Fraud Tools Triggered Rules .................................................................................. 813
XML Validation Error Messages ............................................................................................ 829
Additional Response Header Error Messages ........................................................................ 831
ACH Return Reason Codes .................................................................................................... 833
ACH NoC Change Codes ....................................................................................................... 836
Canadian eCheck Return Codes ............................................................................................ 837

Appendix B Credit Card Number Formats

Appendix C Test Card Numbers

Appendix D PayFac Dynamic Payout

Advantages of Using Dynamic Payout .................................................................................... 848
Overview of Dynamic Payout .................................................................................................. 849
Timing of Transactions, Reports, and Money Movement ....................................................... 849
Money Movement and Accounts ............................................................................................ 850
Same Day Funding .................................................................................................................. 852
FastAccess Funding ............................................................................................................... 852
Account Balance Verifications ............................................................................................... 853
Example of Funding Instructions ............................................................................................ 855
Funding Instruction Void Transactions .................................................................................. 858
Funding Instruction Certification Testing ................................................................................ 859
SSR Reports ............................................................................................................................ 862
Tax ID Validation Process ....................................................................................................... 864
About This Guide

This manual serves as a reference to the cnpAPI transaction formats used for payment processing with Worldpay, LLC. It also explains how to perform unattended transaction testing and attended certification testing with Worldpay, LLC.

Intended Audience

This document is intended for technical personnel who will be setting up and maintaining payment processing using the cnpAPI format.

Revision History

This document has been revised as follows:

<table>
<thead>
<tr>
<th>Doc. Version</th>
<th>Description</th>
<th>Location(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.11</td>
<td>Removed reference to support of FTP with PGP encryption.</td>
<td>Chapter 1</td>
</tr>
<tr>
<td>2.10</td>
<td>Corrected comments about supported card brands for CoF.</td>
<td>Chapters 1 and 4</td>
</tr>
<tr>
<td></td>
<td>Added info about multiple daily deliveries of settlement transactions.</td>
<td>Chapters 1 and 3</td>
</tr>
<tr>
<td></td>
<td>Added note about 20K character limit for Online request submitted by merchants using Open Access (i.e., Transact).</td>
<td>Chapter 3 &amp; Appendix A</td>
</tr>
<tr>
<td></td>
<td>Added response codes returned in the extendedCardResponse.</td>
<td>Chapter 4</td>
</tr>
<tr>
<td></td>
<td>Added a new response codes: 507 related to account updates.</td>
<td>Appendix A</td>
</tr>
<tr>
<td></td>
<td>Corrected Visa card number in Table D-3.</td>
<td>Appendix D</td>
</tr>
</tbody>
</table>
### Document Revision History

<table>
<thead>
<tr>
<th>Doc. Version</th>
<th>Description</th>
<th>Location(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.9</td>
<td>Removed all references to the Post-Live environment. Corrected approvedAmount is test 31. Removed Note about elements used for L2/L3 transactions. Updated description of Response Code 550. Updated list of Dynamic Payout SSR reports.</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix D</td>
</tr>
<tr>
<td>2.8</td>
<td>Updated Table 1-4 - Response Codes not recycled. Updated tests DI3DS2 and DI3DS3 to return correct results. Added Amazon Pay related Response Codes.</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix A</td>
</tr>
<tr>
<td>2.7</td>
<td>Added new FastAccess Funding cert tests (replaces old tests)</td>
<td>Appendix D</td>
</tr>
<tr>
<td>2.6</td>
<td>Rewrote tokenization section for OmniTokens. Removed JCB entry from Credit Card Numbers table. The info in the Discover Card section still applies.</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix C</td>
</tr>
<tr>
<td>2.5</td>
<td>Removed Partial Approval bullet from Healthcare Card section. Changed test card number in Table 2-10 (orderid Sale111111).</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 2</td>
</tr>
<tr>
<td>2.4</td>
<td>Changed Dynamic Payout overbalance message.</td>
<td>Appendix D</td>
</tr>
<tr>
<td>2.3</td>
<td>Added Amazon Pay info and Cert tests.</td>
<td>Chapters 1 and 2</td>
</tr>
<tr>
<td>2.2</td>
<td>Changed eCheck to Direct Debit where applicable. Added Partial Approval bullet to IISA requirements. Added test card and Bin range for UnionPay 81 series card. Removed material for Sepa, IDEAL, Sofort, and Giropay. Added Not Supported notes to Sepa, IDEAL, Sofort, and Giropay elements. Changed text to reflect new behavior for balance checking of Funding Instruction Batch files.</td>
<td>All</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix B and C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapters 1, 2, and 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix D</td>
</tr>
<tr>
<td>2.1</td>
<td>Added Cert test for Refund Authorization. Expanded the description of the authenticationValue element to include its use in the Apple Pay Merchant Decrypt scenario. Also added Note to originalTransactionId element. Added Response Code 942. Corrected cut-off time for Same Day Funding from 12:00 PM to 11:00 AM ET.</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chapter 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix D</td>
</tr>
<tr>
<td>2.0</td>
<td>Re-branded document from Vantiv to Worldpay</td>
<td>All</td>
</tr>
<tr>
<td>1.13</td>
<td>Fixed broken link to Pre-live self-provisioning site. Updated BIN ranges for Discover.</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appendix B</td>
</tr>
<tr>
<td>Doc. Version</td>
<td>Description</td>
<td>Location(s)</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>1.12</td>
<td>Added Fast Access Funding Instruction Certification tests.</td>
<td>Appendix D</td>
</tr>
<tr>
<td>1.11</td>
<td>Returned IIAS/Healthcare support info to the document. Corrected results for test Net_Id3. Added note to totalHealthcareAmount.</td>
<td>Chapters 1 and 2, Chapter 2, Chapter 4</td>
</tr>
<tr>
<td>1.10</td>
<td>Updated Android Pay to Google Pay. Added Note to secondaryAmount element.</td>
<td>Chapter 1, Chapter 4</td>
</tr>
<tr>
<td>1.9</td>
<td>Added info about Response Reason Code 825. Added info about allowing 60 second gap between a transaction and submitting a Void. Added processingType enums for Card on File transactions.</td>
<td>Appendix A, Chapters 1, 3, &amp; 4, Chapter 4</td>
</tr>
<tr>
<td>1.8</td>
<td>Remove reference to Recurring Snapshot report. Updated Android Pay Certification test results. Corrected maxLength of id attribute, which increased from 25 to 36 in V11.3.</td>
<td>Chapter 1, Chapter 2, Chapter 4</td>
</tr>
<tr>
<td>1.7</td>
<td>Removed Mobile Point of Sale sections from guide Updated document for release of V11.3, including Online Dynamic Payout Funding Instructions and Fast Funding transaction. Fixed various links to github and test environments. Fixed error in definition of customerReference element. Added info about support for Visa token transactions with 3-D Secure (TAVV and DTVV), as well as new 3-DS response code definitions.</td>
<td>Chapters 1 and 2, All, Chapter 2, Chapter 4, Chapter 4 &amp; Appendix A</td>
</tr>
<tr>
<td>1.6</td>
<td>Added Response Codes 940 and 941. Added text clarifying limit of 25K for Same day Funding transaction and removed debit transaction restrictions. Added note about 30 minute wait for funds to be restored to Settlement account balance after issuing a Funding Instruction Void transaction.</td>
<td>Appendix A, Appendix D, Appendix D</td>
</tr>
<tr>
<td>1.5</td>
<td>Added/modified information in section 2.1. orderId element added to accountUpdateResponse. Removed &quot; not yet supported &quot; note from checkoutId element and added Cert tests Modified requirements for Level 2/3 data (enhancedData element).</td>
<td>Chapter 2, Chapter 4, Chapters 2 and 4, Chapter 4</td>
</tr>
<tr>
<td>1.4</td>
<td>Corrected Response Code for Test GC17 - 281 changed to 218. Updated Litle to Vantiv in Response messages. Added Note about expDate to token element definition.</td>
<td>Chapter 2, All, Chapter 4</td>
</tr>
</tbody>
</table>
### Table 1: Document Revision History

<table>
<thead>
<tr>
<th>Doc. Version</th>
<th>Description</th>
<th>Location(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3</td>
<td>Replaced LitleXML with cnpAPI. Added three new tests for new MC 3DS behavior.</td>
<td>All</td>
</tr>
<tr>
<td></td>
<td>Moved Dupe Tests from Optional to Required test group.</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td>Added info about checkoutId element (not GA).</td>
<td>Chapter 3</td>
</tr>
<tr>
<td></td>
<td>Added info about sameDayFunding Batch Request attribute.</td>
<td>Chapter 4 &amp; Appendix D</td>
</tr>
<tr>
<td></td>
<td>Added info about SOFORT and Giropay alternate payment methods.</td>
<td>Chapters 1, 2, and 4</td>
</tr>
<tr>
<td></td>
<td>Revised/renamed Customer Insights to Issuer Insights.</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td>Corrected several examples, replacing Response Code 001 - Transaction Received with 000 - Approved.</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>1.2</td>
<td>Corrected maintenance window times for Pre-/Post-Live.</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td>Added &quot;Appears in Declined Transaction report.&quot; note to certain Response Codes.</td>
<td>Appendix A</td>
</tr>
<tr>
<td>1.1</td>
<td>Added &lt;customIdentifier&gt; to several eCheck transactions.</td>
<td>Chapters 3 and 4</td>
</tr>
<tr>
<td></td>
<td>Added information about the iDEAL international payment method, which is now GA.</td>
<td>Chapters 1, 2, and 4</td>
</tr>
<tr>
<td>1.0</td>
<td>Initial release of LitleXML Reference Guide for schema V11.0, including new Gift Card transactions and international alternate payment methods.</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Document Structure

This manual contains the following sections:

**Chapter 1, "Introduction"**
This chapter provides an introduction to transaction processing using cnpAPI.

**Chapter 2, "Testing Your cnpAPI Transactions"**
This chapter provides information concerning the testing and certification process, which you must complete prior to submitting transactions to the Worldpay eComm production environment.

**Chapter 3, "cnpAPI Transaction Examples"**
This chapter provides information concerning the cnpAPI structure for transaction submission, as well as examples.

**Chapter 4, "cnpAPI Elements"**
This chapter provides definitions and other information concerning each cnpAPI element.

**Appendix A, "Payment Transaction Response Codes"**
This appendix lists all of the possible response codes and messages.

**Appendix B, "Credit Card Number Formats"**
This appendix provides information about credit card number formats and Mod-10 validation.

**Appendix C, "Test Card Numbers"**
This appendix provides credit card number that can be used for testing.

**Appendix D, "PayFac Dynamic Payout"**
This appendix provides information about PayFac Dynamic Payout transactions.

Documentation Set

For additional information concerning the Worldpay, LLC application, see any of the following guides in the documentation set:

- *iQ Reporting and Analytics User Guide*
- *Worldpay eComm Chargeback API Reference Guide*
- *Worldpay eComm Chargeback Process Guide*
- *Worldpay eComm Partner Integration Overview Guide*
- *Worldpay eComm PayPal Integration Guide*
- *Worldpay eComm PayFac API Reference Guide*
- *Worldpay eComm PayFac Portal User Guide*
- *Worldpay eComm PayFac Integration Overview Guide*
About This Guide

- Worldpay eComm eProtect Integration Guide
- Worldpay Enterprise eProtect Integration Guide
- Worldpay eComm XML Differences Guide
- Worldpay eComm Secure Scheduled Reports Reference Guide
## Typographical Conventions

Table 2 describes the conventions used in this guide.

**TABLE 2** Typographical Conventions

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>. . .</td>
<td>Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.</td>
</tr>
<tr>
<td>. . .</td>
<td>Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted.</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>Angle brackets are used in the following situations:</td>
</tr>
<tr>
<td></td>
<td>• user-supplied values (variables)</td>
</tr>
<tr>
<td></td>
<td>• XML elements</td>
</tr>
<tr>
<td>[ ]</td>
<td>Brackets enclose optional clauses from which you can choose one or more option.</td>
</tr>
<tr>
<td>bold text</td>
<td>Bold text indicates emphasis.</td>
</tr>
<tr>
<td><em>Italicized text</em></td>
<td>Italic type in text indicates the name of a referenced external document.</td>
</tr>
<tr>
<td>blue text</td>
<td>Blue text indicates either a hypertext link or an element name (in xml examples).</td>
</tr>
<tr>
<td>monospaced text</td>
<td>Used in code examples and elsewhere to designate field/element names.</td>
</tr>
</tbody>
</table>
Contact Information

This section provides contact information for organizations within Worldpay.

**Chargebacks** - For business-related issues and questions regarding financial transactions and documentation associated with chargeback cases, contact the Chargebacks Department.

Chargebacks Department Contact Information

<table>
<thead>
<tr>
<th>Telephone</th>
<th>1-844-843-6111 (option 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail</td>
<td><a href="mailto:chargebacks@vantiv.com">chargebacks@vantiv.com</a></td>
</tr>
<tr>
<td>Hours Available</td>
<td>Monday – Friday, 7:30 A.M.– 5:00 P.M. EST</td>
</tr>
</tbody>
</table>

**Technical Support** - For technical issues such as file transmission errors, email Technical Support. A Technical Support Representative will contact you within 15 minutes to resolve the problem.

Technical Support Contact Information

<table>
<thead>
<tr>
<th>Telephone</th>
<th>For critical production issues only: 1-888-829-1907</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail</td>
<td><a href="mailto:ecommercesupport@worldpay.com">ecommercesupport@worldpay.com</a></td>
</tr>
<tr>
<td>Hours Available</td>
<td>Monday – Friday, 8:00 A.M.– 6:00 P.M. EST</td>
</tr>
</tbody>
</table>

**Relationship Management/Customer Service** - For non-technical issues, including questions concerning IQ Reporting and Analytics, help with passwords, modifying merchant details, and changes to user account permissions, contact the Relationship Management/Customer Service Department. If you are a Payment Facilitator, refer to the second table.

Relationship Management/Customer Service Contact Information - Merchants

<table>
<thead>
<tr>
<th>Telephone</th>
<th>1-844-843-6111 (Option 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail</td>
<td><a href="mailto:ecomcustomercare@worldpay.com">ecomcustomercare@worldpay.com</a></td>
</tr>
<tr>
<td>Hours Available</td>
<td>Monday – Friday, 8:00 A.M.– 6:00 P.M. EST</td>
</tr>
</tbody>
</table>

**PayFac Account Management/Customer Service** - For non-technical issues, including questions concerning the user interface, help with passwords, modifying merchant details, and changes to user account permissions, contact the PayFac Account Management/Customer Service Department.

TABLE 3  PayFac Account Management/Customer Service Contact Information

<table>
<thead>
<tr>
<th>Telephone</th>
<th>1-844-843-6111 (Option 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-mail</td>
<td><a href="mailto:PayFacEComm@worldpay.com">PayFacEComm@worldpay.com</a></td>
</tr>
<tr>
<td>Hours Available</td>
<td>Monday – Friday, 8:00 A.M.– 5:00 P.M. EST</td>
</tr>
</tbody>
</table>
Technical Publications - For questions or comments about this document, please address your feedback to the Technical Publications Department. All comments are welcome.

Technical Publications Contact Information

<table>
<thead>
<tr>
<th>E-mail</th>
<th><a href="mailto:TechPubs@vantiv.com">TechPubs@vantiv.com</a></th>
</tr>
</thead>
</table>
Introduction

The cnpAPI data format supports two types of transaction submission methods: Online and Batch. With the Online method, you submit each transaction independently and receive a response in real-time. Typically, merchants use the Online method for Authorization transactions, as well as transactions available only via Online (for example, Void). The Batch method enables you to submit multiple transactions simultaneously in a single Session file. Worldpay recommends the Batch method for all transaction submissions except Authorizations and the transaction available only via Online.

This chapter provides an overview of the cnpAPI data format, including some basic XML coding requirements. Also discussed are the advantages of using batch processing, duplicate transaction detection, report groups, Value Added Services, and supported transaction types.

The topics discussed in this chapter are:

- The cnpAPI Data Format
- Batch Transaction Processing
- Payment Integration Platform (cnpAPI SDKs)
- Duplicate Transaction Detection
- Coding for Report Groups
- Recovery
- Recurring Engine
- Issuer Insights
- Fraud Toolkit
- Tokenization Feature
- Direct Debit Processing
- eCommerce Solution for Apple Pay™
- eCommerce Solution for Google Pay™
- Amazon Pay
- Healthcare Card Feature
- Supported Transaction Types
1.1 The cnpAPI Data Format

There are several advantages to using the cnpAPI format as follows:

- **Easier Implementation, Operations, and Debugging** - Compared to fixed length or binary formats, the XML format is considerably easier for operations staff to read and edit, using virtually any text editor. This allows Worldpay’s Implementation, First Line Support, and Relationship Managers to quickly communicate any issues and work with your own operations staff to make necessary corrections without worrying about line lengths, padding or encoding.

- **Fewer Downgrades** - Since the cnpAPI format allows you to explicitly tie deposits to their associated authorizations via the `<litleTxnId>` element, your transactions qualify for the best interchange rates at a higher frequency than with formats that do not support this transaction cross-referencing.

- **Simpler Capture (Deposit) and Refund Transactions** - Because the cnpAPI format associates related transactions using the `<litleTxnId>` element, our format does not require you to resubmit all of the authorization information on a deposit nor all of the deposit information on a refund. When you submit the unique transaction id, Worldpay automatically pulls the information from the original transaction. Most other formats require you to resubmit the related data with each transaction.

- **Superior Reporting** - The cnpAPI format allows you to separate your transactions into different categories by specifying a Report Group on each transaction. When accessing your data on the iQ reporting and Analytics Interface, this feature allows you to filter your financial reports by Report Groups, providing more granular detail based on a reporting hierarchy the Report Groups create. Most other formats restrict reporting categories to a batch or specific merchant id.

- **Improved Chargeback Management** - Unlike most other formats where transactional relationships can be a “best guess” proposition, the cnpAPI format explicitly ties related transactions, allowing you and Worldpay to see authorization-to-deposit and deposit-to-refund relationships with precision. This knowledge is indispensable when fighting chargebacks.

- **New Features** - New features developed for the Worldpay eCommerce platform are first exposed via the XML API. While the SDKs add new capabilities shortly after development, direct coding to the XML API gains access to these feature and capabilities earlier.

1.1.1 Communications Protocols

Worldpay supports the submission of Batch transactions to Worldpay eCommerce platform for processing via sFTP and Online submissions via HTTPS POST, as shown in Table 1-1.

**NOTE:** Although the Worldpay eComm platform supports both TLS 1.1 and TLS 1.2, Worldpay strongly recommends the use of TLS 1.2 to ensure the best encryption security.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Encryption</th>
<th>Batch</th>
<th>Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPS Post</td>
<td>TLS 1.2</td>
<td>N/A</td>
<td>Required</td>
</tr>
<tr>
<td>sFTP</td>
<td>SSH Key</td>
<td>Recommended</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The Worldpay eComm platform supports the following ciphers for the TLS 1.2 encryption protocol:
1.1.2 General XML Coding Requirements

As part of the on-boarding process, you receive XML schema files from your Implementation Consultant. Using those files and this document as a guide, you create the required XML documents for submission of your transactions. You should validate all XML you create using the supplied schema. Also, working with your Implementation Consultant, you are required to perform various tests of your XML (see Chapter 2, "Testing Your cnpAPI Transactions") prior to submitting transactions to the production environment.

In addition to the process outlined above, there are a few XML basics of which you should be aware.

- Encode all data using the UTF-8 format.
- Although it is not required, Worldpay recommends that when formatting your XML, you keep each element on its own line. This will aid in debugging situations where an error message specifies an issue in a particular line of XML code (for example, line 20).
- Be aware of special characters that require specific handling (see Table 1-3). For example, the less than (<) and greater than (>) symbols define element tags in the XML code. Using the entity names &lt; and &gt; instead of < and > prevents a browser from interpreting these characters as element brackets.

Be sure to review data provided by customers for special character handling. For example, an address of "4th & Main," must be rewritten as "4th & Main" (including quotes) before being submitted via XML. Failure to quote this type of input causes rejection of XML submissions due to syntax errors.

### TABLE 1-2 Supported TLS Ciphers

<table>
<thead>
<tr>
<th>Server Priority</th>
<th>Cipher Id</th>
<th>Cipher Name</th>
<th>TLS 1.2 Support</th>
<th>TLS 1.1 Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>c030</td>
<td>TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>c02f</td>
<td>TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>c028</td>
<td>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>c014</td>
<td>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>c027</td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>c013</td>
<td>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>009d</td>
<td>TLS_RSA_WITH_AES_256_GCM_SHA384</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>009c</td>
<td>TLS_RSA_WITH_AES_128_GCM_SHA256</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>9</td>
<td>003d</td>
<td>TLS_RSA_WITH_AES_256_CBC_SHA256</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>003c</td>
<td>TLS_RSA_WITH_AES_128_CBC_SHA256</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>11</td>
<td>0035</td>
<td>TLS_RSA_WITH_AES_256_CBC_SHA</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>002f</td>
<td>TLS_RSA_WITH_AES_128_CBC_SHA</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
1.1.3 Other XML Resources

There are several Internet sites that provide both reference and educational information that may help you when implementing your XML. A few of these sites are:

- http://www.w3schools.com/xml/xml_syntax.asp
- http://www.w3.org/

### TABLE 1-3 Coding for Special Characters

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
<th>Entity Reference (case sensitive)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>less than</td>
<td>&lt;</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
<td>&gt;</td>
</tr>
<tr>
<td>“</td>
<td>quotation</td>
<td>&quot;</td>
</tr>
<tr>
<td>‘</td>
<td>apostrophe</td>
<td>'</td>
</tr>
<tr>
<td>&amp;</td>
<td>ampersand</td>
<td>&amp;</td>
</tr>
</tbody>
</table>
1.2 Batch Transaction Processing

Batch processing involves a group of transactions submitted in a single file. In the case of a cnpAPI Batch, the parent or root element is the `<litleRequest>` element. A single `<litleRequest>`, referred to as a Session, can contain many batches and each batch can contain multiple transactions. We recommend that you use Batch processing for all transaction types except Authorizations and Voids (Online only).

Some of the advantages of using Batch processing are:

- **Better Performance** - We optimize batch processing by processing multiple transactions in the batch simultaneously. This allows you to process thousands of transactions quickly without writing complicated logic or managing complicated processes.

- **Easier Reconciliation** - When processing a batch, all transactions within that batch post on the same day. In the case of Online transactions, you could submit two transactions at the same time and one could post today and the other tomorrow. This can cause confusion in your accounting process.

- **Easier Error Recovery** - A batch processes as a single unit, thus if you experience any system or communication issue while processing a batch, you can easily determine if the file was processed. With Online transactions, determining which individual transactions were not processed can be more difficult.

1.2.1 Recommended Session File Size

As stated above, a Session file is a group of batches. A batch is a set of transactions for a single merchant. Normally, you send in a single file which has one batch for each merchant. This works well when the overall number of transactions is small. The number of transactions you should submit in any individual Session or Batch depend on a number of factors, including whether or not you are an individual merchant or a presenter submitting transactions for multiple merchants. In general, you should keep the following recommendations and rules in mind when determining the number of transactions you submit in an Session/Batch file:

- A Batch should not exceed 20,000 transactions. If the number of transaction for a single merchant exceeds 20,000, you should create multiple batches for the same merchant, each batch containing not more than 20,000 transactions.

- A Batch should not contain only one transaction, unless the merchant has only one transaction for the day.

- A Session file must never contain more than 9,999 Batches.

- A Session file must never contain more than 1,000,000 transactions across all Batches.

- Always allow sufficient time between your submission time and your cut-off time for the processing of the Session. Larger files take longer to process.

1.2.2 Multiple Daily Delivery

Starting on October 16, 2019, Worldpay will send multiple settlement files to the card networks each day. We will send settlements files to the networks three hours after the timestamp of the submitted transaction rounded up to the next hour. This enables us to align with the network settlement cutoff times.

This delivery schedule does impact Void transactions. If you wish to void a settlement transaction, you must submit the void within three hours of submitting the settlement transaction. If it’s outside the
three-hour window, you must send a Credit transaction to make the cardholder whole. Similarly, you will have three hours to request us to discard a full Batch file with deposits.

There are two possible cases:

Case 1: End of Merchant Fiscal Day (or merchant Cutoff) Void Window

The three-hour window for Voids does not apply to the end-of-day delivery. This is how the system works now and will continue to work. If a your cutoff time is 11 p.m., you cannot void a transaction after 11 p.m. So, if your cut-off time is 11:00 PM and you submit a settlement transaction any time after 8:00 PM, you will have less than three hours to Void the transaction.

Case 2: Multiple Daily Settlement Void Window

For any settlement transactions you submit more than three hours prior to your cut-off time you will always have a minimum of three hours to void the transaction or discard a batch. This is independent of when our deliveries schedule. When you submit a settlement transaction, we apply a time stamp of the top of the next hour. For example, an Online transaction or a Batch that you submit at 1:05 PM receives a 2 p.m. time stamp. We determine the time stamp for Batch files based upon when you complete the submission, not when you start. For both types of submissions, you have three hours from the time stamp to submit a void or discard the Batch. So, you would have until 5:00 PM to void a transaction or discard a Batch submitted at 1:05 PM (i.e., 2:00 PM time stamp + three-hour window = 5:00PM).
1.3 Payment Integration Platform (cnpAPI SDKs)

In order to facilitate integration to the platform, Worldpay provides several language specific SDKs (Software Development Kits). The developers page on our website (http://www.vantiv.com/developers) provides links to SDK libraries for several popular languages, including:

- PHP
- Ruby
- Java
- .NET
- Python

Extensions for:

- Magento
- Opencart

In addition to the SDKs and extensions, Worldpay provides examples of each supported transaction type, as well as demonstration applications. Once you install the library appropriate to your language, the Worldpay eComm Sandbox, which functions as an emulator of our production environment, is available to validate your transaction format.
1.4 Duplicate Transaction Detection

In order to help you avoid transaction errors, Worldpay performs duplicate transaction checking for both Online and Batch transaction submissions. While we use a robust duplicate checking methodology, we cannot guarantee our system will catch all duplicates and bear no responsibility for correcting the impact of erroneously submitted transactions. This section discusses the different checking methodologies used depending upon the type of submission.

NOTE: For tokenized transactions, the token is used in place of the card numbers by the Duplicate Transaction Detection process.

For PayPal transactions a combination of the PayPal Id + the (consumer’s) email is used by the Duplicate Transaction Detection process.

For transactions submitted using other formats (e.g., PTI, Nabanco, etc.), the logic used to detect duplicates for these supported, but foreign APIs, is less robust and may miss duplicates in certain scenarios. Worldpay recommends, for better dupe checking, convert to the cnpAPI format, as soon as possible.

1.4.1 Batch Duplicate Checking

When processing a Batch, the system acts to detect duplicate transactions for the following transaction types: Authorization, Auth Reversal, Capture, Force Capture, Capture Given Auth, Credit, Sales, eCheck Credit, and eCheck Sales.

NOTE: For information about duplicate checking of Dynamic Payout Funding Instructions see Batch Dupe Checking for Dynamic Payout Funding Instructions on page 9.

For each of these transaction types, the application compares the transaction type, transaction amount, the <orderId> element from the request, credit card number, and credit card expiration date against transactions in other Batch processed within the previous five days. If the characteristics of the new transaction match a previously processed transaction, the system marks it as a duplicate.

The system only performs duplicate detection against valid transactions from the previous five days. For example, if an Authorization request matches a declined Authorization from the previous day, the system would not count it as a duplicate, because the declined Authorization was not a valid Authorization.

Also, a Batch must be processed completely to be included in the previous five days of data. For example, if multiple submitted Batches are processing simultaneously, the system will not compare the transactions in one batch with the transactions in the other, because neither has completed processing. For this same reason the system cannot detect duplicate transactions within the same Batch.

If the system detects ten consecutive duplicate transactions or if the number of duplicate transactions is greater than or equal to 25% of the total transactions in the batch, the system flags the Batch as a duplicate. When either threshold is met, Worldpay will not process the Batch. If neither threshold is met, Worldpay continues processing the Batch, including any transactions that may have been duplicates.
1.4.1.1 Batch Dupe Checking for Dynamic Payout Funding Instructions

Duplicate checking for funding instructions takes place at the Batch level. The system compares a submitted Batch of instructions to other Batches submitted and accepted on the same day. If a submitted Batch has the same totals and counts as a Batch accepted on the same day, the system rejects the new Batch as a duplicate. If you resubmit a previously rejected Batch, it will not fail dupe checking, because the initial submission was not accepted and is not included in dupe checking comparisons.

**NOTE:** If you believe we rejected a funding instruction Batch in error, please contact your Partner Account Manager. If necessary, we can disable the dupe check feature for a particular Batch.

1.4.2 Online Duplicate Checking

When processing an Online transaction, the system acts to detect duplicate transactions for the following transaction types: Auth Reversal, Capture, Force Capture, Capture Given Auth, Credit, Sales, eCheck Credit, eCheck Sales, eCheck Void, and Void, as well as Gift Card transactions.

For most transactions, the system compares the transaction type, the `id` attribute from the request, and the credit card number against other Online transactions processed within the previous two days. For transactions that reference other transactions (for example, a deposit referencing an authorization or a refund referencing a deposit), the system compares the transaction type, `id` attribute, and the card number from the referenced transaction (i.e. the transaction identified by the `<litleTxnId>` element) against other Online transactions processed within the previous two days.

The system only performs duplicate detection against valid transactions. For example, if a Capture request matches a declined Capture from the previous day, the system would not count it as a duplicate, because the declined Capture was not a valid transaction.

**NOTE:** While it is uncommon, under certain circumstances network latency may cause a duplicate Sale transaction to go undetected. This can occur if you submit a second, duplicate Sale transaction while the response from the network for the Authorization portion of the first transaction is sufficiently delayed such that the first Sale has not been recorded as a valid transaction in the system.

If you elect to submit Online Sale transactions, Worldpay recommends a timeout setting of not less than 60 seconds to reduce the chances of undetected duplicate Sale transactions.

If the system determines a transaction to be a duplicate, The duplicate transactions appears in the Declined Transaction report with a Response Reason Code of 251 - Duplicate Transaction. You can access this report in Worldpay iQ or via the Worldpay eComm Secure Scheduled Report. The iQ version provides information in near real-time, while the SSR version runs daily, providing information for the transaction submitted the previous day.

**NOTE:** If you do not receive a response for a submitted transaction, Worldpay recommends you use the queryTransaction to determine the status of the original transaction (see Status Query Transactions (Online Only) on page 280).
1.5 Coding for Report Groups

You use Report Groups (reportGroup attribute) to separate your transactions into different categories, so you can view the financial reports by your specific report group names. If you are unsure what groupings to use, your Relationship Manager can help you determine the best practice for your business.

**NOTE:** Creation of an excessive number of Report Groups (in excess of 250) will impact the amount of time require to compile various reports in the iQ. Report Groups are intended to allow you to segregate transactions by logically grouping them into the different segments of your business.

Report Groups are not intended to track sales or marketing programs that exist for limited times. For this type of tracking, Worldpay provides other transaction tagging methods detailed in Additional/Alternate Methods of Tagging Transactions on page 11.

**Example:** Report Groups

The merchant, Demo, wants to separate their domestic and international sales information. To do this, the company submits all domestic transactions using reportGroup = "Domestic Business", and all international transactions using reportGroup = "International Business". When they access the Authorization Report in the iQ using the By Reporting Group tab, the transactions would be separated as shown in Figure 1-1.

**NOTE:** The reportGroup attribute is case and space sensitive. A reportGroup = “Picture Frame” is a different report group than a reportGroup = “pictureframe”.

![Figure 1-1 Report Group Example - 2 Groups](image)

The plus sign next to the Domestic Business report group signifies that there are child groups present. When fully expanded (see Figure 1-2), the iQ shows a report group hierarchy with information for the Domestic Business group further separated into Service A and Service B groups and Service A containing two additional child groups. If you find it necessary to establish this type of nested hierarchy, your Implementation Consultant will assist you.
1.5.1 Additional/Alternate Methods of Tagging Transactions

If you are using schema version 7.x or above you can use the merchantData element and its children to tag transactions (Authorization, Sale, Credit, Force Capture, Capture Given Auth, eCheck Sale, and eCheck Credit) with additional information. The three children of merchantData: campaign, affiliate, and merchantGroupingId, allow you to designate transactions as members of different groups enabling a deeper analysis of sales patterns.

For example, if the merchant from the previous example were trying a new sales initiative for Product 2 during the month of September. They plan to run ads in Boston and New York to test the new offering. To allow a deeper analysis of sales resulting from the new campaign, they can add the campaign element with a value of "September Ads" to the transactions originating in both test market. They can also include the merchantGroupingId with values that reflect the city where the order originates. By exporting either the Session report or the NSS by Transaction report from the iQ, the company can sort their sales data based upon these fields and gain a better understanding of the effectiveness of the sales campaign.

**NOTE:** The transaction tagging elements described above appear in the exported Session and NSS by Transaction reports. They are also visible within the iQ in the new Transaction Detail reports.
1.6 Recovery

Recovery is a bundle of services that include both Account Updater and Recycling Engine. By combining these two managed services into a single bundle, Worldpay simultaneously increases your approval rates, optimizes customer lifetime value, and improves your cash flow, while reducing the cost of implementing the individual features separately in terms of IT resources. For additional information about the capabilities included in this bundle, please refer to Recycling Engine on page 12, and (AAU) Account Updater Service on page 14.

1.6.1 Authorization/Sale Recycling

Authorization recycling is the process of retrying declined authorization attempts. Every merchant, especially those with a business model that uses recurring or installment payments, should devise a strategy for dealing with declined authorizations. As part of optimizing their operations, merchants must devise plans for both the timing and number of recycling attempts before contacting the cardholder or risking interruption of service. If a merchant does not recycle enough, they risk losing customers and revenue; whereas, if the merchant recycles too often, they risk increasing their total cost of payments. Implementing an optimal recycling strategy aids customer retention and therefore yields higher revenues, while lowering the costs of payment acceptance and improving cash flow.

1.6.1.1 Recycling Engine

The Recycling Engine is a managed service that automatically retries declined authorization attempts on your behalf. It requires little or no IT investment on your part. Also, implementing the Worldpay service removes the need to plan your own recycling strategy.

Recycling Engine has the following benefits:

- Increases approval rates
- Shortens time to approval, improving cash flow
- Reduces the number of authorization retries
- Lowers the risk of account/service cancellation

In order to determine the most effective recycle timing, Worldpay performs statistical analysis of past recycling attempts across our entire merchant portfolio. This analysis examines many factors, including method of payment, response codes, and transaction amount among others, to determine the optimum intervals between attempts to obtain a successful authorization. When you receive a declined Authorization, the system automatically queues the transaction for a retry at a designated time. Recycling of the Auth continues until it is either successful or the algorithm determines that it is no longer advantageous to retry.

NOTE: For Visa transactions, the Recycling Engine will retry declined Authorizations a maximum of 4 times within 16 days, per Visa regulations. For MasterCard and Discover transactions, we retry declined Authorizations a maximum of 8 times within 28 days.

Worldpay provides the results of the recycling efforts to you in a Batch posted daily to your Worldpay eComm sFTP account. This file contains transactions that either approved or exhausted the recycling pattern on the previous day. If you submit an Authorization for a transaction in the recycling queue,
Worldpay returns the response from the last automatic recycling attempt. To halt recycling of a particular transaction, submit either an Authorization reversal transaction, if the original transaction was an Auth, or a Void transaction, if the original transaction was a Sale.

**Declined Transactions Not Recycled**

Table 1-4 below provides information about Response Codes not recycled by the Recycling Engine, except under the conditions noted. The <recycleEngineActive> element in the response files indicates if the transaction is being handled by the Recycling Engine.

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Message</th>
<th>Recycling Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>213</td>
<td>Pickup Card - Lost Card</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>214</td>
<td>Pickup Card - Stolen Card</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>229</td>
<td>Illegal Transaction</td>
<td>Does not recycle.</td>
</tr>
<tr>
<td>301</td>
<td>Invalid account number</td>
<td>Not recycled for Mastercard, unless updated account information is available.</td>
</tr>
<tr>
<td>303</td>
<td>Pick up Card</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>304</td>
<td>Lost/Stolen Card</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>305</td>
<td>Expired Card</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>308</td>
<td>Restricted Card - Chargeback</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>309</td>
<td>Restricted Card - Prepaid Card Filtering Service</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>312</td>
<td>Restricted Card - International Card Filtering Service</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>315</td>
<td>Restricted Card - Auth Fraud Velocity Filtering Service</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>318</td>
<td>Restricted Card - Auth Fraud Advice Filtering Service</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>319</td>
<td>Restricted Card - Fraud AVS Filtering Service</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>323</td>
<td>No such Issuer</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>328</td>
<td>Cardholder requested that recurring or installment payment be stopped</td>
<td>Does not recycle</td>
</tr>
</tbody>
</table>
Introduction

Transaction Signature

The Recycling Engine analyzes each Authorization or Sale request message to determine if it is a new request. The result of the analysis determines if the transaction should be added to the recycling pool upon decline or if the system should intercept the transaction to prevent a duplicate transaction entering the recycling pool. To perform the analysis, the system checks the transaction signature. Depending upon your configuration, the transaction signature can be:

- Value of the <recycleId> element
- Value of the <orderId> element
- Values of the <orderId>, <number>, and <amount> elements

**NOTE:** If you submit a transaction with the identical signature, but containing new information (for example, a new card number), the system updates the transaction in the recycling pool with the new info and continues to recycle.

### Additional Configuration Options

The Recycling Engine allows you the additional flexibility of excluding certain transactions from automatic recycling. You can exclude transactions manually by including the <recycleBy> element set to None. There are also global controls that allow you to exclude transactions based upon either submission by a particular presenter, or based upon the transaction type (authorization or sale). Please consult your Relationship Manager about the global options, since they must be configured in your Worldpay eComm Merchant Profile.

#### 1.6.2 Account Updater Service

Credit and debit card numbers change for a variety of reasons including card expirations, card product type upgrades, portfolio conversions, and compromised account numbers among others. For merchants who offer services that are billed on a recurring or installment basis (for example, web hosting, gym memberships, specialized social networking, career services, monthly donation plans, etc.) out-of-date payment information can result in lost revenue, involuntary churn and decreased customer satisfaction.

Prior to the development of the Account Updater service, the standard method for merchants to obtain updated account information was to submit a Batch containing existing card information, requesting that Worldpay check for updates. Typically, merchants request updates for customer accounts scheduled to be billed in the next billing cycle. This legacy method is a relatively slow process, requiring several days for Worldpay to accumulate responses from the card networks/issuers and then to make the response file available to the merchant. Merchants must then update their billing systems with the new information, requiring IT processing cost. Failure to update their files can result in multiple requests (and charges) for

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Message</th>
<th>Recycling Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>358</td>
<td>Restricted by due to security code mismatch</td>
<td>Only recycled if card repair information is available.</td>
</tr>
<tr>
<td>550</td>
<td>Restricted Device or IP - ThreatMetrix Fraud Score Below Threshold</td>
<td>Only recycled if card repair information is available.</td>
</tr>
</tbody>
</table>

**TABLE 1-4**  Response Codes Not Recycled (except as noted)
The update information, as well as delays in or lost revenue, higher Authorization expenses, and possibly chargebacks when old account information is used.

The Account Updater service shifts the workload of obtaining and maintaining updated account information to Worldpay. Utilizing configurable scheduling algorithms, we initiate account update requests on your behalf and then store the updated card information for use in future transactions. You simply submit billing transactions normally and, if necessary, Worldpay updates the transaction with the stored card information before submitting it to the networks for authorization. This fully managed service requires no code update on your systems.

**FIGURE 1-3** Account Updater Overview

---

1.6.2.1 **Match Back**

If you decide you wish to have the updated card information returned to you, Worldpay offers the Match Back option. In this case, you can opt to receive updated information either in a Batch deposited to the merchant sFTP account, or in the XML transaction response messages. Once you update your systems, you can resubmit the failed transaction with the new card information. If, after receiving an update, you...
submit a transaction with the old information, systems detect that you are using the old data and update the transaction for you prior to submitting it to the networks for authorization.

Please consult your Relationship Manager for additional information and configuration options.

**FIGURE 1-4** Match Back Overview

### 1.6.2.2 Merchant Requirements

In order to use the Account Updater service, you must first apply for membership to the following:

- MasterCard Automatic Billing Updater
- Visa Account Updater
- Discover Account Updater (not required by Discover for Worldpay acquired merchants)
Your Account Updater Welcome Kit includes the required application forms. If you have any questions about these forms, contact your Relationship Manager, who can walk you through the application process. Approval from Visa and MasterCard typically takes between 10-15 business days. Normally, merchants are approved without issue; however, you can be declined for a variety of reasons. For example, merchants on a risk mitigation program typically are not accepted.

**NOTE:** Visa does not allow merchants with SIC numbers 5962, 5966, 5967, or 7995 to participate in their Account Updater service. MasterCard has no restrictions against any specific MCC numbers.

### 1.6.2.3 Account Updater Features

The Account Updater service can include the following features depending upon the implementation option you select:

- Worldpay initiates requests for updated account information to card networks based upon your billing cycle.
- Worldpay initiates requests for updated account information following certain failed Authorization attempts.
- All updated card information stored (per merchant) in our secure database.
- Return of the updated account information in the cnpAPI response message when auto-repair occurs.
- Maintenance of card information history, so that the system can repair a card even if multiple updates have occurred during the card's billing lifecycle.
- All linked (to an Authorization) transactions will use the updated account information from the repaired parent transaction, including Captures, Refunds, and Reversals. If a re-Auth is needed on an attempted capture due to an expired authorization, the system uses the updated account information.
- Integration with Vault for merchants utilizing Worldpay’s tokenization solution.
- Return of Extended Response Codes in the cnpAPI response messages.
1.7 Recurring Engine

The Recurring Engine is a managed service that relieves the burden of developing an in-house billing solution for merchant engaged in installment or recurring transactions. This powerful, but flexible service allows you to create virtually any payment Plan required by your business model, whether it is part of a predetermined campaign or a marketing test, and then apply the Plan to customers as part of the standard Authorization or Sale transaction.

The Recurring Engine provides the following benefits:

- **Reduced Infrastructure Costs** - since you do not need to program your own solution, you save the up-front development cost, as well as ongoing maintenance expenses.

- **Reduced Labor** - once you create a Subscription in the Recurring Engine via a standard Authorization or Sale transaction, no further action is required for the life of the Subscription.

- **Integration with other Worldpay Value Added Services** - if you include the Recovery Services (Account Updater and Recycling Engine) as part of your implementation, you eliminate any concerns (and reduce expenses) associated with issues such as account number changes and recycling of declined payments.

- **Flexible Plans** - You can define the billing interval (weekly, monthly, quarterly, etc.), number of payments (including open ended schedules), amount of payments, as well as trial periods within a Plan. To add flexibility you can override several of these settings and set a specific start date at the individual Subscription level.

- **Flexible Creation of Discounts** - if you wish to offer a discount to selected customers, simply include the information at the time of the Subscription creation, or add it anytime afterward by updating the Subscription.

- **Flexible Creation of Add Ons** - similar to a discount, you can apply changes for additional services at the time of the Subscription, or anytime afterward.

- **Integrated Reporting** - in addition to the normal revenue reconciliation information available in the iQ Reporting and Analytics platform, there are a number of recurring specific reports that allow you to better analyze your revenue stream associated with recurring payment plans and strategies.

1.7.1 Payment Plans

The first step in setting up recurring billing on the Worldpay eCommerce platform is to establish one or more payment Plans. To establish a payment Plan you use a Create Plan transaction type, which allows you to define the payment interval, the number of payments, and the amount. For example, you could easily define any number of Plans to fulfill your business needs.

For example, suppose you are a SaaS company that sells your product under 1, 2, or 3 year deals, with either monthly or quarterly payment schedules and reduced rates for longer deals. You could easily set-up six Plans as shown in the table below.

<table>
<thead>
<tr>
<th>Plan Code</th>
<th>Payment Interval</th>
<th>Amount per Payment</th>
<th># of Payments</th>
<th>Total Subscription</th>
</tr>
</thead>
<tbody>
<tr>
<td>1_Year_Monthly</td>
<td>Monthly</td>
<td>$50.00</td>
<td>12</td>
<td>$600.00</td>
</tr>
</tbody>
</table>
Introduction

As part of the Plan, you can also specify trial period. You want to have longer trials for longer Plans, so for either 1-year Plan, there is a 1 week trial, for either 2-year Plan there is a 2 week trial, and for the 3-year Plans, a 1 month trial. Below is a cnpAPI example transaction to create the 3_YEAR_MONTHLY Plan.

Example: 3-Year Monthly Plan

```xml
<createPlan>
  <planCode>3_YEAR_MONTHLY</planCode>
  <name>3Year_Monthly</name>
  <description>3 Year, monthly Payments, 1 month trial</description>
  <intervalType>MONTHLY</intervalType>
  <amount>4166</amount>
  <numberOfPayments>36</numberOfPayments>
  <trialNumberOfIntervals>1</trialNumberOfIntervals>
  <trialIntervalType>MONTH</trialIntervalType>
  <active>true</active>
</createPlan>
```

### 1.7.2 Subscriptions

Subscriptions marry a customer order to a particular payment Plan and initiate the Recurring Engine to manage your future billing. You create a Subscription using either an Authorization or a Sale transaction. In the Auth/Sale you simply include a `<recurringRequest>` element to initialize the Subscription using a named Plan and set the start date for the first recurring bill. If you do not include a start date, the Recurring Engine uses the current date for the first payment.

If the recurring bill had an associated set-up or one-time fee use a Sale transaction. The amount of the Sale transaction would represent that fee, whereas the amount of future recurring payments are defined in the Plan. If you use an Authorization to create the Subscription, the transaction would normally be a $0 Auth (or small amount followed by a reversal) and would include the billing information for Address Verification.

As part of the Subscription creation, you can also override both the number of payments and the amount, as well as include Add Ons and Discounts (discussed in the next section). The overrides give you a granular control to modify a standard payments Plan for a particular consumer without creating additional

---

**TABLE 1-5  Example Plans**

<table>
<thead>
<tr>
<th>Plan Code</th>
<th>Payment Interval</th>
<th>Amount per Payment</th>
<th># of Payments</th>
<th>Total Subscription</th>
</tr>
</thead>
<tbody>
<tr>
<td>1_YEAR_QUARTERLY</td>
<td>Quarterly</td>
<td>$150.00</td>
<td>4</td>
<td>$600.00</td>
</tr>
<tr>
<td>2_YEAR_MONTHLY</td>
<td>Monthly</td>
<td>$46.66</td>
<td>24</td>
<td>$1119.84</td>
</tr>
<tr>
<td>2_YEAR_QUARTERLY</td>
<td>Quarterly</td>
<td>$140.00</td>
<td>8</td>
<td>$1120.00</td>
</tr>
<tr>
<td>3_YEAR_MONTHLY</td>
<td>Monthly</td>
<td>$41.66</td>
<td>36</td>
<td>$1499.76</td>
</tr>
<tr>
<td>3_YEAR_QUARTERLY</td>
<td>Quarterly</td>
<td>$125.00</td>
<td>12</td>
<td>$1500.00</td>
</tr>
</tbody>
</table>

As part of the Plan, you can also specify trial period. You want to have longer trials for longer Plans, so for either 1-year Plan, there is a 1 week trial, for either 2-year Plan there is a 2 week trial, and for the 3-year Plans, a 1 month trial. Below is a cnpAPI example transaction to create the 3_YEAR_MONTHLY Plan.
Plans. For example, if you offered a 1-year Plan with monthly payments as shown in the previous section, you could allow a consumer to complete their payments in 10 months instead of a year. In this case you would override the number of payments defined in the Plan (12) with 10 payments, while increasing the amount of each payment from $50 to $60.

**Example: Subscription with Overrides**

```xml
<authorization id="834262" reportGroup="ABC Division" customerId="038945">
  <orderId>65347567</orderId>
  <amount>0</amount>
  <orderSource>ecommerce</orderSource>
  <billToAddress>
    
    
    </billToAddress>
  <card>
    
    
    </card>
  <recurringRequest>
    <subscription>
      <planCode>1_Year_Monthly</planCode>
      <numberOfPayments>10</numberOfPayments>
      <startDate>2016-09-21</startDate>
      <amount>6000</amount>
    </subscription>
  </recurringRequest>
</authorization>
```
1.7.2.1 Add Ons and Discounts

Occasionally, you might wish to modify a Subscription with either a Discount or an Add On without creating a new Plan that has limited use. A Discount reduces the recurring amount for one or more payments, while an Add On increases the payments in return for an added service or item. You can apply either of these payment modifications at the time you initialize the Subscription or anytime afterward by updating the Subscription. In both cases you define the start date, end date, and amount of the Discount/Add On.

For example, suppose as part of your standard offering, your customers received 2GB of cloud-based storage. You also offer additional storage in 2GB blocks for $10 each. One of your customers wants an additional 4GB of storage for the 8 months remaining on his contract and you are discounting the first month at 50%, or $10. The example below show the updateSubscription transaction with the Add On and Discount.

Example: Update Subscription with Discount and Add On

```xml
<updateSubscription>
  <subscriptionId>1234</subscriptionId>
  <createDiscount>
    <discountCode>4GBExtraDeal</discountCode>
    <name>Half-Off 1st Payment 4GB Extra</name>
    <amount>1000</amount>
    <startDate>2016-09-15</startDate>
    <endDate>2016-10-14</endDate>
  </createDiscount>
  <createAddOn>
    <addOnCode>4GB_Extra</addOnCode>
    <name>Four_GB_Extra</name>
    <amount>2000</amount>
    <startDate>2016-09-15</startDate>
    <endDate>2016-04-15</endDate>
  </createAddOn>
</updateSubscription>
```

1.7.3 Recurring Reports

In addition to recurring transactions appearing in the normal Worldpay reports (i.e., Payment Detail, Reconciliation report, etc.), there is an additional report associated specifically with the Recurring Engine. Available via sFTP, this report is in Batch format and contains cnpAPI_saleResponse messages for all recurring transactions run that day, including all approvals and declines (see the example, Batch Sales Response with Recurring Info on page 298).
### 1.7.4 Transaction Types and Uses

The table below provides information about the various Recurring Engine transaction types and their uses.

<table>
<thead>
<tr>
<th>Use Case/Intent</th>
<th>Parent Element</th>
<th>Description/Uses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Plan</td>
<td>createPlan</td>
<td>Used to create new Plans.</td>
</tr>
<tr>
<td>Update Plan</td>
<td>updatePlan</td>
<td>Used to toggle a Plan between an active and inactive state. You can not associate an inactive Plan with a Subscription. Toggling a Plan to an inactive state does not affect existing Subscriptions associated with the Plan.</td>
</tr>
<tr>
<td>Create Subscription</td>
<td>&lt;authorization&gt; or &lt;sale&gt; &lt;recurringRequest&gt; &lt;subscription&gt;</td>
<td>Used to initiate a Subscription with an associated Plan. The Subscription can include amount and numberOfPayment overrides to the Plan. Also, the Subscription can include createAddOn and createDiscount children.</td>
</tr>
<tr>
<td>Update Subscription</td>
<td>updateSubscription</td>
<td>Used to modify Subscription information including: changing the Plan, changing the billing name/address, and changing the method of payment. You can also use this transaction type to create/delete/update Add Ons and Discounts.</td>
</tr>
<tr>
<td>Cancel Subscription</td>
<td>cancelSubscription</td>
<td>Used to cancel an existing Subscription.</td>
</tr>
<tr>
<td>Create Add On</td>
<td>&lt;authorization&gt; or &lt;sale&gt; &lt;recurringRequest&gt; &lt;subscription&gt; &lt;createAddOn&gt; Or &lt;updateSubscription&gt; &lt;createAddOn&gt;</td>
<td>Used to create an Add On charge associated with the Subscription.</td>
</tr>
<tr>
<td>Update Add On</td>
<td>&lt;updateSubscription&gt; &lt;updateAddOn&gt;</td>
<td>Used to modify one or more of the parameters associated with the Add On.</td>
</tr>
<tr>
<td>Delete Add On</td>
<td>&lt;updateSubscription&gt; &lt;deleteAddOn&gt;</td>
<td>Used to delete an Add On charge from the associated Subscription.</td>
</tr>
</tbody>
</table>
## Recurring Engine Transaction Types

<table>
<thead>
<tr>
<th>Use Case/Intent</th>
<th>Parent Element</th>
<th>Description/Uses</th>
</tr>
</thead>
</table>
| Create Discount  | `<authorization>` or `<sale>`  
|                  | `<recurringRequest>`  
|                  | `<subscription>`  
|                  | `<createDiscount>`  
| Or               | `<updateSubscription>`  
|                  | `<createDiscount>` | Used to create a Discount charge associated with the Subscription. |
| Update Discount  | `<updateSubscription>`  
|                  | `<updateDiscount>` | Used to modify one or more of the parameters associated with the Discount. |
| Delete Discount  | `<updateSubscription>`  
|                  | `<deleteDiscount>` | Used to delete a Discount from the associated Subscription. |
1.8 Issuer Insights

Issuer Insights provides you additional card/transaction, performance descriptive data, allowing you to understand the differences among Issuers and card products, and identify trends that may impact future transactions. Currently, Worldpay bundles four feature sets into Issuer Insights: the Issuer Insights Secure Scheduled Report, Real Time Indicators (i.e., Prepaid Indicator, Affluence Indicator, Issuer Country Indicator, Card Type Indicator), Network Response Data, and the Insights Analytics Dashboard.

1.8.1 Issuer Insights Secure Scheduled Report

Every merchant wants to maximize their approval rate and with it their customer conversion rate and sales. Unfortunately, issuing banks do not all use the same criteria and/or algorithms when approving or declining transactions. Getting an approval on an authorization is not a one size fits all proposition.

The weekly Issuer Insights report provides a range of data points that breakdown authorization approvals and declines by card type, BIN (i.e. Issuing Bank), and account ranges (indicating card product types). It also provides data about the four of the most often seen reasons for declines (i.e., Non-sufficient Funds, Do Not Honor, Invalid Account, and Lost/Stolen Card), as well as benchmark data for your MCC(s) across the Worldpay eComm portfolio, and Issuer participation in AU, as well as account update counts per account range across our portfolio. Using this data, you can create a wide range of useful analytics, such as approval/decline trending by BIN, account updates trending by account range, new card issuance/portfolio changes, or customer lifetime value by card type to name a few.

For additional information about this report please refer to the Worldpay eComm Scheduled Secure Reports Reference Guide.

1.8.2 Real Time Indicators

The Real Time Indicators provide six data points in the Authorization/Sale transaction response message. This information includes the Prepaid Indicator, Affluence Indicator, Issuer Country Indicator, Card Type Indicator, Funding Source Indicator, and Virtual Account Number Indicator. Each of these pieces of information allows you to make a real time decision about how to proceed with the transaction. The sections that follow explain each indicator along with possible uses in your decision making process.

1.8.2.1 Prepaid Indicator

Studies show that branded prepaid cards are growing in popularity with consumers. These cards are available in the form of non-reloadable Gift cards, Consumer Rebate/Incentive cards, and Teen cards among others. The Prepaid Indicator feature acts to determine if the submitted card is a prepaid card. If so, the system returns the type element with a value of Prepaid and the availableBalance element stating the outstanding balance remaining on the card (if available).

Knowing that the card is prepaid at the time of sale, as well as if it is reloadable and the available balance, is especially useful for merchants engaged in recurring payment, installment payment, or deferred billing scenarios. Merchants in these situations can use the information made available by this feature to make intelligent decisions concerning the profitable management of prepaid card usage by avoiding several factors that may contribute to lost revenue, while taking advantage of other opportunities that may add to revenue and enhance the customer experience.
For example, one possible situation merchant can avoid is fraudulent deferred/installment payment purchases made with a prepaid card that does not have enough available balance to cover the subsequent payments. With the available balance known, merchants can determine if the card can meet the required payment structure. If the card’s balance does not meet the required threshold, the merchant can request another payment method, which may result in eliminating fraudsters, while retaining legitimate customers. If the card is non-reloadable, it is advisable to not accept the card at all for recurring payments regardless of the available balance.

Another more common situation occurs when the consumer is unaware of the card balance. If the transaction is rejected due to inadequate balance, perhaps repeatedly, it could result in an unsatisfied customer and an abandoned purchase. Alternately, the card could have slightly more that the required balance, which the consumer would spend, if they had the knowledge. If the available balance is insufficient for the purchase, the merchant can obtain a second or alternate payment method. If the balance is higher than required for the purchase, the merchant may be able to encourage additional purchases.

In addition to indicating if the submitted card is a prepaid card and the available balance, this feature includes information about whether the card is reloadable and the specific type of prepaid card (e.g., TEEN, GIFT, PAYROLL, etc.). You can use this information to further refine your sales and marketing strategies.

1.8.2.2 Affluence Indicator

Visa, MasterCard, and Discover provide enhanced credit card products for consumers with high disposable incomes and high card spending. These cards encourage usage by offering the cardholders additional benefits usually including reward incentives, no pre-set spending limit, higher authorization approval rates, faster access to a customer service representatives, and dedicated chargeback resolution support.

Worldpay analysis of payments data indicates that consumers using these cards types typically spend more per order than consumers using traditional credit and debit cards. The Affluence Indicator feature provides the ability for merchants to segment their consumers based on the affluence level as determined by the issuer. Within the cnpAPI Authorization response, consumers using these enhanced card products are classified either as Mass Affluent or Affluent. Based upon the specific card type, high income consumers are classified as Mass Affluent, while high income-high spending consumers are classified as Affluent.

Having this information at the time of authorization, allows merchants the opportunity to adjust their sales approach to the needs and spending patterns of the consumer, potentially generating additional sales. Having this information on file for later analysis also may provide the opportunity for targeted marketing campaigns and future sales.

1.8.2.3 Issuer Country Indicator

Knowing the country of the Issuing bank helps you in two respects. From a sales and marketing standpoint, this knowledge allows you to better analyze the purchasing patterns of your customers based upon their country of origin. Knowing these customer demographics can allow you to tailor marketing campaigns to take advantage of this geographic information. Likewise, you can use this information to analyze the successfulness of tailored campaigns.

The second advantage to having this information readily available is that you can use it to help determine possible patterns of fraud. With this knowledge in hand, you can use the International Card Filtering feature to limit your exposure to international fraud originating in particular geographic locations.
1.8.2.4 Cardholder Type Indicator

The Card Holder Type indicator is an additional data point Worldpay provides as part of Issuer Insights. This indicator returns an element specifying whether the submitted card is a commercial or consumer card, providing you with additional data useful when analyzing sales patterns and/or planning marketing campaigns, such as offering different products/services to a corporate customer as opposed to a regular consumer.

1.8.3 Extended Network Data

In an effort to provide more data for use in analyzing your transactions and customers, Worldpay added a number of data elements originating directly from the ISO 8583 network response messages. If you use V11.0 or above of the Worldpay cnpXML, you can receive up to 17 data elements as part of the Enhanced Auth Response (<enhancedAuthResponse> element). These elements provide more granular data, allowing you to gain a more nuanced understanding of the approval/decline behavior of particular Issuers. Table 1-7 provides information about the fields returned.

**Table 1-7 Possible Extended Network Data ISO 8583 Fields**

<table>
<thead>
<tr>
<th>Data Element</th>
<th>Field Name</th>
<th>Type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Transaction Amount</td>
<td>n 12</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Settlement Date</td>
<td>n 4</td>
<td>MMDD format</td>
</tr>
<tr>
<td>38</td>
<td>Authorization Identification Response</td>
<td>an 6</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>Response Code</td>
<td>an 2</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>Additional Response Data</td>
<td>an..25</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>Private Use Additional Data</td>
<td>an...999</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>Settlement Currency Code</td>
<td>n 3</td>
<td>Defines the currency of the settlement</td>
</tr>
<tr>
<td>51</td>
<td>Cardholder Billing Currency Code</td>
<td>n 3</td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>Additional Amounts</td>
<td>an...120</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>Reserved Private</td>
<td>an...999</td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>Transaction Description</td>
<td>an...100</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>Reserved for National Use</td>
<td>an...999</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>Reserved for Private Use</td>
<td>an...999</td>
<td></td>
</tr>
</tbody>
</table>
**Example: Extended Network Data**

```xml
<enhancedAuthResponse>
  
  
  
  <networkResponse>
    <endpoint>VISA</endpoint>
    <networkField fieldNumber="4" fieldName="Transaction Amount">
      <fieldValue>000000000962</fieldValue>
    </networkField>
    <networkField fieldNumber="44" fieldName="Additional Response Data">
      <networkSubField fieldNumber="0">
        <fieldValue></fieldValue>
      </networkSubField>
      <networkSubField fieldNumber="1">
        <fieldValue>5</fieldValue>
      </networkSubField>
    </networkField>
  </networkResponse>
</enhancedAuthResponse>
```

### 1.8.4 Insights Analytics Dashboard

The Insights Dashboard provides insight into the quality of customers your organization acquires, based on the number and percentage of authorizations by card type or other indicator, and contrasts that with the number of customers whose authorizations are successfully converted to deposits. There are four categories of indicators: Affluence, Prepaid, Corporate Customers, and Country of Issuing Bank.

Using these values and the related charts, your organization can decide whether the customer mix you attract yields maximum revenue potential, or if a change in marketing is required. By using the adjustable date range and reporting groups, you can compare the quality of customers between different business units and affiliates, or monitor the response to marketing campaigns.

Please refer to the *Worldpay eComm iQ Reporting and Analytics User Guide*. 
1.9 Fraud Toolkit

Just because a credit card network/company returns a valid authorization for a purchase does not always mean that completing the transaction is in your best interest. There are multiple reasons you may wish to decline a sale on a particular card at a particular time. In many cases there are indicators that the transaction could be or likely is fraudulent. Acting to stop these transactions at submission prevents loss, as well as reducing the number fraud related chargebacks in the future. Worldpay offers a robust fraud solution, the Fraud Toolkit, to assist you in reducing the number of possibly fraudulent transactions inflicted upon you by bad actors.

The Fraud Toolkit has three tiers or levels of implementation, each providing more rigorous examination of transaction properties and data points, as well as valuable information and guidance. The table below provide an overview of the tool provided at each level. The items highlighted in blue require the inclusion of a small snippet of code on your checkout page.

**TABLE 1-8  Fraud Toolkit Implementation Levels**

<table>
<thead>
<tr>
<th>Filter/Feature</th>
<th>Essential</th>
<th>Extended</th>
<th>Premium</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVS Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>CVV No-Match Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>International BIN Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Prepaid Non-Reloadable Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Prior Chargeback Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Prior Fraud Advice Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Card Velocity Filter (Approved or Attempted)</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Email Velocity Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Phone Velocity Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IP Velocity Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Device Velocity Filter</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>IP Address, Geolocation, and Proxy Detection</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Merchant Customizable Rules Template</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ThreatMetrix Cybercrime Dashboard</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>(Asynch) Transaction Review Queues</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Rule Management and Portal Training</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Standalone Transaction API Access</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Cybercrime Industry Report (Quarterly)</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Access to Fraud Consultant</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
1.9.1 Essential Tier

The Essential tier includes a suite of eleven Fraud Filters that you can apply individually or in combination. Nine of the eleven filters, based on card/submitted data, potentially require no additional integration on your part, assuming you already submit the necessary information. The remaining two filters require you to add a a small snippet of code on your checkout page.

NOTE: Technically, you can make use of the IP Velocity filter without integrating the code snippet on your checkout page. Instead you can simply include the originating IP Address that you detect in your transaction by submitting it as a customAttribute. Please note that this method will likely be less effective than making use of the ThreatMetrix functionality, which includes IP piercing to determine the true IP of the consumer’s device.

1.9.1.1 Prepaid Card Filtering

Many merchants engaged in recurring payment, installment payment, or deferred billing experience some loss due to fraud schemes that make use of prepaid cards. Consider the case of a consumer using a prepaid card with a balance of $100 to make a purchase that involves an initial charge of $50 followed by three installments of $50 each. The authorization would be approved for the initial transaction, and the card might have adequate balance for an additional charge, but if the consumer was attempting to defraud the merchant or simply used the card for other purchases, the card may not have sufficient balance for any additional payments. While the Prepaid Indicator feature provides you with the information necessary to make a decision at the time of the sale, and to request a secondary or different payment method, instead you may wish to have Worldpay filter these transactions automatically when you send the Authorization transaction.

If you elect to use the Prepaid Card Filtering Service, you can select one of two methods of implementation. Using the first filtering method, our system declines all Authorization and Sale transactions when the consumer uses a prepaid card. Upon a decline, the system returns a Response Reason Code of 309 - Restricted Card - Prepaid Card Filtering Service. This method also allows you to disable the filtering logic on a transactional basis by including the <prepaid> element set to a value of false, allowing you to accept any prepaid card for these transactions.

The second method of implementing the Prepaid Card Filtering Service is per transaction. To enable the filter on a particular transaction, set the <prepaid> element to a value of true. This method is useful to a merchant who offers products with both one-time payments and installment payments. For products involving a single payment, you may want to allow the use of prepaid cards, while for the product with multiple payments you may want to filter prepaid cards.

NOTE: Within either implementation method, you can elect to filter all prepaid cards, or only non-reloadable prepaid cards. Please speak to your Implementation Consultant for additional information about setting these global parameters.

1.9.1.2 International BIN Filtering

An examination of your historical fraud data may show a high percentage of fraudulent transactions originating with certain international cards. You can limit your exposure to this type of fraud by taking advantage of the International Card Filtering Service. This feature allows you to filter MasterCard and Visa cards originating in either all foreign countries or selected foreign countries based upon the country of the card issuer.
If you elect to use this feature, when you submit an Authorization/Sale transaction, the system determines the country of origin of the card. If the card originates outside the United States and you have elected to filter all international cards, the system declines the transaction. Likewise, if you have elected to filter a specific country or countries and the card originates from a designated country, the system declines the transaction. Upon a decline, the system returns a Response Reason Code of **312 - Restricted Card - International Card Filtering Service**.

You can override your settings on a transactional basis by including the `<international>` element set to `false` when you submit the Authorization/Sale transaction. In this case, the system ignores the filtering service and processes the transaction normally.

### 1.9.1.3 Prior Chargeback Filtering

If you elect to use the Chargeback Filter Service, there are two configuration options. You can elect to filter all transactions using a card for which you received a chargeback, or you can elect to filter only the subset of transactions for which you received a fraud related chargeback (determine by the associated chargeback reason code). In both cases, the system checks your historical data to see if you received an applicable chargeback from the same account within the last 90 days. Upon a decline, the system returns a Response Reason Code of **308 - Restricted Card - Chargeback**.

### 1.9.1.4 Security Code No-Match Filter

The Card brands added the 3- or 4-digit security code to act as a verification that the person ordering your product in a card-not-present environment has physical possession of the card. While this validation can be a useful anti-fraud tool, typically, many issuing banks do not decline the transaction based upon a failure to match the security code. Declining the transaction is left to the discretion of the merchant.

**NOTE:** Since American Express declines the transaction when the security code does not match, the Security Code No-Match filter does not apply to American Express transactions. Transactions declined by American Express for a failure to match the security code use the Response Reason Code of **352 - Decline CVV2/CID Fail**.

Similarly, if Visa, MasterCard, or Discover decline a transaction based upon the security code results, Worldpay does not apply the filter and the transaction response contains the 352 Reason Code.

If you elect to use the Security Code No-Match Filter Service, the system takes action only if the issuer approves the submitted authorization/sale transaction, but includes a no-match code for the CVV2/CVC2/CID card validation check. In this case, the Worldpay declines the transaction with a Response Reason Code of **358 - Restricted by Vantiv due to security code mismatch**. The system also issues an Auth Reversal transaction on your behalf to remove the funds hold on the account.

### 1.9.1.5 Card Velocity Filtering

Often, when a person attempts to use a stolen credit card successfully, they will follow the initial purchase with a number of additional purchases within a short period of time. If you elect to use the Fraud Velocity Filter, the system filters the transaction based upon the number of previously approved Auth/Sale transactions plus the number of Auth/Sale transactions declined by another filter, for the same account.
within a designated time period. Both the total number of transactions and the time period are configured in the Worldpay eComm Merchant Profile.

Upon a decline, the system returns a Response Reason Code of **315 - Restricted Card - Auth Fraud Velocity Filtering Service**.

1.9.1.6   **Prior Fraud Advice Filtering**

Worldpay maintains a database of Fraud Advice information received from the Visa and MasterCard networks for transactions you processed in the last 200 days. If you use the Prior Fraud Advice Filter, the system compares the account information from the new transaction against the database of accounts with prior Fraud Advice and filters the transaction if there is a match.

Upon a decline, the system returns a Response Reason Code of **318 - Restricted Card - Auth Fraud Advice Filtering Service**.

1.9.1.7   **AVS Filter**

One of the fraud prevention tools provided by all card networks is an Address Verification System. By submitting the customer’s address information in the billToAddress section of the cnpAPI message, you can verify that the address/zip code supplied by the consumer matches the issuer’s records. The card networks, however, do not decline transactions based upon the failure to match the address or zip code. Using the AVS Filter, you can filter potentially fraudulent transactions based upon failure to match any of the following:

- the address
- the zip/postal code
- the address + zip/postal code (ANDed)
- the address or zip/postal code (ORed).

Upon a decline, the system returns a Response Reason Code of **319 - Restricted Card - Fraud AVS Filtering Service**.

1.9.1.8   **Email Velocity Filter**

Often, card testers or other bad actors submit a number of transaction using multiple cards, but with a common email address. The only requirement to make use of this filter is that you collect and include the consumer's email address with each transaction. We communicate the email address to our fraud partner, ThreatMetrix, who tracks and analyzes the information. If the filter detects the same email used in the configured number of transactions within the configured period of time, the system declines new transactions (using the same email) on your behalf and returns Response Code **550 - Restricted Device or IP - ThreatMetrix Fraud Score Below Threshold**.

1.9.1.9   **Phone Velocity Filter**

Similar to email, card testers or other bad actors often submit a number of transaction using multiple cards, but with a common phone number. The only requirement to make use of this filter is that you collect and include the consumer’s phone number with each transaction. We communicate the phone number to our fraud partner, ThreatMetrix, who tracks and analyzes the information. If the filter detects
the same phone number used in the configured number of transactions within the configured period of time, the system declines new transactions (using the same email) on your behalf and returns Response Code 550 - Restricted Device or IP - ThreatMetrix Fraud Score Below Threshold.

### 1.9.1.10 IP Velocity Filter

The IP Velocity filter is one of the two filters in the Essential tier that requires (see note below) the addition of a code snippet to your checkout page. This snippet, which you also need to implement for the higher tiers of Fraud Toolkit, allows our partner, ThreatMetrix, to perform IP interrogation/piercing to determine the true IP Address of the device originating the order. As with the other velocity filters, if the filter detects the same IP Address used in the configured number of transactions within the configured period of time, the system declines new transactions from the same IP Address on your behalf and returns Response Code 550 - Restricted Device or IP - ThreatMetrix Fraud Score Below Threshold.

**NOTE:** Technically, you can make use of the IP Velocity filter without integrating the code snippet on your checkout page. Instead you can simply include the originating IP Address that you detect in your transaction. Please note that this method will likely be less effective than making use of the ThreatMetrix functionality, which includes IP piercing to determine the true IP of the consumer’s device.

### 1.9.1.11 Device Velocity Filter

The Device Velocity filter is the second Essential tier filter in the that requires the addition of a code snippet to your checkout page. In this case, the snippet allows ThreatMetrix to construct a device fingerprint of the system originating the order. As with the other velocity filters, if the filter detects the same device used in the configured number of transactions within the configured period of time, the system declines new transactions from the same device on your behalf and returns Response Code 550 - Restricted Device or IP - ThreatMetrix Fraud Score Below Threshold.

### 1.9.1.12 Application of Filters - Filtering Rules

**NOTE:** We define your Filter Rules as part of your Merchant Profile. Please consult with your Relationship Manager and/or your Implementation Consultant concerning the provisioning of Filter Rules.

While you can have all submitted transactions flow through the Fraud toolkit, you likely want to exercise a finer control over the application of the filters based upon a particular product, service or other criteria. The system provides you the flexibility of restricting which transactions are submitted to the filtering service and which filters the system applies to which groups. This is accomplished by defining Filtering Rules.

For each Filtering Rule you first define a subgroup of transactions by selecting one of the following Flow Selectors: Report Group, Billing Descriptor, orderSource, or MID (for Payment Facilitators, flow control by MID or order source only). Only one selector can be applied per rule. After selecting a particular Flow Selector, you then select which filters to have applied to that subset of transactions. You can define the Filter Rules so that filters are ORed (transaction filtered when any one of the filters conditions met), or ANDed (transaction filtered when multiple filter conditions met). Table 1-9 defines five rules that a merchant might define.
Table 1-9 defines five Filter Rules that a merchant might use. These rules would be applied as follows:

- Filters 1 and 2 apply to the subset of transactions that are members of Report Group XYZ and use the Prepaid and International Filters. Since the Filter Rules are defined separately, the rules are ORed. So, if a transaction uses either a Prepaid card or a card of International origin, the filtering occurs.

- Filter 3 applies to the subset of transactions that have an orderSource value set to recurring. Filtering occurs only if both the criteria for the Prepaid Filter AND the Prior Chargeback Filter are met.

- Filter 4 applies to the subset of transactions that have an orderSource value set to ecommerce. Filtering occurs only if both the criteria for the Card Velocity Filter AND the Security Code No-Match Filter are met.

- Filter 5 applies to the subset of transactions that have an Billing Descriptor value set to GoldMember. Filtering occurs only if both the criteria for the Prepaid Filter AND the International Filter are met.

### 1.9.2 Extended Tier

The Extended Tier include all of the Essential Tier filters, but offers an additional levels of fraud detection made available through Worldpay’s partnership with ThreatMetrix. The addition of the same code snippet used for the IP and Device Velocity filters to your checkout page allows ThreatMetrix to gather additional data points, such as the consumer's device, proxy use, and location. Unlike the filters in the Essential Tier, which are basic accept/decline filters, the Extended Tier takes the data and compares the information to a rule list. Worldpay supplies an initial, Best Practices rules list designed for your business type (i.e., Retail, Digital, Non-profit, etc.), which you can modify and refine for you particular business model. Each rule, when triggered, add or subtract a preset value from the transaction score. If the score fall below a set threshold, the system declines the transaction, unless you prefer to make the final decision yourself. In either case, Worldpay returns the score and a list of triggered rules in the transaction response message.

In addition to the ThreatMetrix rules engine, you get access to the ThreatMetrix Portal allowing you to customize your rules list and scoring values. This level also allows you to white list/black list items, such as email addresses and phone numbers. Other items included in this tier are:

- ThreatMetrix Cybercrime Dashboard
- Asynchronous Transaction Review Queues
- Monthly Rules and Portal training
- API Access to Standalone Fraudcheck transaction.

---

**Table 1-9** Example - Fraud Filtering Service Rules

<table>
<thead>
<tr>
<th>Filter</th>
<th>Flow Selector</th>
<th>Filters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Report Group = &quot;XYZ&quot;</td>
<td>Prepaid</td>
</tr>
<tr>
<td>2</td>
<td>Report Group = &quot;XYZ&quot;</td>
<td>International</td>
</tr>
<tr>
<td>3</td>
<td>orderSource = &quot;recurring&quot;</td>
<td>Prepaid + Prior Chargeback</td>
</tr>
<tr>
<td>4</td>
<td>orderSource = &quot;ecommerce&quot;</td>
<td>Card Velocity + Security Code No-match</td>
</tr>
<tr>
<td>5</td>
<td>Billing Descriptor = &quot;GoldMember&quot;</td>
<td>Prepaid + International</td>
</tr>
</tbody>
</table>
1.9.3 Premium Tier

The Premium Tier provides all of the tools from the Essential and Extended Tiers, and most importantly, access to the Worldpay eComm Fraud Consulting service. With the service, the Fraud Consultant assigned to you helps analyze your transactional data, recommend rule changes to fine-tune your results, and advises you on fraud detection strategy.

1.9.4 Modifications to Your Web Page

For ThreatMetrix to gather information for analysis, you must add certain profiling tags (see example below) to selected pages served by your web application. These tags allow ThreatMetrix to collect information by loading objects used for detection into the consumer’s browser. These tags are invisible to the consumer and add only a fraction of a second to your page’s rendering time. Once loaded, these objects require only 3-5 seconds to gather profiling information from the consumer device.

Place the tags as early as possible on the page, inside the `<body></body>` tags of the page HTML.

**Example: ThreatMetrix Profiling Tags**

```html
<!-- Begin ThreatMetrix profiling tags below -->
<script type="text/javascript"
src="https://h.online-metrix.net/fp/tags.js?org_id=ORG_ID&session_id=UNIQUE_SESSION_ID&pageid=PAGE_ID"></script>
<noscript>
<iframe style="width: 100px; height: 100px; border: 0; position: absolute; top: -5000px;"
src="https://h.online-metrix.net/tags?org_id=ORG_ID&session_id=UNIQUE_SESSION_ID&pageid=PAGE_ID"></iframe>
</noscript>
<!-- End profiling tags -->
```

**NOTE:** Replace UNIQUE_SESSION_ID with a uniquely generated handle that includes the Worldpay supplied prefix.

The value for ORG-ID is a Worldpay supplied value.

The pageid tag is not used at this time. The value for PAGE-ID defaults to 1.

For production, replace h.online-metrix.net with a local URL and configure your web server to redirect to h.online-metrix.net.

1.9.4.1 cnpAPI Transactions

To subject a transaction to the advanced fraud checks performed by ThreatMetrix and retrieve the results, you simply submit the `<threatMetrixSessionId>` element as part of your cnpAPI Authorization (or Sale) transaction. This session Id is the same unique value you assigned and sent to ThreatMetrix when your web page called the application (designated as UNIQUE_SESSION_ID in the ThreatMetrix Profiling Tags example). When we receive an Authorization/Sale that includes the `<threatMetrixSessionId>`, our system automatically queries the ThreatMetrix platform for the associated results. The cnpAPI
response message includes the `<advancedFraudResults>` element containing the score and status and information about any triggered rules. The following two examples show a standard Authorization transaction, including a `<threatMetrixSessionId>` and a pass response.

**Example: Authorization including `<threatMetrixSessionId>` Element**

```xml
<cnpOnlineRequest version="12.6" xmlns="http://www.vnativcnp.com/schema"
merchantId="81601">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <authorization id="002" reportGroup="001601">
    <orderId>10102013_sessionId_app</orderId>
    <amount>1002</amount>
    <orderSource>ecommerce</orderSource>
    <billToAddress>
      <name>John Doe</name>
      <addressLine1>15 Main Street</addressLine1>
      <city>San Jose</city>
      <state>CA</state>
      <zip>95032-1234</zip>
      <country>USA</country>
      <phone>9782750000</phone>
      <email>nobody@vantiv.com</email>
    </billToAddress>
    <card>
      <type>MC</type>
      <number>5405102001000003</number>
      <expDate>1115</expDate>
    </card>
    <advancedFraudChecks>
      <webSessionId>ASDFG-AXXXXAB999</webSessionId>
    </advancedFraudChecks>
  </authorization>
</cnpOnlineRequest>
```

**Example: Authorization Response including `<advancedFraudResults>` Element**

```xml
<cnpOnlineResponse version="12.6" xmlns="http://www.vantivcnp.com/schema"
response="0" message="Valid Format">
  <authorizationResponse id="002" reportGroup="001601">
    <cnpTxnId>82823534116454639</cnpTxnId>
    <orderId>10102013_sessionId_app</orderId>
    <response>000</response>
  </authorizationResponse>
</cnpOnlineResponse>
```
1.9.4.2 Information Only Option

If you wish to retain full control of the decision to accept or decline transactions, Worldpay offers the option of using the Advanced Fraud Tools in an Information Only mode. In this configuration, you receive the same information in the response as you would with the full implementation; however, Worldpay will not automatically decline transactions with a failing score.

If the authorization is declined by the network, you can choose to recycle the transaction or do nothing. If an authorization with a failing score receives approval from the network, it would be up to you to reverse the authorization should you decide not to proceed with the transaction. This is similar to the case of an approved transaction that has a status of Review, but you decide not to proceed. Issuing an authorization reversal allows you to avoid any misuse of Auth fees otherwise imposed by the card networks.

1.9.4.3 Fraud Check Transactions

If you wish to retrieve the Advanced Fraud results without introducing a Authorization or Sale transactions, use a Fraud Check transaction (see Fraud Check Transaction on page 268). Fraud Check transactions are only supported as Online transactions.
1.10 Tokenization Feature

Tokenization is the process by which a reference number, referred to as a token, replaces a credit card number or Direct Debit account number. Unlike the card or account number, you can store the token on your system without concern of a security breach exposing critical customer information. Instead, Worldpay stores the information in a secure vault and accesses it only when you submit a transaction using the supplied token.

**NOTE:** To use the Direct Debit tokenization feature, you must also use card tokenization.

In the case of credit cards, since you do not store the customer’s account information, the scope of PCI requirements to which you must comply may be minimized. This may greatly reduce the cost of compliance and may limit your liability in the event of a breach. You can reduce the requirements further, as well as the possibility of exposure from a breach, through the use of the Worldpay eProtect. By sending the card information from your checkout page directly to our systems you eliminate one more facet of handling the card information.

**NOTE:** Worldpay recommends you consult your own PCI Compliance and Legal departments to determine the specific advantages of tokenization for your company.

This section discusses the following topics:

- **How Tokenization Works**
- **Token Formats**
- **Obtaining Tokens**
- **Supported Token Transactions**

**NOTE:** Please refer to the *Worldpay eProtect Integration Guide for additional* information about the use of and integration to the Worldpay eProtect value added service.
1.10.1 How Tokenization Works

In a non-tokenized environment, multiple parties handle and store customer data, including the card/Direct Debit account number, for each transaction. From a merchant standpoint, they receive the information, store it in their own database, and transmit it to their processor with the transaction request, as shown in Figure 1-5 for card information. While the access and transmission of the data may occur a single time, as in the case of a Sale transaction, frequently data transmission occurs multiple times in order to complete a single sale, as in the case of an Auth followed by a Capture or several partial Captures. The local storage and repeated transmission of the information creates additional possible breach points, where a malicious third party could compromise the information.

**FIGURE 1-5** Card Information Flow in Non-Token Environment

In a tokenized environment transmission of customer data ideally occurs a single time and the merchant never stores it locally, as shown in Figure 1-6 for card data. Once account number registration occurs, using either a registerTokenRequest or by submitting the account number (or low value token, when using eProtect) with any supported transaction, Worldpay returns a (high value) token. You store the token locally and use it for all future transactions concerning that account. Worldpay takes responsibility for storing and safeguarding the account information.

**NOTE:** The difference between card data flow and Direct Debit data flow is that the entities upstream of Worldpay are different. The operation remains the same from a merchant standpoint.

**FIGURE 1-6** Card Information Flow in Tokenized Environment

**NOTE:** The use of eProtect allow the account information to come directly to Worldpay, so the merchant handles the token only.
1.10.2 Token Formats

Worldpay supports both High-Value and Low-Value tokens, designed to address specific payment processing operations. Both token types replace sensitive data, though the generated token values differ based on specific domain and transaction controls.

We generate High-Value tokens statically, replacing PANs with a consistent surrogate value each time you present the same PAN for tokenization. PCI DSS Tokenization Guidelines 2.0 (2011) defines a High-Value as a token usable as a payment instrument to perform a payment transaction. Worldpay OmniToken is an example of a High-Value token. The generation of a Worldpay OmniToken is card-based, embodying a one-to-one correlation with the PAN it replaces, and does not expire unless the PAN is no longer valid with the card issuer.

PAN to OmniToken pairing is unique to a given organization, and is channel and platform agnostic. This means the same token value persists across the entire merchant payment processing environment, but does not persist across different merchant organizations. For example, Table 1-10 shows the same PAN submitted for three different transactions. The first two transactions involve the same merchant with one transaction from a Point of Sale terminal and one Online. Since these transactions use the same PAN and come from the same merchant, they use a common Worldpay OmniToken. The third transaction, from a different merchant results in the generation of a different OmniToken even though it is the same PAN.

<table>
<thead>
<tr>
<th>Merchant</th>
<th>PAN</th>
<th>Worldpay OmniToken</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merchant A - POS</td>
<td>4418 3789 1620 3675</td>
<td>4418 3711 2222 3675</td>
</tr>
<tr>
<td>Merchant A - Online</td>
<td>4418 3789 1620 3675</td>
<td>4418 3711 2222 3675</td>
</tr>
<tr>
<td>Merchant B - Online</td>
<td>4418 3789 1620 3675</td>
<td>4418 3724 9587 3675</td>
</tr>
</tbody>
</table>

NOTE: The OmniToken shown in the example above make use of the Informative token format, which preserves the first-6 and last-4 digits.

Generated dynamically, Low-Value tokens have a unique value for each transaction, even if you submit the same PAN for tokenization multiple times. Low-value tokens utilize additional compensating controls, such that the token applies only to specific transactions and processing environments. The Worldpay eProtect RegistrationID is an example of a Low-Value token used for initial data capture of payment card data in ecommerce and mobile environments. Worldpay eProtect RegistrationID is a transaction-based token value requested through a low-trust environment over the Internet. The token value is valid only for 24-hours, after which you can no longer use it for payment processing.

There are various attributes, including the length, format and character set, that define the structure of the card numbers and the tokens that replaces them. For example, all PAN values regardless of length, conform to the Luhn MOD10 algorithm. Worldpay supports several token format options to meet your implementation needs. These token formats, while always using the same character set as PANs, range from a completely random, fixed-length number to a format preserving token that reuses the first-6 and last-four digits of the PAN. In addition, with the exception of the legacy eComm token version, each token format is available in either a MOD10 or a MOD10+1 format. The MOD10+1 format allows you to more easily distinguish a real PAN from a token.
The table below provides information about the various OmniToken formats available for credit cards. Since you cannot intermix token formats within a Token Group and your entire organization may be a single Token Group, you should carefully select the format that best meets your needs. When selecting a format, you should consider if you need access to the IIN (Issuer Identification Number; formerly known as BIN), as well as whether you need the last-four digits of the PAN.

**TABLE 1-11** High-Value Credit Card Tokens

<table>
<thead>
<tr>
<th>Name</th>
<th>Format-Preserved Digits</th>
<th>MOD10 Option</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First-Six (IIN)</td>
<td>Middle</td>
</tr>
<tr>
<td>Informative A1</td>
<td>6 Random 4</td>
<td>MOD10+1</td>
</tr>
<tr>
<td>Informative A</td>
<td>6 Random 4</td>
<td>MOD10</td>
</tr>
<tr>
<td>Last 4 Preserved 1</td>
<td>Random Random 4</td>
<td>MOD10+1</td>
</tr>
<tr>
<td>Last 4 Preserved</td>
<td>Random Random 4</td>
<td>MOD10</td>
</tr>
<tr>
<td>Complete Random 1</td>
<td>Random Random Random</td>
<td>MOD10+1</td>
</tr>
<tr>
<td>Complete Random</td>
<td>Random Random Random</td>
<td>MOD10</td>
</tr>
<tr>
<td>Minimum 16A 1</td>
<td>6 Random 4</td>
<td>MOD10+1</td>
</tr>
<tr>
<td>Minimum 16A</td>
<td>6 Random 4</td>
<td>MOD10</td>
</tr>
<tr>
<td>Existing Worldpay eComm Merchants 1</td>
<td>Random* Random Random 4</td>
<td>MOD10+1</td>
</tr>
</tbody>
</table>

* First-six returned in the <bin> element of the response message

**TABLE 1-12** Low-Value Tokens

<table>
<thead>
<tr>
<th>Format Code</th>
<th>Name</th>
<th>Format-Preserved Digits</th>
<th>MOD10 Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>RegistrationID</td>
<td>None</td>
<td>MOD10+1</td>
</tr>
<tr>
<td>N/A</td>
<td>Checkout Id</td>
<td>None</td>
<td>MOD10+1</td>
</tr>
</tbody>
</table>

**1.10.2.1 High Value Token Data Format Summary**

This section provides additional information about the various Worldpay OmniToken formats. In all examples the randomly generated portion of the token is shown in **green**.

**Informative A and Informative A1**

This token format duplicates the length of the original PAN and preserves both the first-six and the last-four digits of the original PAN. The Informative A version is MOD10 compliant, while the Informative A1 version is MOD10+1 compliant.
Example: Informative A and A1 Formats

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>4418 3789 1620 3675</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenized Value</td>
<td>4418 3711 2222 3675</td>
</tr>
</tbody>
</table>

Last 4 Preserved and Last 4 Preserved 1
This token format duplicates the length of the original PAN and only the last-four digits of the original PAN. The Last 4 Preserved version is MOD10 compliant, while the Last 4 Preserved 1 version is MOD10+1 compliant.

Example: Last 4 Preserved and Last 4 Preserved 1 Formats

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>4418 3789 1620 3675</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenized Value</td>
<td>3938 7311 2222 3675</td>
</tr>
</tbody>
</table>

Completely Random and Completely Random 1
This token format duplicates the length of the original PAN, but does not preserve any portion of the original PAN. The system generates all digits randomly, except the checksum (last) digit, which the system sets to make the overall token either MOD10 or MOD10+1 compliant. The Completely Random version is MOD10 compliant, while the Completely Random 1 version is MOD10+1 compliant.

Example: Completely Random and Completely Random 1 Formats

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>4418 3789 1620 3675</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenized Value</td>
<td>3938 7311 2222 2871</td>
</tr>
</tbody>
</table>

Minimum 16A and Minimum 16A 1
This token format returns a token with a length of sixteen digits or greater depending upon the length of the original PAN, while always preserving the first-six digits and the last-four digits. If the PAN is sixteen digits, the resulting token would be identical to the corresponding Informative format. If the PAN is less than sixteen digits, the resulting token adds random digits in the middle to bring the total to sixteen. If the PAN is greater than sixteen digits, the resulting token preserves the length, replacing the middles digits with random numbers. The Minimum 16A version is MOD10 compliant, while the Minimum 16A 1 version is MOD10+1 compliant.

Example: Minimum 16A and Minimum 16A 1 Formats

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>4418 37891 3675</th>
<th>4418 37891 556438 3675</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenized Value</td>
<td>4418 3711 2222 3675</td>
<td>4418 3711 2222643 3675</td>
</tr>
</tbody>
</table>
Worldpay eComm Merchant A

This token format duplicates the length of the original PAN and preserves the last-four digits of the original PAN. The first-six digits of the PAN are returned in a separate element of the XML response message. The eComm Merchant A version is MOD10+1 compliant.

**Example: eComm Merchant A**

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>Tokenized Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>4418 3789 1620 3675</td>
<td>2875 3711 2222 3675</td>
</tr>
</tbody>
</table>

**NOTE:** Only existing eComm merchants using the Vault tokenization service can use the Comm Merchant A form. Other merchants requiring this format should use the Last 4 Preserved 1 token format.

Direct Debit Token Format

You can also receive tokens for Direct Debit account information. The generated token represents the routing and account numbers. Direct Debit tokens are randomly generated, always 17 characters in length, and MOD10 + 1 compliant. We return the last three digits of the account number in the `<eCheckAccountSuffix>` element.

<table>
<thead>
<tr>
<th>Original Direct Debit Info</th>
<th>Tokenized Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Type: Checking</td>
<td>22590305509028340</td>
</tr>
<tr>
<td>Route/Transit: 123456789</td>
<td></td>
</tr>
<tr>
<td>Account Number: 123456789ABCDEFGH</td>
<td></td>
</tr>
</tbody>
</table>

**1.10.2.2 Low Value Token Summary**

eProtect Registration ID

The eProtect RegistrationID is a Low-Value token that are merchant specific, dynamically generated, and unique to each transaction. Each token is 19 digits (numeric) long, MOD10+1 compliant, with no format preservation, and a lifespan of 24 hours. The eProtect RegistrationID can work in conjunction with any OmniToken format.

**Example:** Registration ID

<table>
<thead>
<tr>
<th>Original PAN Value</th>
<th>Tokenized Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>4872 3789 1620 3675</td>
<td>9857125845785587</td>
</tr>
</tbody>
</table>
eProtect Checkout ID

The eProtect Checkout ID is a Low-Value token dynamically generated, and uniquely assigned to each capture of the Card Security Code (CVC, CVV2) for eCommerce customer verification use cases such as card on file where the merchant has in possession of a Worldpay High-Value token. Each token is 18 digits (numeric) long, MOD10+1 compliant, with no format preservation, and a lifespan of 24 hours. The eProtect Checkout ID can work in conjunction with any OmniToken format.

Example: Checkout ID

<table>
<thead>
<tr>
<th>Original Card Security Code</th>
<th>123</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenized Value</td>
<td>585471254789632587</td>
</tr>
</tbody>
</table>

1.10.3 Obtaining Tokens

NOTE: You must be token enabled and certified prior to using the Vault feature. Please consult your Relationship Manager concerning the requirements and details of this process.

There are three ways for you to obtain tokens for account numbers. First, you can submit an existing card number/Direct Debit account information (account number and routing number) using a Register Token request. When Worldpay receives this transaction type, we generate a token and return it to you via a Register Token response (see Register Token Transactions on page 286.) Although you can use this method to tokenize an account number at any time, it is most useful when initially tokenizing your customer database. Worldpay recommends that you collect all distinct credit card numbers in your database and submit the information in one or more large Batch. When you receive the response file, parse the returned token information to your database, replacing the card numbers.

The second method you can use to obtain a token is to submit a supported transaction with the card information. If you are a tokenized merchant, Worldpay automatically converts the submitted card number to a token and return it to you in the transaction response. Typically, you would use this method when taking and submitting a transaction during the normal course of business. When you receive the response, you store the token instead of the card information.

NOTE: Once we convert a card number to a token for a particular merchant, subsequent submissions of the same card number return the same token.

The third method of obtaining a token applies only to merchants using the Worldpay eProtect feature. In this case, upon submission of an account number via the eProtect API, Worldpay issues a Registration Id. You then submit the Registration Id in an Authorization or Sale transaction and receive the token in the response message.

1.10.3.1 Bulk Token Registration

If you are new to Worldpay, and have utilized tokens with a previous processor, Worldpay can perform a bulk token registration on all the card numbers that were vaulted with your previous processor. The following is an example of the process:
1. During your implementation with Worldpay, you contact your previous processor and request an encrypted mapping file containing the card and token numbers for your customers. A Implementation Consultant will work with you and your previous processor to facilitate the secure transfer of this file without impacting your PCI compliance. The file can be comma-delimited, tab-delimited, or any other common format.

2. Worldpay performs a bulk token registration of all of the card numbers contained in the file.

3. Worldpay returns a mapping file to your organization containing the old tokens and the new Worldpay-issued tokens, so that you can update your order processing system.

Note that Worldpay supports token-extractor formats of all major token service providers. Contact your Implementation Consultant for more information or to initiate this process.

1.10.4 Supported Token Transactions

The following transactions support the generation and use of tokens:

- **Authorization** - You can submit the transaction either with a token or card information. If you submit card information, Worldpay automatically generates the token and returns it in the response.

- **Capture Given Auth** - You can submit the transaction either with a token or card information. If you submit card information, Worldpay automatically generates the token and returns it in the response.

- **Credit** - You can submit the transaction either with a token or card information. If you submit card information, Worldpay automatically generates the token and returns it in the response.

- **Force Capture** - You can submit the transaction either with a token or card information. If you submit card information, Worldpay automatically generates the token and returns it in the response.

- **Register Token** - You use this transaction to convert a card number or Direct Debit account number to a Worldpay token without an associated authorization, verification or payment transaction.

- **Sale** - You can submit the transaction either with a token or card information. If you submit card information, Worldpay automatically generates the token and returns it in the response.

- **eCheck Credit** - You can submit the transaction either with a token or account information. If you submit account information, Worldpay automatically generates the token and returns it in the response.

- **eCheck Redeposit** - You can submit the transaction either with a token or account information. If you submit account information, Worldpay automatically generates the token and returns it in the response.

- **eCheck Sale** - You can submit the transaction either with a token or account information. If you submit account information, Worldpay automatically generates the token and returns it in the response.

- **eCheck Verification** - You can submit the transaction either with a token or account information. If you submit account information, Worldpay automatically generates the token and returns it in the response.

- **Update Card Validation Number** - This is a special transaction type provided to allow the update of a CVV2/CVC2/CID code supplied at the time of the token registration. You should only use this transaction type if you had previously submitted the account number and security code in a registerTokenRequest transaction and now need to change the CVV2/CVC2/CID value.
1.10.5 Compliance with Visa Best Practices for Tokenization

As shown below, the Vault tokenization solution complies with 11 of the 12 items listed in the Visa Best Practices for Tokenization document. The twelfth item concerns the management of stored historical data (that may contain card information) within your systems. Tokenizing all historical card info when implementing the Worldpay solution would satisfy this item, as would protecting it per PCI DSS requirements.

**TABLE 1-13  Visa Best Practices for Tokenization Compliance**

<table>
<thead>
<tr>
<th>Item #</th>
<th>Who</th>
<th>Domain</th>
<th>Best Practice</th>
<th>Complies?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Worldpay</td>
<td>Tokenization System</td>
<td>Network Segmentation</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Worldpay</td>
<td>Tokenization System</td>
<td>Authentication</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>Worldpay</td>
<td>Tokenization System</td>
<td>Monitoring</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Worldpay</td>
<td>Tokenization System</td>
<td>Token Distinguishability</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Worldpay</td>
<td>Token Generation</td>
<td>Token Generation</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Worldpay</td>
<td>Token Generation</td>
<td>Single- vs. Multi- use Tokens</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>Worldpay</td>
<td>Token Mapping</td>
<td>PAN Processing</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Worldpay</td>
<td>Card Data Vault</td>
<td>PAN Encrypted in Storage</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Worldpay</td>
<td>Card Data Vault</td>
<td>Covered by PCI DSS</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>Worldpay</td>
<td>Cryptographic Keys</td>
<td>Key Strength</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Worldpay</td>
<td>Cryptographic Keys</td>
<td>Covered by PCI DSS</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>Merchant</td>
<td>Historical Data</td>
<td>Non-tokenized data protected</td>
<td>Merchant Implementation Decision</td>
</tr>
</tbody>
</table>
1.11 Direct Debit Processing

An Direct Debit (formerly known as eCheck) is an alternative payment method that directly debits a consumer's account via the Automatic Clearing House (ACH) network in the US and the Electronic Funds Transfer (EFT) service in Canada. From a merchant's standpoint offering Direct Debit as a payment method has several advantages, including a large consumer base in excess of 130 million accounts in the United States and no interchange fees.

This section provides information about several Worldpay Direct Debit processing features. Please consult with your Relationship Manager for additional information.

| NOTE: Worldpay also supports Direct Debit processing for our merchants doing business in Canada. We support all Direct Debit transaction types, except Verification and Prenotification. Please consult your Relationship Manager for additional information. |

1.11.1 Validation Feature

Worldpay performs a validation of the Direct Debit routing number. This is done both to verify that the routing number is correctly formatted and that it exists in the Fed database. If the routing number fails this validation, the transaction is rejected. Worldpay performs this validation on all Direct Debit transactions automatically.

1.11.2 Verification Feature

Since there is no authorization process associated with eChecks allowing you to confirm the availability of funds and hold the purchase amount, there is a higher risk of certain types of fraud. The optional eCheck Verification feature allows you to submit an Direct Debit account number for comparison to a database containing historical information about the account, as well as the account holder. When you submit an eCheck Verification transaction the information you provide is compared to a negative database to see if the account is associated with activities, such as fraud, over drafts, or other items determined to be risk factors.

| NOTE: Worldpay makes use of a third party service, Certegy Check Services Inc., for all verification operations. Worldpay does not support the verification service for Payment Facilitators or Canadian eChecks. |

You can also initiate an account verification operation as part of an eCheck Sale transaction by setting the <verify> element to true. In this case, the eCheck Sale transaction is conditional upon the verification passing. If the verification fails, the sale is not processed.

1.11.2.1 Required Contents of Decline Notice

In the event you elect to perform verification on a transaction and also elect not to proceed with the transaction based upon a verification failure, you must provide your customer with the following Decline Notice. You can provide the notice orally, electronically, via e-mail, and/or via U.S Mail, depending upon the type of transaction. The notice must be substantially as the notice set forth below that contains the
disclosures required under the Fair Credit Reporting Act and instructs your customer how to contact Certegy directly.

**NOTE:** If the required language of the Decline Notice changes, Worldpay notifies you of the change. You must enact the changes within 10 days.

**Example: Decline Notice**

We're sorry, but we are unable to proceed with your transaction. This determination was based on information provided by Certegy Check Services, Inc. ("Certegy"). To protect your privacy, Certegy did not provide any financial information to [Client's Name] during the authorization process.

The reason your transaction was not authorized was due to [mark one of the following based on applicable decline code transmitted by Certegy]:

- account closed
- dishonored check or transfer information contained in Certegy's files
- Certegy had insufficient information available
- the identification information you entered did not conform to established guidelines

You have the right under the Fair Credit Reporting Act to know the information Certegy utilized to make a determination regarding your check. If you find that any information Certegy utilized in its decision is inaccurate or incomplete, you have a right to dispute it with Certegy.

You may call Certegy toll free at 800-695-1854, or write to Certegy Check Services, Inc., P.O. Box 30046, Tampa, FL 33680-3046.

If you contact Certegy, please provide the following information so they can respond promptly to your request:

- First Name
- Current Address
- Date Declined
- Dollar Amount
- Check/Draft/Transfer Number
- Checking Account Number
- Driver’s License Number & State
- Home Telephone Number
- Date of Birth
- Social Security Number
- Merchant Name
- Name of Financial Institution

### 1.11.3 Automatic Notice of Change (NoC) Updates

Similar to an issuing bank providing credit card Account Updater information, RDFIs provide Notification of Change (NoC) files and deliver them through the ACH network. These NoCs include updated account information including bank routing numbers, account numbers, and account names.

Worldpay makes available the NoC information to you for your use in updating your customer files. Additionally, if you submit a transaction containing information that has changed, we automatically update the information and forward the corrected transaction to the ACH network. The cnpAPI response message to you also contains the updated information for your use in correcting your database.
1.11.4 Auto Redeposit Feature

NACHA rules allow merchants to redeposit entries when the initial deposit was returned for either Insufficient Funds or Uncollected Funds. Two redeposit attempts are allowed within 180 days of the settlement date of the initial deposit. Worldpay offers an optional service that allows you to preconfigure automatic redeposits of transactions returned for the those reasons. You define the number of days from the initial return for Worldpay to resubmit the transaction. You also define the number of days from the return of the first resubmission for the attempt of a second resubmission.

For example, you submitted an eCheck Sale transaction on 29 January that is returned for Return Reason Code R01 - Insufficient Funds. The return occurs on 1 February. With the Auto Redeposit feature enabled and a preset period of 5 days for the first redeposit, the system would automatically generate a resubmission of the deposit on 6 February. If this transaction is also returned for the same reason code on 7 February and you have a preset time period for the second redeposit on 7 days, the system generates the second redeposit on 14 February.

1.11.5 eCheck Prenotification

An eCheck Prenotification is a non-monetary transaction used to verify the account information supplied by the consumer is valid. These transactions are sent to the ACH network to help ensure subsequent entries are posted appropriately. Since this is a verification of account information, typically, you would submit a Prenotification transaction in advance of processing the order, during the customer set-up process. There are two types of Prenotification transaction types: echeckPreNoteCredit and eCheckPreNoteSale. Per NACHA requirements, you must submit the Prenotification transaction that corresponds to the intended, subsequent transaction. For example, if you are planning to submit an echeckSale transaction and want to verify the account information, you should submit a echeckPreNoteSale.

The possible ACH network responses to a prenotification are as follows:

- **No response** - applies when the account is open and the account information is correct.
- **Notification of Change** - provides updated account information, including correct routing number, e.g. C02 Incorrect routing/transit number (visible in the SSR eCheck NOC report).
- **Return code** - provides account status, e.g. R02 Account is closed, R03 No account on file, R04 Invalid account number.

**NOTE:** Prenotification transactions are only supported in cnpAPI V9.1 or above and only in Batch submissions. Prenotification transactions are not supported for Payment Facilitators or for Canadian eChecks.
When you submit either of the Prenotification transaction types, the system sends an acknowledgment in the Batch response file. This response file does not provide the results of the prenotification check, but rather a verification that we received the transaction and it was properly formatted. You receive the results to the check in the SSR eCheck Notification of Change report. This report is run daily and you can expect to see the results for submitted prenotification checks by the second or third business day after the settlement day. The settlement day for a Prenotification transaction is defined as the next business day after submission to the ACH network. Remember, if the account you are attempting to verify is open and the account information is correct, the report will not contain an entry for that transaction. The report only contains NOCs (update information).

Per NACHA regulations, you can submit the eCheck Sale or eCheck Credit transaction on the third business day after the settlement day; however, there might still be a forthcoming NOC on that day. Due to the timing of the responses from NACHA, the generation of the reports, and the movement of the transactions, you should wait until the fourth day after the settlement day to submit the live transaction unless you receive a NOC earlier. This timing is illustrated in the example below.

1. You submit a Prenotification transaction on Monday prior to your cut-off time.
2. Our system forwards the transaction to NACHA on Tuesday. Note, this makes Wednesday the settlement day.
3. NACHA responds with a NOC/return, if any, by Thursday night.
4. On Friday, the information from NACHA is processed by our systems.
5. On Saturday morning, the SSR NOC report containing the information is available to you.
6. You can submit the eCheck Sale or eCheck Credit transaction before your cut-off time on Saturday night. This is the fourth day after settlement, the fifth day after submission.
1.12 eCommerce Solution for Apple Pay™

Worldpay supports Apple Pay for in-app and in-store purchases initiated on compatible versions of iPhone and iPad, as well as purchases from your desktop or mobile website initiated from compatible versions of iPhone, iPad, Apple Watch, MacBook and iMac (Apple Pay on the Web).

For in-store transactions, a consumer can use the Near Field Communications (NFC) chip in their device to make a purchase by simply touching the device to an NFC-compliant terminal. Identity verification is provided by Touch ID, a fingerprint reading application built into the device. If you wish to allow Apple Pay transactions from your native iOS mobile application, you must build the capability to make secure purchases using Apple Pay into your mobile application.

If you wish to allow Apple Pay payments on your desktop or mobile website, your website must be configured to work with Apple to request authorization from the consumers iPhone or iPad via Touch ID. See Table 1-14, “Apple Pay on the Web Compatible Devices” for more information.

This section provides an overview of the operation of Apple Pay and Apple Pay on the web, along with the several methods you can use to submit Apple Pay purchases to the Worldpay eCommerce platform. The topics discussed are:

- Overview of Apple Pay Operation
- Worldpay Decryption of Apple Pay PKPaymentToken
- Merchant Decryption of Apple Pay PKPaymentToken
- cnpAPI <applepay> Structure
- Worldpay Mobile API for Apple Pay HTTPS POST Components
- Recurring Payments with Apple Pay

1.12.1 Overview of Apple Pay Operation

The operation of Apple Pay and Apple Pay on the web is relatively simple, but will require either the development of new native iOS applications or the modification of your existing applications or website that include the use of the Apple PassKit Framework (or Apple Pay JavaScript for Apple Pay web) and the handling of the encrypted data returned to your application by Apple Pay. The basic steps that occur when a consumer initiates an Apple Pay purchase using your mobile application or website are:

1. When the consumer selects the Apple Pay option from your application, your application makes use of the Apple PassKit Framework (or Apple Pay JavaScript) to request payment data from Apple Pay.
2. When Apple Pay receives the call from your application and after the consumer approves the Payment Sheet (using Touch ID), Apple creates a PKPaymentToken using your public key. Included in the PKPaymentToken is a network (Visa, MasterCard, Discover, or American Express) payment token and a cryptogram.
3. Apple Pay returns the Apple PKPaymentToken (defined in Apple documentation; please refer to https://developer.apple.com/library/ios/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html).

The remainder of this section discusses the various options for handling the PKPaymentToken in the transaction flow.
1.12.2 Worldpay Decryption of Apple Pay PKPaymentToken

Worldpay recommends using one of the Worldpay Decryption methods. This transaction process relieves you from the burden of creating and maintaining public and private keys, as well as decrypting the PKPaymentToken. If you have already implemented eProtect in a mobile application, you should use eProtect for Apple Pay. A second, similar method, which still allows you to submit the PKPaymentToken without decryption, involves you sending the Authorization/Sale transaction with the PKPaymentToken key values in a new cnpAPI element structure (see applepay), typically from your server. This method can be used even if you are not tokenized with Worldpay.

In all of these implementations, your Worldpay Implementation Consultant will provide a CSR (Certificate Signing Request) you use in your registration process with Apple Pay. The CSR provides Apple Pay with the public key used for encryption, while Worldpay retains the private key used for decryption.

1.12.2.1 Using the Browser JavaScript API for Apple Pay on the Web

In this scenario, the Worldpay eProtect Customer Browser JavaScript API controls the fields on your checkout page that hold sensitive card data. When the cardholder clicks the Apple Pay button, communication is exchanged with Apple Pay via the JavaScript API to obtain the PKPaymentToken. From this point forward, your handling of the transaction is identical to any other eProtect transaction. The eProtect server returns a Registration ID (low-value token) and your server constructs the cnpAPI transaction using that ID. See the Worldpay eProtect Integration Guide for JavaScript and HTML page examples and more information on using the browser JavaScript API.

Step 1, Step 2, and Step 3 are the same as those outlined in Overview of Apple Pay Operation on page 50. The process after Step 3 is detailed below (and shown in Figure 1-7):

4. Your website sends the PKPaymentToken to our secure server via the JavaScript Browser API and eProtect returns a Registration ID.

5. Your website forwards the transaction data along with the Registration ID to your order processing server, as it would with any eProtect transaction.

6. Your server constructs/submits a standard cnpAPI Authorization/Sale transaction using the Registration ID, setting the <orderSource> element to applepay.

7. Using the private key, Worldpay decrypts the PKPaymentToken associated with the Registration ID and submits the transaction with the appropriate information to the card networks for approval.

8. Worldpay sends the Approval/Decline message back to your system. This message is the standard format for an Authorization or Sale response and includes the Worldpay token.

9. You return the Approval/Decline message to your website.

Table 1-14 lists the requirements for your customers’ Apple devices when making purchases via Apple Pay on the Web.

NOTE: Table 1-14 represents data available at the time of publication of this document, and is subject to change. See the latest Apple documentation for current information.
### TABLE 1-14 Apple Pay on the Web Compatible Devices

<table>
<thead>
<tr>
<th>Apple Device</th>
<th>Operating System</th>
<th>Browser</th>
</tr>
</thead>
<tbody>
<tr>
<td>iPhone 6 and later</td>
<td>iOS 10 and later</td>
<td></td>
</tr>
<tr>
<td>iPhone SE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iPad Pro</td>
<td>iOS 10 and later</td>
<td></td>
</tr>
<tr>
<td>iPad Air 2 and later</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iPad Mini 3 and later</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Apple Watch</td>
<td>Watch OS 3 and later</td>
<td>Safari only</td>
</tr>
<tr>
<td><em>Paired with iPhone 6 and later</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iMac</td>
<td>macOS Sierra and later</td>
<td></td>
</tr>
<tr>
<td><em>Paired with any of the above mobile devices with ID Touch for authentication</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MacBook</td>
<td>macOS Sierra and later</td>
<td></td>
</tr>
<tr>
<td><em>Paired with any of the above mobile devices with ID Touch for authentication</em></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### FIGURE 1-7 Data/Transaction Flow using Browser JavaScript API for Apple Pay on the Web
1.12.2.2 Using the Worldpay Mobile API for Apple Pay

In this scenario, your native iOS application performs an HTTPS POST of the Apple Pay PKPaymentToken using the Worldpay Mobile API for Apple Pay. From this point forward, your handling of the transaction is identical to any other PayPage transaction. The PayPage server returns a PayPage Registration ID and your Mobile App (or server) constructs the cnpAPI transaction using that ID.

Step 1, Step 2, and Step 3 are the same as those outlined in Overview of Apple Pay Operation on page 50. The process after Step 3 is detailed below and in Figure 1-8.

4. Your native iOS application sends the PKPaymentToken to our secure server via an HTTPS POST and eProtect returns a Registration ID. Please refer to the Worldpay eProtect Integration Guide for additional information.

5. Your native iOS application forwards the transaction data along with the Registration ID to your order processing server, as it would with any eProtect transaction.

6. Your server constructs/submits a standard cnpAPI Authorization/Sale transaction using the Registration ID, setting the <orderSource> element to applepay.

7. Using the private key, Worldpay decrypts the PKPaymentToken associated with the Registration ID and submits the transaction with the appropriate information to the card networks for approval.

8. Worldpay sends the Approval/Decline message back to your system. This message is the standard format for an Authorization or Sale response and includes the Worldpay token.

9. You return the Approval/Decline message to your mobile application.
1.12.2.3 Submitting the Apple Pay PKPaymentToken in a cnpAPI Message

In this scenario, you submit the Apple Pay PKPaymentToken to the Worldpay eCommerce platform using a new cnpAPI structure (see `applepay`), typically from your server. As with the previous scenario, Worldpay decrypts the PKPaymentToken from Apple Pay using the private key.

**Step 1, Step 2, and Step 3** are the same as those outlined in *Overview of Apple Pay Operation* on page 50. The process after Step 3 is detailed below and in *Figure 1-9*.

4. Your mobile application forwards the PKPaymentToken from Apple Pay, along with other normal information from the transaction (such as Bill To and Ship To Address), to your order processing server.

5. You do not decrypt the PKPaymentToken, but rather forward the data to Worldpay in the Authorization/Sale transaction using the cnpAPI `<applepay>` element structure instead of `<card>` (Server-side API submit) and setting the `<orderSource>` element to `applepay`.

6. Using the private key retained by Worldpay, we decrypt the PKPaymentToken and submit the transaction with the appropriate information to the card networks for approval.

7. Worldpay sends the Approval/Decline message back to your system. This message is the standard format for an Authorization or Sale response.
8. You return the Approval/Decline message to your mobile application.

**FIGURE 1-9** Data/Transaction Flow with Direct Submission of PKPaymentToken via cnpAPI

---

### 1.12.2.4 Merchant Decryption of Apple Pay PKPaymentToken

Using this process, the responsibility for the decryption of the PKPaymentToken from Apple Pay falls to you. After completing the first three steps of the process as detailed in the **Overview of Apple Pay Operation** section and depicted by the green and blue arrows in **Figure 1-10**, the process continues as follows:

4. Your mobile application forwards the PKPaymentToken from Apple Pay, along with other normal information from the transaction (such as Bill To and Ship To Address), to your order processing server.

5. Using your private key, you decrypt the PKPaymentToken, construct the Authorization/Sale transaction, and submit it to Worldpay. In this case, you would populate the cnpAPI `<number>` element with the device primary account number, the `<expDate>` element with the expiration date, and the `<authenticationValue>` field with the cryptogram extracted from the PKPaymentToken. Also, set the `<orderSource>` element to `applepay` (Server-side API submit).

6. Worldpay detects that this is an Apple Pay transaction and submits the transaction with the appropriate information to the card networks for approval.

7. Worldpay sends the Approval/Decline message back to your system. This message is the standard format for an Authorization or Sale response.
8. You return the Approval/Decline message to your mobile application.

FIGURE 1-10 Data/Transaction Flow with Merchant Decryption of Apple Pay PKPaymentToken

1.12.3 Recurring Payments with Apple Pay

When you submit the first transaction in a recurring/installment stream, or when storing credentials for future purchases, you must set the `<processingType>` element to either `initialRecurring`, `initialInstallment`, or `initialCOF` (Card on File), as applicable. With the exception of an American Express transaction, the XML response message includes the `<networkTransactionId>` element. You must retain the value returned for use in future transactions. When you submit the next and all subsequent transactions in the recurring/installment stream, set the `<orderSource>` to recurring or installment as appropriate, and include the `networkTransactionId` value in the `<originalNetworkTransactionId>` element. For a CoF transaction, set the `<orderSource>` to `ecommerce` and the `<processingType>` element to either `merchantInitiatedCOF`, or `cardholderInitiatedCOF` (Card on File), as applicable.
1.13 eCommerce Solution for Google Pay™

Google® makes hundreds of products used by billions of people across the globe, from YouTube and Chrome to Google Pay and, of course, Google Search. Hundreds of millions of them add payment cards to their Google Accounts, which they can use to check out from any Google product - without needing to re-enter their payment info.

With the new Google Pay API, you can enable the same effortless checkout experience for your own products and services. The new API is an on-line payment method that lets your customers use the cards they've saved to their Google Account to pay quickly and easily in your apps. and on your websites. By clicking the Google Pay button, customers can choose a payment method saved in their Google Account and finish checkout in a few, simple steps.

You can use the Google Pay API to simplify payments for customers who make purchases in Android applications or on Chrome with a mobile device.

Worldpay supports two methods for merchants to submit Google Pay transactions from Mobile applications to the eCommerce platform. The preferred method involves you sending certain Worldpay-specific parameters to Google. The response from Google includes a Worldpay paypageRegistrationId, which you use normally in your payment transaction submission to Worldpay. With the alternate method, you receive encrypted information from Google, decrypt it on your servers, and submit the information to Worldpay in a payment transaction.

This document provides an overview of both methods and includes the following sections:

- Branding Requirements
- Google Pay using eProtect
- Merchant Decryption Method
- Recurring Payments with Google Pay

1.13.1 Branding Requirements

You are required to implement the Google Pay branding requirement on pages where the consumer is presented with payment options. Go to https://developers.google.com/payments/brand-guidelines for more information, including guidelines on the Google brand mark, button label, and button container.

1.13.2 Google Pay using eProtect

This is the recommended and typical method of implementing Google Pay for Mobile Applications on the Worldpay eCommerce platform. The steps that follow, along with Figure 1-11, illustrate the high level flow of messages associated with an Google Pay purchase, when utilizing the Worldpay eProtect service.

**NOTE:** This process assumes you have integrated with Google using the method that returns the Worldpay low-value token (paypageRegistrationId) from Google following the Full Wallet request.

1. When the consumer clicks the Google Pay button in your application, the action triggers a PaymentDataRequest to Google. In the PaymentDataRequest, you must set a new object
PaymentMethodTokenizationParameters indicating that you are using Worldpay. Use the following code sample as a guide to setting this field.

**Setting the PaymentMethodTokenizationParameters**

```java
PaymentMethodTokenizationParameters parameters = PaymentMethodTokenizationParameters.newBuilder()
    .setPaymentMethodTokenizationType(PaymentMethodTokenizationType.PAYMENT_GATEWAY)
    .addParameter("gateway","vantiv")
    .addParameter("vantiv:merchantPayPageId",payPageId)
    .addParameter("vantiv:merchantOrderId",orderId)
    .addParameter("vantiv:merchantTransactionId",id)
    .build();
```

**NOTE:** Use the same `orderId` value on all calls (i.e., Google, Register Token, Authorization, Sale, etc.). By using the same `orderId`, customers can track their orders when using a Google-provided app.

**Setting New Object in the PaymentDataRequest**

```java
PaymentDataRequest request = PaymentDataRequest.newBuilder()
    .addAllowedPaymentMethods(new List<int>() { WalletConstants.PAYMENT_METHOD_CARD,
                                                      WalletConstants.PAYMENT_METHOD_TOKENIZED_CARD})
    .setMerchantName(Constants.MERCHANT_NAME)
    .setPhoneNumberRequired(true)
    .setShippingAddressRequired(true)
    .setCurrencyCode(Constants.CURRENCY_CODE_USD)
    .setEstimatedTotalPrice(cartTotal)
    .setCart(Cart.newBuilder()
                .setCurrencyCode(Constants.CURRENCY_CODE_USD)
                .setTotalPrice(cartTotal)
                .setLineItems(lineItems)
                .build())
    .setPaymentMethodTokenizationParameters(parameters)
    .build();
```

The information returned by Google in the PaymentDataRequest object may include a masked card number (last-four digits exposed) and shipping information. The consumer has the option of changing this information. If any info changes, Google Pay returns an updated PaymentDataRequest object.
2. Upon confirmation of the order by the consumer your application initiates a `FullWalletRequest` to Google.

3. After receiving the `FullWalletRequest` from your application, Google submits the card information to Worldpay eProtect. The eProtect servers return a low-value token (`paypageRegistrationId`).

4. Google returns the low-value token to your application along with the Full Wallet information.

5. Your application sends the transaction information to your servers along with the low-value token. Your servers submit the Auth/Sale transaction to the Worldpay eComm platform. You must set the `orderSource` to `androidpay` in the transaction.

NOTE: Instead of submitting a Auth/Sale transaction, you can submit a Register Token transaction to convert the low-value token to a Worldpay high-value token. You would then use the high-value token in subsequent transactions submitted to the eComm platform.

6. Worldpay processes your transaction normally and returns the results along with a high-value token.
1.13.3 Merchant Decryption Method

Using this process, the responsibility for the decryption of the encrypted payload from Google Pay falls to you. The steps that follow, along with Figure 1-12, illustrate the high level flow of messages associated with an Google Pay purchase, when you perform the decryption of the encrypted payload.

NOTE: The process assumes you have integrated with Google using the method that returns the encrypted payload from Google following the Full Wallet request.

1. When the consumer clicks the Google Pay button in your application, the action triggers a PaymentDataRequest to Google. The information returned by Google in the
PaymentDataRequest object may include a masked card number (last-four digits exposed) and shipping information. The consumer has the option of changing this information. If any info changes, Google Pay returns an updated PaymentDataRequest object.

2. Upon confirmation of the order by the consumer your application initiates a FullWalletRequest to Google. Google also returns the encrypted payload. The encrypted payload is a UTF-8 encoded serialized JSON dictionary with the following keys:
   - `encryptedMessage` (string base64) - an encrypted message containing the payment credentials
   - `ephemeralPublicKey` (string base64) - the ephemeral public key associated with the private key to encrypt the message
   - `tag` (string base64) - MAC of encryptedMessage

3. Your application sends the encrypted payload along with the transaction information to your server.

4. Your server decrypts the encrypted payload extracting the payment, which is a UTF-8 encoded, serialized JSON dictionary with the following keys:
   - `dpan` (string (digits only)) - the device-specific personal account number (i.e., device token)
   - `expirationMonth` (number) - the expiration month of the DPAN (1 = January, 2 = February, etc.)
   - `expirationYear` (number) - The four-digit expiration year of the DPAN (e.g., 2015)
   - `authMethod` (string) - the constant 3DS (may change in future releases).
   - `3dsCryptogram` (string) - the 3DSecure cryptogram
   - `3dsEciIndicator` ((optional) string) - ECI indicator per 3DSecure specification

Example of Decrypted Credentials in JSON

```
{
    "dpan": "4444444444444444",
    "expirationMonth": 10,
    "expirationYear": 2018,
    "authMethod": "3DS",
    "3dsCryptogram": "AAAAAA...",
    "3dsEciIndicator": "eci indicator"
}
```

After decryption, submit the Authorization/Sale transaction to Worldpay, setting the `orderSource` element to `androidpay` and populating the following cnpAPI elements with the decrypted information:

- **number** - dpan value
- **expDate** - MMYY derived from the expirationMonth and expirationYear values
- **authenticationValue** - the 3dsCryptogram value

5. Worldpay processes your transaction normally and returns the results in the response message.
1.13.4 Recurring Payments with Google Pay

When you submit the first transaction in a recurring/installment stream, or when storing credentials for future purchases, you must set the `<processingType>` element to either `initialRecurring`, `initialInstallment`, or `initialCOF` (Card on File), as applicable. With the exception of an American Express transaction, the XML response message includes the `<networkTransactionId>` element. You must
retain the value returned for use in future transactions. When you submit the next and all subsequent transactions in the recurring/installment stream, set the \(<orderSource>\) to recurring or installment as appropriate, and include the \(\text{networkTransactionId}\) value in the \(<\text{originalNetworkTransactionId}>\) element. For a CoF transaction, set the \(<orderSource>\) to ecommerce and the \(<\text{processingType}>\) element to either \(\text{merchantInitiatedCOF}\), or \(\text{cardholderInitiatedCOF}\) (Card on File), as applicable.
1.14 Amazon Pay

Amazon Pay provides consumers a secure, trusted, and convenient method of using their Amazon account/credentials to make purchases on merchant sites. To do this, Amazon embeds a widget into your checkout flow that allows the consumer to sign into their Amazon account and select one of their payment methods, without leaving your site. You receive an token from Amazon Pay, which you submit to Worldpay in an authorization or Sale transaction. Figure 1-13 provides an overview of the Amazon Pay/Worldpay message flow.

FIGURE 1-13  Amazon Pay Message Flow

1. After the consumer signs in and consents to the charge using Amazon Pay, your system communicates with Amazon Pay receiving a permission ID. The consumer then selects one of the payment methods saved in the Amazon Pay profile. Your systems again communicates with Amazon Pay receiving an Amazon Pay token and payment information in return.
2. Your system submits an Authorization or Sale transaction to Worldpay (see Authorization example below), using the Amazon Pay token (cnpToken element).

3. Worldpay communicates with Amazon Pay to retrieve the PAN information associated with the Amazon Pay token. We then send the transactional information to the card brands for approval.

4. Worldpay sends the Approval/Decline back to your system.

Example: Authorization Message with Amazon Pay Token

```xml
<authorization id="99999" customerId="444" reportGroup="RG1">
  <orderId>F12345</orderId>
  <amount>15000</amount>
  <orderSource>telephone</orderSource>
  <billToAddress>
    <name>John Doe</name>
    <addressLine1>15 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
    <phone>9782750000</phone>
    <email>jdoe@worldpay.com</email>
  </billToAddress>
  <shipToAddress>
    <name>Jane Doe</name>
    <addressLine1>15 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
    <phone>9782750000</phone>
    <email>jdoe@worldpay.com</email>
  </shipToAddress>
  <token>
    <cnpToken>111100010103AMZN</cnpToken>
  </token>
</authorization>
```
1.15 Healthcare Card Feature

Today, there are several types of Healthcare accounts that allow participants to use pre-tax money for the purchase of IRS approved healthcare products and services, such as prescription medications and office visit payments/co-pays. The most common of these accounts are Flexible Spending Accounts (FSAs), Health Reimbursement Arrangements (HRAs), and Health Savings Accounts (HSAs). In order to provide consumers with a more convenient method of making use of these accounts, certain issuers provide signature based, MasterCard or Visa branded, healthcare payment debit cards.

To facilitate the processing of transactions related to these cards, Worldpay has augmented the cnpAPI with elements specific to Healthcare card purchases. We added the `healthcareIIAS` element to the Authorization and (Conditional) Sale transaction types. You use this element and its children to detail the costs associated with Healthcare related, IIAS approved items purchased by the consumer. The example below shows an Authorization transaction with IIAS items.

**Example:** Authorization with healthcareIIAS Element

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <authorization id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>123456789</orderId>
    <amount>6500</amount>
    <orderSource>ecommerce</orderSource>
    <billToAddress>
      <name>John Smith</name>
      <addressLine1>100 Main St</addressLine1>
      <city>Boston</city>
      <state>MA</state>
      <zip>12345</zip>
      <email>jsmith@someaddress.com</email>
      <phone>555-123-4567</phone>
    </billToAddress>
    <card>
      <type>VI</type>
      <number>4000000000000001</number>
      <expDate>1211</expDate>
    </card>
  </authorization>
</litleOnlineRequest>
```

**NOTE:** The example below includes the `dentalAmount` element to show all available amount classifications. Since this is an optional element and has a value of 0 in the example, you can omit it. Also, note that the `visionAmount` is not included in the `totalHealthcareAmount`, since this is a Visa transaction.
<cardValidationNum>555</cardValidationNum>
</card>
<cardholderAuthentication>
  <authenticationValue>BwABBJQ1gJDUCAAAAAA=</authenticationValue>
  <authenticationTransactionId>gMV75TmjAgk=</authenticationTransactionId>
</cardholderAuthentication>
<allowPartialAuth>true</allowPartialAuth>
<healthcareIIAS>
  <healthcareAmounts>
    <totalHealthcareAmount>5500</totalHealthcareAmount>
    <RxAmount>4000</RxAmount>
    <visionAmount>1000</visionAmount>
    <clinicOtherAmount>1500</clinicOtherAmount>
    <dentalAmount>0</dentalAmount>
  </healthcareAmounts>
  <IIASFlag>Y</IIASFlag>
</healthcareIIAS>
</authorization>
</litleOnlineRequest>

Please keep in mind the additional following requirements/recommendations:

**NOTE:** The information below is not intended as an exhaustive list of the requirements for the acceptance of Healthcare cards by merchants. While Worldpay may be able to provide some information, merchants are responsible for the awareness of and adherence to all applicable regulatory requirements imposed by the Internal Revenue Service, other government agencies, and other interested parties (for example, Visa, MasterCard, SIGIS, etc.)

- You must become a member of the Special Interest Group for IIAS Standards (SIGIS)
- You must obtain and maintain SIGIS certification
- Merchants, except those with healthcare related MCCs, must have an Inventory Information Approval System (IIAS) used to identify eligible healthcare purchases as defined and required by the Internal Revenue Code.
- You must obtain special account numbers from Visa and MasterCard to process these transactions
- You must support data retention and retrieval of line item details for eligible healthcare products included in Healthcare Card transactions
- You must complete Worldpay Certification Testing for the use of this feature, as well as the Partial Auth feature.
- For Visa transaction, do not include the `visionAmount` in the `totalHealthcareAmount`.
- Transactions must include the `IIASFlag` element set to `Y`. 
1.16 Supported Transaction Types

cnpAPI Batch processing supports all transaction types except Voids, eCheck Voids and Private Label Gift Card reversal transactions, which are handled as Online transactions only. Online processing handles all transaction types. This section provides a description of each transaction type, information concerning its use, and any special considerations.

NOTE: For information about Dynamic Payout Funding Instructions please refer to Appendix D, "PayFac Dynamic Payout".

1.16.1 Authorization Transaction

The Authorization transaction enables you to confirm that a customer has submitted a valid payment method with their order and has sufficient funds to purchase the goods or services they ordered. Setting the `<allowPartialAuth>` element to `true` in the Authorization request enables the system to return authorizations for a portion of the order amount for cases where the card does not have an adequate credit limit or balance available for the full amount.

An approved Authorization reduces the customer's credit limit (or bank balance, in the case of a debit card) by the amount of the approval by placing the amount on hold. If you have the Prepaid Indicator feature enabled, the Authorization response also includes an element that indicates if the card is Prepaid, as well as an element indicating the available balance on the card.

NOTE: To obtain better interchange rates, you should always perform an AVS check either as a standalone transaction, or as part of the Authorization/Sale transaction.

If you settle in a currency other than US or Canada, you cannot use partial Authorizations.

While most merchants perform Authorizations as Online transactions, there is no requirement to do so.

The lifespan of an Authorization depends upon the payment method. Table 1-15 provides information concerning Authorization lifespans for various card types and alternate payment methods.

<table>
<thead>
<tr>
<th>Payment Type</th>
<th>Lifespan of Authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterCard</td>
<td>7 days</td>
</tr>
<tr>
<td>Visa</td>
<td>7 days</td>
</tr>
<tr>
<td>American Express</td>
<td>7 days</td>
</tr>
<tr>
<td>Discover</td>
<td>10 days</td>
</tr>
<tr>
<td>PayPal</td>
<td>29 days total by default; Worldpay recommends capture submission within three days. For more information, see the Worldpay PayPal Integration Guide.</td>
</tr>
</tbody>
</table>

TABLE 1-15 Lifespan of Payment Authorizations
As long as the authorization has not expired, or the amount exhausted, you can use it repeatedly to fulfill an order. This would be the case if the Authorization covered multiple items with staggered deliveries. In this scenario, you would issue a Partial Capture transaction as each item shipped until the order was completely fulfilled.

**NOTE:** If you obtain an Authorization through approved vendors for voice and terminal authorizations, you would use a Capture Given Auth transaction to deposit the funds (see Capture Given Auth Transaction on page 72).

### 1.16.1.1 AVS Only Transaction

An AVS Only transaction is a variation of an Authorization transaction that uses the Address Verification System to enable you to verify that a customer supplied address matches the billing address associated with the card. To submit an AVS Only transaction, submit an Authorization transaction with the `<amount>` element set to 000 and the optional `billToAddress` element with appropriate child values.

**NOTE:** To obtain better interchange rates, you should always perform an AVS check either as a standalone transaction, or as part of the Authorization/Sale transaction.

### 1.16.2 Authorization Reversal Transactions

The primary use of Authorization Reversal transactions is to eliminate any unused amount on an unexpired Authorization. Issuing an Authorization Reversal has the benefit of freeing any remaining held amount that reduces the buying power of your customer. Potentially, this both increases customer satisfaction and can allow them to proceed with additional purchases that may otherwise be blocked by credit limits. It also helps you avoid any misuse of Auth fees imposed by the card associations.

**NOTE:** For American Express transactions, the reversal amount must match the authorization amount. American Express does not allow partial reversals and reversals against remaining amounts after a partial capture. Attempts to perform these types of reversals result in a Response Code of 336 - Reversal amount does not match Authorization amount.

For example, consider the following scenario. A customer with a $6,000 credit limit orders a $3,000 stereo system, but the stereo is a discontinued item. The merchant notifies the customer, but does not perform Authorization Reversal. The customer attempts to submit a new order for a $3,001 stereo.

- If the customer uses the same credit card for both orders, the second order could be denied, since the account’s remaining credit limit is only $3,000.
- If the customer had used the same debit card for both orders, the second order places the customer’s bank account in an overdraft situation (assuming a starting balance of $6,000).

Either of these situations can result in the merchant losing a sale, as well as receiving a call from an angry customer.

Another advantage of Authorization Reversal transactions occurs on Visa transactions. In order for you to qualify for the best possible interchange rates from Visa, the amount of the Capture must match the
amount of the associated Authorization. In order to take advantage of this situation for you, if the Capture amount is less than the associated Authorization amount, Worldpay automatically performs a partial Authorization Reversal for the unused amount (also see Capture Request on page 214).

1.16.2.1 Notes on the Use of Authorization Reversal Transactions

This section provides additional information concerning the requirements of and exceptions to the use of Authorization Reversal transactions.

- **Authorization Reversal transactions are supported for the following methods of payment: PayPal, MasterCard, Visa, Discover, Diners Club, and JC and American Express, but American Express and PayPal only support reversals of the entire Authorized amount (no partial reversals).**
- **All transactional data, including associated Authorizations, Captures, and Refunds, must use cnpAPI format.**
- **Worldpay recommends that you send the Authorization Reversal transaction using the same submission method (Batch or Online) as used for the Capture transaction. This is to eliminate a possible race condition that may occur if you submit an Online Authorization Reversal prior to the processing of a Batch containing the associate Capture transaction.**
- **For Batch transactions, send Authorization Reversal transactions in a session separate from the both the associated Authorization and the associated Capture transactions.**
- **For Online transactions, when following an Authorization with an Auth Reversal, allow a minimum of one minute between the transactions.**
- **For Visa transactions and MasterCard, Worldpay automatically performs a partial Authorization Reversal, if the Capture amount is less than the associated Authorization amount.**
- **If you do not specify an amount (<amount> element) in the Authorization Reversal, Worldpay reverses the total amount of the associated Authorization.**
- **Do not send an Authorization Reversal against an expired Authorization. This results in a 306 - Auth expired, so amount does not need to be reversed error. When an Authorization expires, the hold amount is automatically reversed.**
- **Do not send an Authorization Reversal against an Authorization that does not exist in our system. For example, if you sends a reversal against an Authorization that failed or an Authorization that was declined, the Authorization Reversal returns a 360 - No transaction found with specified transaction Id error.**
- **Do not send an Authorization Reversal against a payment type that does not support Authorization Reversals. This results in a 335 - This method of payment does not support reversals error.**
- **Do not send an Authorization Reversal for a fully depleted Authorization. This results in a 111 - Authorization amount has already been depleted error.**
- **Do not use an Authorization Reversal to reverse an Authorization on a (Closed Loop) Gift Card. Use a Gift Card Auth Reversal instead (see Gift Card Auth Reversal Transactions on page 270).**

1.16.2.2 Using Authorization Reversal to Halt Recycling Engine

If you are using the Recycling Engine to optimize your authorizations and need to discontinue the automatic recycling of the transaction, you use an Authorization Reversal transaction to halt the retries. For example, if a customer cancels an order and the authorization for the order is being retried by the Recycling Engine, you submit an Authorization Reversal transaction to halt the automatic recycling of the authorization.
1.16.3 Activate Transaction

You use an Activate transaction to change the status of a (Closed Loop) Gift Card from an inactive to an active state with a value as specified by the `amount` element. You can also use this transaction type to create a Virtual Gift Card.

1.16.4 Activate Reversal Transaction (Online Only)

You use as Activate Reversal transaction to change the status of a newly activated (Closed Loop) Gift Card from active to inactive, thus reversing the operation of an Active transaction. The Activate Reversal references the associated Activate transaction by means of the `litleTxnId` element returned in the Activate response.

1.16.5 Balance Inquiry Transaction

You use the Balance Inquiry transaction to determine the balance available for use on a (Closed Loop) Gift Card.

1.16.6 Cancel Subscription Transaction

If you are using the Recurring Engine, you create Subscriptions as part of an Authorization or Sale transaction. You use the Cancel Subscription transaction to cancel the specified subscription, removing the transaction stream managed by the Recurring Engine.

1.16.7 Capture Transaction

You use a Capture transaction to transfer previously authorized funds from the customer to you after order fulfillment. You can submit a Capture transaction for the full amount of the Authorization, or for a lesser amount by setting the `partial` attribute to `true`.

NOTE: If submitted a Sale (conditional deposit) as the initial transaction rather than an Authorization, you use a Void transaction to halt the recycling.

Do not use a Capture transaction to capture funds on a (Closed Loop) Gift Card transaction; use a Gift Card Capture instead (see Gift Card Capture Transactions on page 272).
1.16.8 Capture Given Auth Transaction

Similar to a Capture transaction, you use a Capture Given Auth transaction to transfer previously authorized funds from the customer to you after fulfillment. However, you typically use a Capture Given Auth transaction if the associated Authorization occurred outside of the system (for example, if you received a telephone Authorization). Another possible use for a Capture Given Auth transaction is if the Authorization transaction occurred within the system, but the \texttt{littletxnId} is unknown by the submitting party (for example, if the Auth was submitted by a merchant, but a fulfiller submits a Capture Given Auth).

Whenever you submit a Capture Given Auth transaction, Worldpay attempts to match it to an existing Authorization using COMAAR data (\texttt{Card Number}, \texttt{Order Id}, \texttt{Merchant Id}, \texttt{Amount}, \texttt{Approval Code}, and (\texttt{Auth}) \texttt{Response Date}) in order to obtain a better Interchange rate for the transaction. The application uses the following matching logic:

- If the \texttt{Order Id} was either not submitted (blank, spaces, or null) or does not match any Auth in the system, it is ignored and the matching attempt proceeds using the remaining COMAAR data.
- If the matching operation results in multiple possible matches, the application selects the Authorization with the lowest amount that is greater than or equal to the Capture Given Auth amount. Whenever a Capture Given Auth transaction is submitted, Worldpay uses the following matching logic:

\begin{itemize}
  \item If the \texttt{Order Id} was either not submitted (blank, spaces, or null) or does not match any Auth in the system, it is ignored and the matching attempt proceeds using the remaining COMAAR data.
  \item If the matching operation results in multiple possible matches, the application selects the Authorization with the lowest amount that is greater than or equal to the Capture Given Auth amount.
\end{itemize}

\begin{itemize}
  \item If necessary, the application further narrows the match candidates to the one with the most recent response date.
\end{itemize}

\textbf{NOTE:} In all cases, the Authorization amount must always be greater than or equal to the Capture Given Auth amount.

\textbf{NOTE:} If Worldpay can match the Capture Given Auth to an Authorization and the following conditions are met: the card type is Visa and the Capture Given Auth amount is less than the Authorization amount, then Worldpay will issue an Auth Reversal transaction for the balance of the Authorization.

\begin{itemize}
  \item This is done to obtain the best possible interchange rates from Visa.
\end{itemize}

1.16.9 Create Plan Transaction

You use the Create Plan transaction to define several attributes of a recurring payment schedule. Later, as part of an Authorization or sale transaction, you can associate existing, active plans with Subscriptions. This association establishes a recurring payment schedule managed by the Worldpay Recurring Engine.

1.16.10 Credit Transaction

You use a Credit transaction to refund money to a customer, even if the original transaction occurred outside of the system. You can submit refunds against any of the following payment transactions:

- Capture Transaction
- Capture Given Auth Transaction
- Force Capture Transaction
- Sale Transaction

\textbf{NOTE:} In all cases, the Authorization amount must always be greater than or equal to the Capture Given Auth amount.

\textbf{NOTE:} If Worldpay can match the Capture Given Auth to an Authorization and the following conditions are met: the card type is Visa and the Capture Given Auth amount is less than the Authorization amount, then Worldpay will issue an Auth Reversal transaction for the balance of the Authorization.

\begin{itemize}
  \item This is done to obtain the best possible interchange rates from Visa.
\end{itemize}
• External Sale or Capture

**NOTE:** Worldpay recommends you send all Credit transactions in a Batch separate from the associated Capture or Sale transactions.

1.16.11 **Deactivate Transaction**

You use a Deactivate transaction to change the status of a (Closed Loop) Gift Card from an active to an inactive state.

1.16.12 **Deactivate Reversal Transaction (Online Only)**

You use a Deactivate Reversal transaction to change the status of a newly deactivated Gift Card from inactive to active, thus reversing the operation of an Deactivate transaction. The Deactivate Reversal references the associated Deactivate transaction by means of the *litleTxnId* element returned in the Deactivate response.

1.16.13 **Deposit Reversal Transaction (Online Only)**

Used only for (Closed Loop) Gift Card related transactions, a Deposit Reversal transaction to reverse the funds capture initiate by either a Capture or Sale transaction. The Deposit Reversal references the associated Capture/Sale transaction by means of the *litleTxnId* element returned in the Capture/Sale response. You should never attempt to use this transaction type to reverse credit card or Direct Debit transactions.

1.16.14 **eCheck Credit Transaction**

Similar to a Credit transaction, you use an eCheck Credit transaction to refund money to a customer, but only when the method of payment was Direct Debit. You can submit an eCheck Credit transaction regardless of whether the original transaction occurred in or out of the system.

1.16.15 **eCheck Prenotification Credit Transaction**

You use this non-monetary transaction to verify the consumer’s account information prior to submitting an eCheck Credit transaction (also see [eCheck Prenotification](#) on page 48). This transaction type is only supported for US transactions.

1.16.16 **eCheck Prenotification Sale Transaction**

You use this non-monetary transaction to verify the consumer’s account information prior to submitting an eCheck Sale transaction (also see [eCheck Prenotification](#) on page 48). This transaction type is only supported for US transactions.
1.16.17 eCheck Redeposit Transaction
You use this transaction type to manually attempt redeposits of eChecks returned for either Insufficient Funds or Uncollected Funds. You can use this element in either Online or Batch transactions.

NOTE: Do not use this transaction type if you are enabled for the Auto Redeposit feature. If you are enabled for the Auto Redeposit feature, the system will reject any echeckRedeposit transaction you submit.

1.16.18 eCheck Sales Transaction
You use an eCheck Sales transaction to transfer funds from the customer to you after order fulfillment. It is the Direct Debit equivalent of a Capture transaction. Funding usually occurs within two days. You can also submit this transaction type as a conditional capture, which makes the processing of the deposit conditional upon a successful verification. If the verification fails, the deposit is not processed.

1.16.19 eCheck Verification Transaction
You use an eCheck Verification transaction to initiate a comparison to a database containing information about checking accounts. The database may include information as to whether the account has been closed, as well as whether there is a history of undesirable behavior associated with the account/account holder. This transaction type is only supported for US transactions.

1.16.20 eCheck Void Transaction (Online Only)
You use an eCheck Void transaction to either halt automatic redeposit attempts of eChecks returned for either Insufficient Funds or Uncollected Funds, or cancel an eCheck Sale transaction, as long as the transaction has not yet settled. This also applies to merchant initiated redeposits. You can use this element only in Online transactions.

1.16.21 Force Capture Transaction
A Force Capture transaction is a Capture transaction used when you do not have a valid Authorization for the order, but have fulfilled the order and wish to transfer funds.

CAUTION: Merchants must be authorized by Worldpay before processing this transaction. In some instances, using a Force Capture transaction can lead to chargebacks and fines.

1.16.22 Gift Card Auth Reversal
You use the Gift Card Auth Reversal transaction to reverse an Authorization on a (Closed-loop) Gift Card. This removes the hold on the previously authorized amount, freeing those funds for additional purchases by the cardholder.
1.16.23 Gift Card Capture
You use the Gift Card Capture transaction to capture or deposit previously authorized funds on a (Closed-loop) Gift Card.

1.16.24 Gift Card Credit
You use a Gift Card Credit transaction to refund money to a customer that used a (Closed Loop) Gift Card for the purchase, even if Worldpay did not process the purchase. You have a choice of two structures. You use the message type containing a transaction Id to apply a credit for a transaction processed by Worldpay. You use the message type containing a order Id to apply a credit for a transaction not processed by Worldpay.

1.16.25 Load Transaction
You use a Load transaction to add funds to an active Gift Card. The load amount cannot exceed the maximum allowed amount for the Gift Card. If you attempt to load more than the maximum amount, the transaction will be declined with a response Code of 221 - Over Max Balance.

1.16.26 Load Reversal Transaction (Online Only)
You use a Load Reversal transaction to reverse the operation of a Load transaction, removing the newly loaded amount from the Gift Card. You cannot perform a partial Load Reversal. This transaction always reverses the full amount of the referenced Load transaction.

1.16.27 Refund Reversal Transaction (Online Only)
The Refund Reversal transaction is a (Closed Loop) Gift Card only transaction that reverses the operation of a Refund transaction on the Gift Card. You cannot perform a partial Refund Reversal. This transaction always reverses the full amount of the referenced Refund transaction.

1.16.28 Register Token Transaction
You use the Register Token transaction to submit a credit card number, Direct Debit account number, or Registration Id to us and receive a token in return. While you can submit Register Token transactions at any time, typically, you would make use of this transactions when initially converting to the use of tokens. In this case you would submit large quantities of credit cards/Direct Debit account numbers in Batches and replace them in your database with the tokens returned.

1.16.29 Sale Transaction
The Sale transaction enables you to both authorize fund availability and deposit those funds by means of a single transaction. The Sale transaction is also known as a conditional deposit, because the deposit takes place only if the authorization succeeds. If the authorization is declined, the deposit will not be processed.
1.16.30 Status Query Transaction

The Status Query Transaction allows you to verify that an Online transaction submitted within the prior 24 hours exists in the system. The response will be one of the following:

- A single transaction matching the search criteria
- Multiple transactions matching the search criteria
- Empty results, if no transactions matched the criteria
- A limited response, if a transaction was found, but processing was not complete

As search criteria, you must submit, at a minimum, the id (id attribute) and transaction type (i.e., authorization, deposit, void, etc.) of the original transaction, but to narrow the search, you can also include the transaction id, order id, and the account number (credit, debit, or gift card) from the original transaction. The response message contains one of four response codes, 150 through 153 (see Payment Transaction Response Codes on page 786), and the results for the search.

NOTE: Use of this transaction type requires specific permissions. Please speak to your Worldpay Implementation Consultant for additional information.

1.16.31 Unload Transaction

You use an Unload transaction to remove funds from an active Gift Card. The unload amount cannot exceed the available balance on the Gift Card. If you attempt to unload more than the available balance, the transaction will be declined with a response Code of 209 - Invalid Amount.

1.16.32 Unload Reversal Transaction (Online Only)

The Unload Reversal transaction reverses the operation of a Unload transaction, returning the value removed from the Gift Card by the Unload transaction. You cannot perform a partial Unload Reversal. This transaction always reverses the full amount of the referenced Unload transaction.

1.16.33 Update Card Validation Number Transaction

When you submit the CVV2/CVC2/CID in a registerTokenRequest, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. This is done to accommodate merchant systems/workflows where the security code is available at the time of token registration, but not at the time of the Auth/Sale. If for some reason you need to change
the value of the security code supplied at the time of the token registration, use an updateCardValidationNumOntoken transaction.

1.16.34 Update Plan Transaction

You use the Update Plan transaction to activate/deactivate Plans associated with recurring payments. When you deactivate a Plan, by setting the active flag to false, you can no longer reference that Plan for use with subscriptions. Existing subscriptions already using the deactivated Plan will continue to use the Plan until the subscription is either modified or completed. You can also reactivate a deactivated Plan by updating the Plan and setting the active flag to true.

1.16.35 Update Subscription Transaction

You use an Update Subscription transaction to change certain subscription information associated with a recurring payment. Using this transaction type you can change the plan, card, billing information, and/or billing date. You can also create, update, or delete a Discount and/or an Add On.

1.16.36 Void Transaction (Online Only)

The Void transaction enables you to cancel any settlement transaction as long as the transaction has not yet settled (see Multiple Daily Delivery on page 5) and the original transaction occurred within the system (Voids require a reference to a littleTxnId). Before submitting a Void, please allow a minimum of 60 seconds to elapse after submitting the transaction you wish to void. This timing ensures our system fully records the first (to be voided) transaction in our database.

NOTE: Do not use Void transactions to void an Authorization. To remove an Authorization use an Authorization reversal transaction (see Authorization Reversal Transactions on page 69.)

1.16.36.1 Using Void to Halt Recycling Engine

If you use Recovery or the Recycling Engine service and use Sale transactions (conditional deposits) to authorize and capture the funds, you must use a Void transaction to discontinue the automatic recycling of the transaction should the need arise. For example, if a customer cancels an order and the Sale transaction is being retried by the Recycling Engine, you submit a Void transaction to halt the automatic recycling of the transaction.

NOTE: If you submitted an Authorization rather than an Sale as the initial transaction, use an Authorization Reversal transaction to halt the recycling.

When using a Void transaction to halt recycling, there is a possibility that the recycled transaction has already been approved and captured. If this condition occurs, depending upon your configuration, the system takes one of two actions:
If you are not configured for the Automatic Refund option (default = disabled), the system declines the Void transaction. You must issue the Credit transaction to refund the money to the consumer. The daily Recycling file will include the approved/captured transaction.

If you are configured for the Automatic Refund option, the system issues a Credit transaction on your behalf. The system returns the transaction Id for the Credit transaction in the Void response message (creditLitleTnxId element). The daily Recycling file will include the approved/captured transaction only if the file was generated prior to the system receiving the Void and issuing the automatic refund.

1.16.37 Instruction-Based Dynamic Payout Transactions

If you are a Payment Facilitator using the Instruction-Based Dynamic Payout model, there are a number of Batch only transaction types you use to move funds between various accounts, including funding Sub-merchants. This section provides information about these transaction types. For additional information, please refer to the Appendix D, "PayFac Dynamic Payout".

- **Funding Instruction Void** - You use this transaction type to void a submitted funding instruction. You must submit the void prior to your daily cutoff time for an instruction not yet processed.

- **PayFac Credit Transaction** - You use this transaction type to move funds from the PayFac Settlement Account to your Operating Account.

- **PayFac Debit Transaction** - You use this transaction type to move funds from your Operating Account to the PayFac Settlement Account.

- **Physical Check Credit** - You use this transaction type to move funds from the PayFac Settlement Account to the account of a third party that issues physical checks on your behalf.

- **Physical Check Debit** - You use this transaction type to move funds from the physical check issuer’s Account to the PayFac Settlement Account.

- **Reserve Credit Transaction** - You use this transaction type to move funds from the PayFac Settlement Account to the Reserve Account.

- **Reserve Debit Transaction** - You use this transaction type to move funds from the Reserve Account to the PayFac Settlement Account.

- **Sub-merchant Credit Transaction** - You use this transaction type to move funds from the PayFac Settlement Account to the Sub-merchant Account, funding the Sub-merchant.

- **Sub-merchant Debit Transaction** - You use this transaction type to move funds from the Sub-merchant Account to the PayFac Settlement Account.

- **Vendor Credit** - You use this transaction type to move funds from the PayFac Settlement Account to the account of a third party involved in the sale transactions, but are not the sub-merchant.

- **Vendor Debit** - You use this transaction type to move funds from the Vendor Account to the PayFac Settlement Account.
Testing Your cnpAPI Transactions

The information provided in this chapter enables you to test and verify that your submitted transaction data conforms to the required cnpAPI schema. This chapter contains the following topics:

- Certification and Testing Environments
- Overview of Testing
- Transferring Files
- Performing the Required Certification Tests
- Performing the Optional Tests

**IMPORTANT:** Per PCI DSS Requirements and Security Assessment Procedure, Section 6.4.3, "Production data (live PANs) are not used for testing or development."

**NOTE:** This chapter does not include Certification tests for the PayFac Instruction-Based Dynamic Payout transaction types. Additional information about these transactions, as well as the Certification tests are in Appendix D, "PayFac Dynamic Payout".
2.1 Certification and Testing Environments

Worldpay eComm has two testing and certification platforms optimized for different uses. These environments are called: Sandbox and Pre-Live. This section discusses these various environments, their uses, and limitations. The certification and testing environments do not have the full capacity, performance, or availability of the Worldpay eComm Production platform.

2.1.1 Sandbox Environment

The Sandbox environment is a simulator that provides a basic interface for functional level testing of transaction messages. Typically, merchants using one of the available Software Development Kits (SDKs) would use the Sandbox to test basic functionality, but it could also be used by any merchant using cnpAPI. This is a stateless simulator, so it has no historical transactional data, and does not provide full functionality. This environment is not intended for certification or regression testing.

NOTE: At his time, the Sandbox does not support Batch transactions (Online transactions only).

Use of the Sandbox does not require a password. The environment is available on the public internet (at www.testvantivcnp.com/sandbox/communicator/online). Supporting documentation, including test case documentation, is available at vantiv-ecommerce.github.io/sandbox.

2.1.2 Pre-Live Environment

The Pre-Live test environment is the platform used for all merchant Certification testing. This environment should be used by both newly on-boarding Worldpay merchants (for example, coding a direct XML integration for the first time), and existing Worldpay merchants seeking to incorporate additional features or functionalities into their current XML integrations. While the Pre-Live system provides a working version of the Worldpay production system, it does not have the full capacity or performance of the production platform and operates using simulators rather than communicating with the card associations.

The Pre-Live environment provides for the self-provisioning of a basic test account via www.testvantivcnp.com/prelivemid. Using this account, you can submit most standard transaction types, including those specified in the Certification test sections provided later in this chapter. You will not be able to test many of the Worldpay eCommerce Value Added Services (VAS) using this basic account. To add the capability to perform test scenarios for the VAS services, please contact a Worldpay Implementation Consultant. They will adjust the configuration of your test account to enable the additional, desired features.

2.1.2.1 Pre-Live Environment Limitations and Maintenance Schedules

When using the Pre-Live environment for testing, please keep in mind the following limitations and maintenance schedules:

- The number of merchants configured per organization will be limited to the number necessary to perform the required certification testing.
- Data retention is limited to a maximum of 30 days.
NOTE: Depending upon overall system capacity and/or system maintenance requirements, data purges may occur frequently. Whenever possible, we will provide advanced notification.

- Merchant profile and data deleted after 7 consecutive days with no activity.
- Maintenance window - each Tuesday and Thursday from 4:00-8:00 AM ET.
- Maintenance window (Enterprise eProtect) - every other Thursday from 10:00 PM to 6:00 AM ET.
- Daily limit of 1000 Online transactions.
- Daily limit of 10000 Batch transactions.

NOTE: Due to the planned maintenance windows, you should not use this environment for continuous regression testing.
2.2 Overview of Testing

The purpose of the testing and certification process is to verify that your order entry and supporting systems construct and send XML messages that comply with the cnpAPI requirements. The testing process involves submitting Worldpay supplied data for specific fields in a request, and receiving specific data back in a response. The response returned by Worldpay allows you to verify that you parse the cnpAPI response file correctly.

Various tables in this chapter provide the data you use while testing, including card numbers, expiration dates, transaction amounts, and other values as required by the testing process. You use this data as input to your systems resulting in structured requests that conform to the required cnpAPI schema.

**IMPORTANT:** The test data supplied does not necessarily account for all data fields/xml elements in a particular request. Where test data is not supplied, you should provide appropriate information. You should never override your own system to enter supplied data. If you are unable to enter the supplied data without overriding your system, please consult your Implementation Consultant concerning the test and how to proceed.

Typically, Worldpay assigns an Implementation Consultant to work with you after the completion of contract negotiations. Before you begin the testing phase, your assigned Implementation Consultant establishes a test account and supplies you with instruction for accessing the account along with the username and password. You must supply the IP address from which the data originates so we can grant access.

**NOTE:** Until you complete all required testing, you will only have access to the test and certification environment.

The testing process involves the following steps:

### 2.2.1 Planning for Certification Testing

Before you begin testing, determine which transaction types you will use according to your business needs. Virtually everyone will make use of the following basic transaction types: Authorization, Capture, Sale, Credit, and Void. There are several other transaction types and most transaction types offer many options/optional fields that you may wish to utilize and therefore test. For example, if you decide to offer Direct Debits as an alternate payment method, there are several associated certification tests required. Similarly, if you elect to make use of the Insights feature set, there are additional Authorization tests required for certification.
2.2.2 Required Certification Testing

Certification testing is a required phase of implementing the cnpAPI format. During the certification testing, a Worldpay Implementation Consultant works with you to verify that your transaction submissions meet the requirements of the cnpAPI specifications. Each transaction type has specific test scenarios that use specific data sets simulating real transactions. The Worldpay Certification environment responds to each submission with an XML response message, allowing you to verify that you have coded correctly to parse and store the transaction data returned to you. For more detailed information, see Performing the Required Certification Tests on page 91.

2.2.3 Optional Testing

Worldpay provides you with test data, test scenarios, a test environment, and credentials for using that test environment so you can perform these tests on your own keeping in mind the limitations of the certification environment (see Certification and Testing Environments on page 80). During unattended testing, you use these resources to perform all of the tests that apply to your business needs.

NOTE: If you have questions or need assistance while performing unattended testing, feel free to call your assigned Implementation Consultant or email implementation@vantiv.com.

2.2.4 System Doctor

The System Doctor is a tool you can use to both verify the operation of the Pre-Live environment and to verify operation and message structures of individual certification tests. Every 30 minutes the System Doctor runs each certification test specified in this chapter, except the Recycling and Recurring tests. You access the System Doctor through the Pre-Live iQ (Operations>System Doctor).
As shown in Figure 2-1, the system displays an overall assessment of the system status in the top panel and the results from each test set in the Test Execution History section (bottom panel). The center panel has a list of all individual tests by Order Id (pull-down), which you can filter by clicking on or more of the Search Tag buttons. Selecting one of the Order Ids from the list displays the XML request and response for that test (see Figure 2-2). You can also display the XML request and response information by selecting a test run from the Test Execution History and then selecting an individual test from the expanded list.

NOTE: The search tags filter the Order Ids shown in the selection pull-down, but do so in an ORed manner. For example, if you select the Visa Filter and the Prepaid Filter, the pull-down displays Order Ids for tests that either use a Visa card OR a Prepaid card, rather than tests for a Prepaid Visa card.
If one of your certification tests did not yield the expected results, you can use the System Doctor to determine if the test is performing as expected in the environment and if there is a discrepancy between your submitted XML and the XML message used in the automated test. If the test failed in the last automated run, you will know that there is a system issue and can contact your Implementation Consultant to determine when it will be resolved. If the test passed in the automated run, you can compare the automated XML message to your submission to determine why you are not receiving the expected results.
2.3 Transferring Files

As discussed in Communications Protocols on page 2, there are several protocols you can use to submit your transactions to Worldpay for processing. This section provides additional information concerning the recommended methods for transferring your cnpAPI Batch and Online transaction files.

2.3.1 Transferring Session Files

This section describes how to FTP your files (not test system specific) and includes the following topics:

- Submitting a Session File for Processing
- Retrieving Processed Session Files

**NOTE:** Before you begin transferring files via FTP, Worldpay provides the FTP Host and a username/password for the Worldpay test system.

2.3.1.1 Submitting a Session File for Processing

**CAUTION:** When submitting a file via FTP/sFTP, verify that the file permission is set to 664.

Also, file naming conventions are crucial to the file submission process. Incorrect file names will prevent the file from being processed or may stop processing due to an incomplete file transfer.

**NOTE:** Do not append .asc to the end of the filename (Step 3). You must replace the .prg extension with .asc. If .prg appears in the filename, the system will not process the file.

1. On your local system, add the extension .prg (lowercase) to the name of the file you want to submit. For example, you could name the file MerchantName_YYMMDD.prg. Keep in mind the following rules:
   - Spaces are not allowed in the file name
   - The .prg extension must be lower case
2. Open your FTP connection to the Worldpay inbound directory and move your file to the Worldpay directory.
3. After the FTP process completes, change the extension of the transmitted file (in the Worldpay inbound directory) from .prg to .asc (lowercase). Also, change the file permission to 664. The system polls the directory for files with an .asc extension every thirty seconds. When the system encounters files with the proper extension, it retrieves them for processing.

2.3.1.2 Retrieving Processed Session Files

Depending on the size of your file, your response should be ready within a few minutes. Batches containing large number of transactions take longer. For example, a batch of 10,000 transactions may require as long as ten minutes to process.
The initial response represents an acknowledgment that we received the transactions and notification that we will deliver them upstream to Visa and/or MasterCard for review. Since we perform validation operations against the credit card number and the expiration date, you may also receive a decline responses containing the appropriate response code.

To retrieve response files from the outbound directory:

**NOTE:** Worldpay removes response files from the outbound directory after 24 hours. Plan to retrieve your files daily.

1. Open your FTP connection to the Worldpay outbound directory.
2. Locate the response file, which will have the same name as the file you submitted. If the response file has a .prg extension, it is still transferring. The extension changes to .asc when the transfer to the outbound directory completes.

### 2.3.2 Transferring Online Files

The recommended method for submitting Online transactions is via HTTPS POST. The sections that follow provide examples of ASP and Java programming methods for submitting your data using HTTPS POST.

- ASP Programming Example
- Java Programming Example
- Helpful Web Sites

**NOTE:** Before you begin testing, Worldpay provides the test system URL, a username/password, and any additional information required to test your XML transactions.
2.3.2.1 **ASP Programming Example**

The following code is an example of a cnpAPI Authorization transaction submitted via HTTPS Post using ASP.

```javascript
Dim xml
Dim strXML
strXML = strXML & "<litleOnlineRequest version=""11.1"
xmlns=""http://www.litle.com/schema"" merchantId=""MERCHANTID"">"
strXML = strXML & "<authentication>"
strXML = strXML & "<user>USERNAME</user>"
strXML = strXML & "<password>########</password>"
strXML = strXML & "</authentication>"
strXML = strXML & "<authorization id=""834262"" reportGroup=""123"
customerId=""038945"">"
strXML = strXML & "<orderId>3235059</orderId>"
strXML = strXML & "<amount>54399</amount>"
strXML = strXML & "</authorization>"
strXML = strXML & "</litleOnlineRequest>"
```

```xml
<orderSource>ecommerce</orderSource>
<billToAddress>
  <name>Todd Wilson</name>
  <addressLine1>123 Blue Street</addressLine1>
  <addressLine2>Suite 108</addressLine2>
  <addressLine3>Address Line 3</addressLine3>
  <city>Lowell</city>
  <state>MA</state>
  <zip>01851</zip>
  <country>USA</country>
  <email>twilson@email.com</email>
  <phone>323-222-2222</phone>
</billToAddress>
<card>
  <type>VI</type>
  <number>###########</number>
  <expDate>0521</expDate>
  <cardValidationNum>###</cardValidationNum>
</card>"
```

strXML = strXML & "</billToAddress>"
strXML = strXML & "</card>"
strXML = strXML & "</litleOnlineRequest>"
```
set xml = CreateObject("Microsoft.XMLHTTP")
xml.setRequestHeader "Content-type", "text/html; charset=UTF-8"
xml.Open "POST", "https://site.info.com/from_Litle", False
xml.Send strXML
Response.write (xml.responseText)
set xml = Nothing

2.3.2.2 Java Programming Example

The following is an example of Java code used for HTTPS Post.

PostMethod and HttpClient are both part of the Apache HttpClient library located at http://jakarta.apache.org/commons/httpclient/.

PostMethod post = new PostMethod(url); // url = fully qualified url of the server to which you are posting
post.setRequestHeader("Content-type", "text/html; charset=UTF-8");
post.setRequestBody(data); //data = request data in XML format

HttpClient client = new HttpClient();
client.setTimeout(10000); //10 second timeout (in milliseconds) is suggested minimum, 30 second recommended for alternate payment methods
client.executeMethod(post);

String response = post.getResponseBodyAsString();

//if the server throws an exception you get a null response
//to get around this set it to ""
if (response == null) {
    response = "";
}
post.releaseConnection();

2.3.2.3 Notes on Timeout Settings

While Worldpay optimizes our systems to ensure the return of Authorization responses as quickly as possible, some portions of the process are beyond our control. The round-trip time of an Authorization can be broken down into three parts, as follows:

1. Transmission time (across the Internet) to Worldpay and back to the merchant
2. Processing time by the card association or authorization provider

3. Processing time Worldpay

Under normal operating circumstances, the transmission time to and from Worldpay does not exceed 0.6 seconds and processing time by Worldpay occurs in 0.1 seconds. Typically, the processing time by the card association or authorization provider can take between 0.5 and 3 seconds, but some percentage of transactions may take significantly longer.

Because the total processing time can vary due to a number of factors, Worldpay recommends using a timeout setting of 10 seconds minimum for card transactions and 30 seconds minimum for alternate payment methods. These settings should ensure that you do not frequently disconnect prior to receiving a valid authorization causing dropped orders and/or re-authorizations and duplicate authorizations.

**NOTE:** While it is uncommon, under certain circumstances network latency may cause a duplicate Online Sale transaction to go undetected as a duplicate. This can occur if you submit a second, duplicate Sale transaction while the response from the network for the Authorization portion of the first transaction is sufficiently delayed such that the first Sale has not been recorded as a valid transaction in the system.

If you elect to submit Online Sale transactions, Worldpay recommends a timeout setting of not less than 60 seconds to reduce the chances of undetected duplicate Sale transactions.

**2.3.2.4 Notes on Persistent Connections**

In order to provide a highly scalable service that meets the needs of high-throughput merchants, while reducing the number of idle connections that could result in some merchants exceeding their connection limits, Worldpay systems allow for 10 seconds of idle time before closing a connection. We selected this value because it is the midpoint between the Apache httpd 2.0 default value of 15 seconds and the Apache 2.2 default value of 5 seconds. Also, HTTP/1.1, which most modern HTTP client libraries use, employ persistent connections by default. When using a persistent connection, please keep the 10 second idle time limit in mind, when coding. If you do not use persistent connections, to avoid connection limit issues, please validate that your default configuration closes connections after each request.

**NOTE:** Proper use of persistent connections is considerably faster than opening and closing connections with each request.

**2.3.2.5 Helpful Web Sites**

The following web sites provide additional information, helpful hints, and examples of different programming methods used in combination with HTTPS POST.

- [http://p2p.wrox.com](http://p2p.wrox.com)
- [http://www.java-samples.com/java/POST-toHTTPS-url-free-java-sample-program.htm](http://www.java-samples.com/java/POST-toHTTPS-url-free-java-sample-program.htm)
2.4 Performing the Required Certification Tests

You are required to complete a number of certification tests prior to submitting real transactions to the Worldpay production system. This testing process allows you to verify that your system not only submits correctly formatted transaction data, but also correctly parses the data returned to you in the response messages. To facilitate the certification process, Worldpay has established a certification environment that simulates the production environment.

**IMPORTANT:** To determine the final status and response Code of a declined or duplicate transactions, please refer to the Declined Transaction Report in Worldpay eComm iQ. Once in the production environment, you can also obtain this daily report via Secure Scheduled Reports.

During certification testing, a Worldpay Implementation Consultant will guide you through each required test scenario. For each transaction type specific data is supplied that you must use in your cnpAPI transactions. Use of this data allows the validation of your transaction structure/syntax, as well as the return of a response file containing known data.

**NOTE:** The test data supplied does not account for all data fields/xml elements in a particular request. Where data is not supplied, you should provide appropriate information. You should never override your own system to enter supplied data. If you are unable to enter the supplied data without overriding your system, please consult your Implementation Consultant concerning the test and how to proceed.

Never use the supplied test data in the production environment.

2.4.1 Testing Authorization (including Indicators), AVS Only, Capture, Credit, Sale, and Void Transactions

**NOTE:** To test your handling of 2-series (BIN) number for MasterCard, you can substitute a 2-series test card number (see Appendix C, "Test Card Numbers") in any Auth/Sale test and you will receive a 000 - Approved Response Code in the response message.

Table 2-1 provides 26 data sets you use to test your construction of Authorization, AVS Only, Sale and Force Capture transactions, as well as your ability to parse the information contained in the XML response messages. You also use some of the approved authorizations as the basis for testing Capture and Credit transactions.

You do not necessarily have to perform all Authorization test. The tests you perform depend upon the Worldpay features you have elected to use. The tests are divided as follows:

- Order Ids 1 through 9 - used to test standard Authorization requests and responses. Also used for AVS Only test and Sale test. (Capture test use the little TxnId returned in the response messages.)
- Order Ids 10 through 13 - include if you plan to use Partial Authorization
• Order Ids 14 and 20 - include if you plan to use the PrepaidIndicator feature (see Prepaid Indicator on page 24)
• Order Ids 21 through 24 - include if you plan to use the Affluence Indicator feature (see Affluence Indicator on page 25)
• Order Id 25 - include if you plan to use the Issuer Country feature (see Issuer Country Indicator on page 25)

To test Authorization transactions:

1. Verify that your Authorization XML template is coded correctly (refer to Authorization Transactions on page 192.)
2. Submit authorization transactions using the data shown in the Supplied Data Elements of Order Ids 1 through 9 of Table 2-1.
3. Verify that your response values match those shown in Key Response Elements for Order Ids 1 through 9 as shown in Table 2-1.
4. If you wish to test AVS only transactions, re-submit Order Ids 1 through 5, 7, 8, and 9 (skip order 6), but substitute 000 for the amount. The AVS result returned will be the value shown in the Key Response Elements section.

5. If you plan to use Partial Authorizations, submit authorization transactions using the data shown in the Supplied Data Elements of Order Ids 10 through 13 of Table 2-1.
6. Verify that your response values match those shown in Key Response Elements for Order Ids 10 through 13 as shown in Table 2-1.
7. If you elected to receive Prepaid Indicators, submit authorization transactions using the data shown in the Supplied Data Elements of Order Ids 14 through 20 of Table 2-1. Verify that your response values match those shown in Key Response Elements for Order Ids 14 and 15 as shown in Table 2-1.
8. If you elected to receive Affluence Indicators, submit authorization transactions using the data shown in the Supplied Data Elements of Order Ids 21 through 24 of Table 2-1. Verify that your response values match those shown in Key Response Elements for Order Ids 16 through 19 as shown in Table 2-1.
9. If you elected to receive Issuer Country information, submit an authorization transaction using the data shown in the Supplied Data Elements of Order Id 25 of Table 2-1. Verify that your response values match those shown in Key Response Elements for Order Id 25 as shown in Table 2-1.
10. If you plan to handle transactions using Healthcare (IIAS) cards, submit authorization transactions using the data shown in the Supplied Data Elements of Order Ids 26 through 31A of Table 2-1. Verify that your response values match those shown in Key Response Elements for Order Ids 26 through 31 as shown in Table 2-1.

To Test Sale transactions:

1. Verify that your Sale XML template is coded correctly (refer to Sale Transactions on page 292.)
2. Submit sale transactions using the data shown in the Supplied Data Elements of Order Ids 1 through 9 of Table 2-1.

NOTE: Some Issuers do not return an Auth Code for $0 Authorizations. You should code your systems to handle this possibility.
3. Verify that your response values match those shown in Key Response Elements for Order Ids 1 through 9 as shown in Table 2-1.

To Test Capture transactions:
1. Verify that your Capture XML template is coded correctly (refer to Capture Transactions on page 214.).
2. Submit capture transactions for Order Ids 1A through 5A using the littleTnxId value returned in the response messages for Authorization Order Ids 1 through 5.
3. Verify that your response values match those shown in Key Response Elements for Order Ids 1A through 5A as shown in Table 2-1.

To test Credit transactions:
1. Verify that your Credit XML template is coded correctly (refer to Credit Transactions on page 229.)
2. Submit credit transactions for Order Ids 1B through 5B using the littleTnxId value returned in the response messages for Capture Order Ids 1A through 5A.
3. Verify that your response values match those shown in Key Response Elements for Order Ids 1B through 5B as shown in Table 2-1.

To test Void transactions (in this test you have the option of voiding either Credit or Sale transactions):
1. Verify that your Void XML template is coded correctly (refer to Void Transactions (Online Only) on page 308.)
2. Submit void transactions for Order Ids 1C through 5C (and 6B if voiding Sale transactions) using the littleTnxId value returned in either the response messages for Credit Order Ids 1B through 5B, or the response messages for Sale (not Auth) Order Id 1 through 6.
3. Verify that your response values match those shown in Key Response Elements for Order Ids 1C through 5C (include 6B only if you elect to void the Sales transactions) as shown in Table 2-1.

**TABLE 2-1  Authorization Test Data**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 10100</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; John &amp; Mary Smith</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 1 Main St.</td>
<td>&lt;authCode&gt; 11111</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; Burlington</td>
<td>&lt;avsResult&gt; 01</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; MA</td>
<td>&lt;cardValidationResult&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 01803-3747</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 445701000000009</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0121</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 349</td>
<td></td>
</tr>
</tbody>
</table>
### Testing Your cnpAPI Transactions

**TABLE 2-1** Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A</td>
<td>Capture:</td>
<td>Capture Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>Value returned in Auth response for Order Id 1</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td>1B</td>
<td>Credit:</td>
<td>Credit Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>Value returned in Capture response for Order Id 1A</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td>1C</td>
<td>Void:</td>
<td>Void Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>Use either the value returned in the Credit response for Order Id 1B, or the value returned in the Sale response (not Auth) for Order Id 1.</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td>2</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 10100</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Mike J. Hammer</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 2 Main St.</td>
<td>&lt;authCode&gt; 22222</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine2&gt; Apt. 222</td>
<td>&lt;avsResult&gt; 10</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; Riverside</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; RI</td>
<td>&lt;authenticationResult&gt; Note: Not returned for MasterCard</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 02915</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 511201000000003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0221</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 261</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;authenticationValue&gt; BwABBJQ1AgAAA AAgJDUCAAAAAA A=</td>
<td></td>
</tr>
<tr>
<td>2A</td>
<td>Capture:</td>
<td>Capture Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>Value returned in Auth response for Order Id 2</td>
<td>&lt;message&gt; Approved</td>
</tr>
</tbody>
</table>
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>2B</td>
<td>Credit:</td>
<td>&lt;litleTxnId&gt; Value returned in Capture response for Order Id 2A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2C</td>
<td>Void:</td>
<td>&lt;litleTxnId&gt; Use either the value returned in the Credit response for Order Id 2B, or the value returned in the Sale response (not Auth) for Order Id 2.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Authorization/Sale/AVS:</td>
<td>&lt;amount&gt; 10100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;name&gt; Eileen Jones</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;addressLine1&gt; 3 Main St.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;city&gt; Bloomfield</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;state&gt; CT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;zip&gt; 06002</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;country&gt; US</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;type&gt; DI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;number&gt; 60110100000000003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;expDate&gt; 0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;cardValidationNum&gt; 758</td>
</tr>
<tr>
<td>3A</td>
<td>Capture:</td>
<td>&lt;litleTxnId&gt; Value returned in Auth response for Order Id 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3B</td>
<td>Credit:</td>
<td>&lt;litleTxnId&gt; Value returned in Capture response for Order Id 3A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>3C</td>
<td>Void: &lt;litleTxnId&gt; Use either the value returned in the Credit response for Order Id 3B, or the value returned in the Sale response (not Auth) for Order Id 3.</td>
<td>Void Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>4</td>
<td>Authorization/Sale/AVS: &lt;amount&gt; 10100 &lt;name&gt; Bob Black &lt;addressLine1&gt; 4 Main St. &lt;city&gt; Laurel &lt;state&gt; MD &lt;zip&gt; 20708 &lt;country&gt; US &lt;type&gt; AX &lt;number&gt; 375001000000005 &lt;expDate&gt; 0421</td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved &lt;authCode&gt; 44444 &lt;avsResult&gt; 13</td>
</tr>
<tr>
<td>4A</td>
<td>Capture: &lt;litleTxnId&gt; Value returned in Auth response for Order Id 4</td>
<td>Capture Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>4B</td>
<td>Credit: &lt;litleTxnId&gt; Value returned in Capture response for Order Id 4A</td>
<td>Credit Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>4C</td>
<td>Void: &lt;litleTxnId&gt; Use either the value returned in the Credit response for Order Id 4B, or the value returned in the Sale response (not Auth) for Order Id 4.</td>
<td>Void Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
</tbody>
</table>
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Authorization/Sale/AVS:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>4100200300011001</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>0521</td>
<td>&lt;authCode&gt; 55555</td>
</tr>
<tr>
<td></td>
<td>463</td>
<td>&lt;avsResult&gt; 32</td>
</tr>
<tr>
<td></td>
<td>BwABBJQ1AgAAA</td>
<td>&lt;cardValidationResult&gt;M</td>
</tr>
<tr>
<td></td>
<td>A=</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Capture:</td>
<td></td>
</tr>
<tr>
<td>5A</td>
<td>&lt;litleTxnId&gt;</td>
<td>Value returned in Auth response for Order Id 5</td>
</tr>
<tr>
<td>5B</td>
<td>Credit:</td>
<td></td>
</tr>
<tr>
<td>5C</td>
<td>Void:</td>
<td></td>
</tr>
</tbody>
</table>

**Table Example:**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>4100200300011001</td>
<td>&lt;authCode&gt; 55555</td>
</tr>
<tr>
<td></td>
<td>0521</td>
<td>&lt;avsResult&gt; 32</td>
</tr>
<tr>
<td></td>
<td>463</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>BwABBJQ1AgAAA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A=</td>
<td></td>
</tr>
</tbody>
</table>

**Example Output:**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>4100200300011001</td>
<td>&lt;authCode&gt; 55555</td>
</tr>
<tr>
<td></td>
<td>0521</td>
<td>&lt;avsResult&gt; 32</td>
</tr>
<tr>
<td></td>
<td>463</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>BwABBJQ1AgAAA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A=</td>
<td></td>
</tr>
</tbody>
</table>

**Example Output:**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>4100200300011001</td>
<td>&lt;authCode&gt; 55555</td>
</tr>
<tr>
<td></td>
<td>0521</td>
<td>&lt;avsResult&gt; 32</td>
</tr>
<tr>
<td></td>
<td>463</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>BwABBJQ1AgAAA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>A=</td>
<td></td>
</tr>
<tr>
<td>orderId</td>
<td>SUPPLIED DATA ELEMENTS</td>
<td>KEY RESPONSE ELEMENTS</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>6</td>
<td>Authorization/Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 10100</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Joe Green</td>
<td>&lt;response&gt; 110</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 6 Main St.</td>
<td>&lt;message&gt; Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; Derry</td>
<td>&lt;avsResult&gt; 34</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; NH</td>
<td>&lt;cardValidationResult&gt; P</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 03038</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457010100000008</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0621</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 992</td>
<td></td>
</tr>
</tbody>
</table>

| 6A      | Void:                  |                       |
|         | <littleTxnId>          | Void Response:        |
|         | Use the value returned in the Sale response (not Auth) for Order Id 6. | <response> 000 |
|         |                        | <message> Approved    |

| 7       | Authorization/Sale/AVS: | Authorization Response: |
|         | <amount> 10100         | <response> 301         |
|         | <name> Jane Murray     | <message> Invalid Account Number |
|         | <addressLine1> 7 Main St. | <avsResult> 34 |
|         | <city> Amesbury        | <cardValidationResult> N |
|         | <state> MA            |                       |
|         | <zip> 01913           |                       |
|         | <country> US          |                       |
|         | <type> MC             |                       |
|         | <number> 5112010100000002 |                       |
|         | <expDate> 0721        |                       |
|         | <cardValidationNum> 251 |                       |
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Authorization/Sale/AVS:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>Mark Johnson</td>
<td>&lt;response&gt; 123</td>
</tr>
<tr>
<td></td>
<td>8 Main St.</td>
<td>&lt;message&gt; Call Discover</td>
</tr>
<tr>
<td></td>
<td>Manchester</td>
<td>&lt;avsResult&gt; 34</td>
</tr>
<tr>
<td></td>
<td>NH</td>
<td>&lt;cardValidationResult&gt;P</td>
</tr>
<tr>
<td></td>
<td>03101</td>
<td></td>
</tr>
<tr>
<td></td>
<td>US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6011010100000002</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0821</td>
<td></td>
</tr>
<tr>
<td></td>
<td>184</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Authorization/Sale/AVS:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>10100</td>
<td>&lt;response&gt; 303</td>
</tr>
<tr>
<td></td>
<td>James Miller</td>
<td>&lt;message&gt; Pick Up Card</td>
</tr>
<tr>
<td></td>
<td>9 Main St.</td>
<td>&lt;avsResult&gt; 34</td>
</tr>
<tr>
<td></td>
<td>Boston</td>
<td>&lt;cardValidationResult&gt;P</td>
</tr>
<tr>
<td></td>
<td>MA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>02134</td>
<td></td>
</tr>
<tr>
<td></td>
<td>US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AX</td>
<td></td>
</tr>
<tr>
<td></td>
<td>375001010000003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0921</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0421</td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** The data sets for orderId 10 through 13 are designed to test Authorization transactions resulting in partial authorizations. If you are not coding to use partial authorizations, you may skip these tests.

10 | 40000 | 010 | Partially Approved |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4457010140000141</td>
<td>32000</td>
<td>0921</td>
<td>true</td>
</tr>
</tbody>
</table>
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>11</td>
<td>&lt;amount&gt;</td>
<td>60000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112010140000004</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td>12</td>
<td>&lt;amount&gt;</td>
<td>50000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>3750010140000009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0421</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td>13</td>
<td>&lt;amount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>DI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>6011010140000004</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0821</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
</tbody>
</table>

**NOTE:** The data sets for orderId 14 through 20 are designed to test Authorization transactions that return Prepaid Indicator information in the response message. If you are not coding to use the optional Prepaid Indicator feature of the Insights feature set, you may skip these tests.

| 14      | <amount> | 10100 | <response> | 000 |
|         | <type> | VI | <message> | Approved |
|         | <number> | 445701020000247 | <type> | PREPAID |
|         | <expDate> | 0821 | <availableBalance> | 2000 |
|         |         |     | <reloadable> | NO |
|         |         |     | <prepaidCardType> | GIFT |
| 15      | <amount> | 10100 | <response> | 000 |
|         | <type> | MC | <message> | Approved |
|         | <number> | 5500000254444445 | <type> | PREPAID |
|         | <expDate> | 0321 | <availableBalance> | 2000 |
|         |         |     | <reloadable> | YES |
|         |         |     | <prepaidCardType> | PAYROLL |
### Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Element</strong></td>
<td><strong>Value</strong></td>
</tr>
<tr>
<td>16</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5592106621450897</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5590409551104142</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5587755665222179</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5445840176552850</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5390016478904678</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0321</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**NOTE:** The data sets for orderId 21 through 24 are designed to test Authorization transactions that return Affluence Indicator information in the response message. If you are not coding to use the optional Affluence Indicator feature of the Insights feature set, you may skip these tests.
### TABLE 2-1  Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>21</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300012009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0921</td>
</tr>
<tr>
<td>22</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300013007</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td>23</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112010201000109</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0421</td>
</tr>
<tr>
<td>24</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112010202000108</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0821</td>
</tr>
</tbody>
</table>

**NOTE:** The data set for orderId 25 is designed to test Authorization transactions that return Issuer Country information in the response message. If you are not coding to use the optional Issuer Country feature of the Insights feature set, you may skip these tests.

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200310000002</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
</tbody>
</table>

**NOTE:** The data sets for orderId 26 through 31A are designed to test Authorization transactions involving Healthcare Care (IIAS) transaction. If you are not coding to perform Healthcare Care (IIAS) transactions, you may skip these tests.

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>&lt;amount&gt;</td>
<td>18698</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5194560012341234</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>20000</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td></td>
</tr>
<tr>
<td>orderId</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>27</td>
<td>&lt;amount&gt;</td>
<td>18698</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5194560012341234</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>16000</td>
</tr>
<tr>
<td>28</td>
<td>&lt;amount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5194560012341234</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>3698</td>
</tr>
<tr>
<td>29</td>
<td>&lt;amount&gt;</td>
<td>18699</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4024720001231239</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>31000</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>20910</td>
</tr>
<tr>
<td></td>
<td>&lt;visionAmount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;clinicOtherAmount&gt;</td>
<td>9050</td>
</tr>
<tr>
<td></td>
<td>&lt;dentalAmount&gt;</td>
<td>1049</td>
</tr>
<tr>
<td>30</td>
<td>&lt;amount&gt;</td>
<td>20000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4024720001231239</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>20000</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>20901</td>
</tr>
<tr>
<td></td>
<td>&lt;visionAmount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;clinicOtherAmount&gt;</td>
<td>9050</td>
</tr>
<tr>
<td></td>
<td>&lt;dentalAmount&gt;</td>
<td>1049</td>
</tr>
</tbody>
</table>
2.4.2 Testing Authorization Reversal Transactions

If you plan to use Authorization Reversal Transactions, you must perform this test. If you do not plan to use Authorization Reversal transactions, skip this test and go to Testing Direct Debit Transactions on page 108.

To test Authorization Reversal Transactions:

1. Verify that your Authorization Reversal XML templates are coded correctly (refer to Authorization Reversal Transactions on page 204).
2. Submit the Authorizations, Captures (if applicable), and Authorization Reversal Transactions using the orders shown in Table 2-2.
3. Verify that your response values match those shown in Key Response Elements as shown in Table 2-2.

### TABLE 2-1 Authorization Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>31</td>
<td>&lt;amount&gt;</td>
<td>40000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4024720001231239</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>18699</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;visionAmount&gt;</td>
<td>15099</td>
</tr>
<tr>
<td></td>
<td>&lt;clinicOtherAmount&gt;</td>
<td>17699</td>
</tr>
<tr>
<td>31A</td>
<td>&lt;amount&gt;</td>
<td>6000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4024720001231239</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>&lt;totalHealthcareAmount&gt;</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>&lt;RxAmount&gt;</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>&lt;visionAmount&gt;</td>
<td>1000</td>
</tr>
</tbody>
</table>
## TABLE 2-2  Authorization Reversal Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>Authorization:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>10010</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John Smith</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>1 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>Burlington</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>MA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>01803-3747</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457010000000009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0121</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>349</td>
</tr>
<tr>
<td>32A</td>
<td>Capture:</td>
<td>Capture Response:</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5050</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value returned in Auth response for Order Id 32</td>
</tr>
<tr>
<td>32B</td>
<td>Authorization Reversal:</td>
<td>Auth Reversal Response:</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value returned in Auth response for Order Id 32</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Do Not Submit an Amount</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** This transaction returns 111 instead of 000, because it is unnecessary to submit an Authorization Reversal for the Visa payment card.

Declined Transaction report result = 111 - Authorization amount has already been depleted
### TABLE 2-2  Authorization Reversal Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>33</td>
<td>Authorization:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 20020</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Mike J. Hammer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 2 Main St.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine2&gt; Apt. 222</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; Riverside</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; RI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 02915</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112010000000003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0221</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 261</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;authenticationValue&gt;</td>
<td>BwABBJQ1AgAAA AAgJDUCAAAAAA A=</td>
</tr>
<tr>
<td>33A</td>
<td>Authorization Reversal:</td>
<td>Value returned in Auth response for Order Id 33</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; Do Not Submit an amount</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>Authorization:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 30030</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Eileen Jones</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 3 Main St.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; Bloomfield</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; CT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 06002</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 6011010000000003</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0321</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 758</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-2  Authorization Reversal Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>34A</td>
<td>Authorization Reversal:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value returned in Auth response for Order Id 34</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>Do Not Submit an Amount</td>
</tr>
<tr>
<td>35</td>
<td>Authorization:</td>
<td>10100 Bob Black 4 Main St. Laurel MD 20708 US AX 375001000000005 0421</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Bob Black</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>4 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>Laurel</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>MD</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>20708</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>375001000000005</td>
</tr>
<tr>
<td>35A</td>
<td>Capture:</td>
<td>5050 Order Id 35</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5050</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value returned in Auth response for Order Id 35</td>
</tr>
<tr>
<td>35B</td>
<td>Authorization Reversal:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value returned in Auth response for Order Id 35</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5050</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Declined Transaction report result = 336 - Reversal amount does not match Authorization amount</td>
</tr>
<tr>
<td>36</td>
<td>Authorization:</td>
<td>20500 AX 375000026600004 0521</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>20500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>375000026600004</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0521</td>
</tr>
</tbody>
</table>
2.4.3 Testing Direct Debit Transactions

If you have elected to offer Direct Debits as an alternate payment method, you are required to complete the tests in this section. Data sets are provided for you to use to test your construction of XML request messages, as well as the parsing of the response messages for Direct Debit transactions.

**NOTE:** The eCheck Verification test is required only if you plan to perform eCheck Verifications.

To test eCheck Verification transactions:

1. Verify that your XML template is coded correctly (refer to eCheck Verification Transactions on page 258.)
2. Submit the eCheck Verification transactions using the data sets supplied in Table 2-3.

**NOTE:** In addition to the test data provided in the table, you must also provide appropriate data for other child elements of the billToAddress element, such as Address1 (2, 3), city, state, phone, etc.

For Corporate accounts (Order ID 39 and 40) you must include the firstName, lastName, and companyName in the echeckVerification request.

3. Verify that your response values match those shown in the Key Response Elements section of Table 2-3.
4. After you complete this test, go to the next Direct Debit test.

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>36A</td>
<td>Authorization Reversal:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Auth Reversal Response:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>10000</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>Value returned in Auth</td>
<td>Declined Transaction</td>
</tr>
<tr>
<td></td>
<td>response for Order Id 36</td>
<td>result = 336 - Reversal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>amount does not match</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization amount</td>
</tr>
</tbody>
</table>

TABLE 2-2 Authorization Reversal Test Data
<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>37</td>
<td>&lt;amount&gt;</td>
<td>3001</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Tom</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Black</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>10@BC99999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>053100300</td>
</tr>
<tr>
<td>38</td>
<td>&lt;amount&gt;</td>
<td>3002</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>John</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Smith</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1099999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
<tr>
<td>39</td>
<td>&lt;amount&gt;</td>
<td>3003</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Robert</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Jones</td>
</tr>
<tr>
<td></td>
<td>&lt;companyName&gt;</td>
<td>Good Goods Inc</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>309999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>053100300</td>
</tr>
<tr>
<td>40</td>
<td>&lt;amount&gt;</td>
<td>3004</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Peter</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Green</td>
</tr>
<tr>
<td></td>
<td>&lt;companyName&gt;</td>
<td>Green Co</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>80999999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
</tbody>
</table>
To test eCheck Prenotification transactions:

**NOTE:** The eCheck Prenotification tests are required only if you plan to perform eCheck Prenotification.

You can only submit eCheck Prenotification transactions in Batches.

1. Verify that your XML templates are coded correctly (refer to eCheck Prenotification Credit Transactions (Batch Only) on page 247 and eCheck Prenotification Sale Transactions (Batch Only) on page 250.)

2. Submit the eCheck Prenotification transactions using the data sets supplied in Table 2-4.

3. Verify that your response values match those shown in the Key Response Elements section of Table 2-4.

4. After you complete this test, go to the next Direct Debit test.

### TABLE 2-4 eCheck Prenotification Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>ECPreNoteSale</td>
<td>&lt;orderSource&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>PreNote Sale Corp</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1092969901</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
<tr>
<td>ECPreNoteCredit</td>
<td>&lt;orderSource&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>PreNote Credit Corp</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1099339999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
<tr>
<td>PreNoteSaleAcc</td>
<td>&lt;orderSource&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td>NumErr</td>
<td>&lt;name&gt;</td>
<td>PreNote Sale Corp</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>10@2969901</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>0111000012</td>
</tr>
</tbody>
</table>
To test eCheck Sale transactions:

1. Verify that your XML template is coded correctly (refer to eCheck Sale Transactions on page 255).
2. Submit the echeckSale transactions using the Supplied Data elements shown in Table 2-5.

   NOTE: In addition to the test data provided in the table, you must also provide appropriate data for other child elements of the billToAddress element, such as Address1, city, state, phone, etc.

3. Verify that your response values match those shown in Table 2-5.
4. After you complete this test, go to the next test.
### TABLE 2-5  eCheck Sale Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>41</td>
<td>&lt;amount&gt;</td>
<td>2008</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Mike</td>
</tr>
<tr>
<td></td>
<td>&lt;middleInitial&gt;</td>
<td>J</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Hammer</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>10@BC99999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>053100300</td>
</tr>
<tr>
<td>42</td>
<td>&lt;amount&gt;</td>
<td>2004</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Tom</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Black</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>4099999992</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
<tr>
<td>43</td>
<td>&lt;amount&gt;</td>
<td>2007</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Peter</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Green</td>
</tr>
<tr>
<td></td>
<td>&lt;companyName&gt;</td>
<td>Green Co</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>6099999992</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
<tr>
<td>44</td>
<td>&lt;amount&gt;</td>
<td>2009</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Peter</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Green</td>
</tr>
<tr>
<td></td>
<td>&lt;companyName&gt;</td>
<td>Green Co</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>9099999992</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>053133052</td>
</tr>
</tbody>
</table>
To test **eCheck Credit** transactions:

1. Verify that your XML template is coded correctly (refer to **eCheck Credit Transactions** on page 244.)

   **NOTE:** In addition to the test data provided in the table, you must also provide appropriate data for other child elements of the **billToAddress** element, such as **Address1 (2, 3), city, state, phone, etc.**

2. Submit the **echeckCredit** transactions using the data in the Supplied Data Elements section of **Table 2-6**.

3. Verify that your response values match those shown in the Key Response Elements section of **Table 2-6**.

4. After you complete this test, go to the next test.

**TABLE 2-6  eCheck Credit Test Data**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>45</td>
<td>&lt;amount&gt;</td>
<td>1001</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>John</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Smith</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>10@BC99999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>053100300</td>
</tr>
<tr>
<td>46</td>
<td>&lt;amount&gt;</td>
<td>1003</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>telephone</td>
</tr>
<tr>
<td></td>
<td>&lt;firstName&gt;</td>
<td>Robert</td>
</tr>
<tr>
<td></td>
<td>&lt;lastName&gt;</td>
<td>Jones</td>
</tr>
<tr>
<td></td>
<td>&lt;.companyName&gt;</td>
<td>Widget Inc</td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Corporate</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>30999999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011075150</td>
</tr>
</tbody>
</table>
To test eCheck Void transactions:

1. Verify that your XML template is coded correctly (refer to eCheck Void Transactions (Online Only) on page 262.)

2. (Re)Submit the echeckSale transaction for Order ID #42 as shown in Table 2-5 with a different value for the id attribute.
   a. Using the litleTxnId returned in the echeckSaleResponse message, submit an echeckVoid transaction.
   b. The system returns an echeckVoidResponse Response Code of 000 and a message of Approved.

3. Using the litleTxnId returned in the echeckCreditResponse message for Order ID #46, submit an echeckVoid transaction.
   a. The system returns an echeckVoidResponse Response Code of 000 and a message of Approved.

4. Submit an echeckVoid request using a value of “2” for the litleTxnId.
   a. The system returns a Response Code of 000 and a message of Approved; however, the Declined Transaction Report will contain a Response Code of 360 and a message of No transaction found with specified litleTxnId for this transaction.

5. After you complete this test, go to the next test.
2.4.4 Testing Token Transactions

You can obtain tokens in two ways. The first method is explicit registration using the registerTokenRequest transaction. The second method is implicit registration, which is achieved by submitting the card or account information (for Direct Debits) in a normal payment transaction. This section provides test data sets using both methods for both credit card and Direct Debit tokenization.

Perform this test only if you plan to use the Vault feature.

**NOTE:** The test data does not include values for all elements. You should use appropriate values for all elements as required to create a properly structured cnpAPI request.

To test explicit Token Registration transactions:
1. Verify that your cnpAPI template is coded correctly for this transaction type (refer to registerTokenRequest on page 667.)
2. To test credit card tokenization, submit registerTokenRequest transactions using the data for Order Ids 50 through 52 from Table 2-7.
3. If you also use Direct Debit transactions and have elected to tokenize Direct Debit account numbers, submit registerTokenRequest transactions using the data for Order Ids 53 and 54 from Table 2-7; otherwise, skip those two tests.
4. Verify that your registerTokenResponse values match those shown in the Key Response Elements section of Table 2-7. The complete token values are not defined in the table, because the system generates the tokens dynamically.
5. After completing this test, proceed to the next set of tests for implicit tokenization.

### TABLE 2-7 Register Token Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>&lt;accountNumber&gt; 4457119922390123</td>
<td>&lt;littleToken&gt; xxxxxxxxx0123&lt;br&gt; &lt;bin&gt; 445711&lt;br&gt; &lt;type&gt; VI&lt;br&gt; &lt;response&gt; 801&lt;br&gt; &lt;message&gt; Account number was successfully registered</td>
</tr>
<tr>
<td>51</td>
<td>&lt;accountNumber&gt; 4457119999999999</td>
<td>none returned&lt;br&gt; &lt;response&gt; 820&lt;br&gt; &lt;message&gt; Credit card number was invalid</td>
</tr>
</tbody>
</table>
To test the submission of card data in an tokenized environment using Authorization transactions, as well as the submission of tokens in transactions, do the following:

1. Verify that your cnpAPI template is coded correctly for this transaction type (refer to Authorization Transactions on page 192.)
2. Submit three authorization transactions using the Supplied Data Elements from Order Ids 55 through 57 from Table 2-8.
3. Verify that your authorizationResponse values match those shown in the Key Response Elements section of Table 2-8 for Order Ids 55 through 57.
4. To verify that your cnpAPI template is coded correctly for the submission of tokens in authorization transactions, submit authorization transactions using the Supplied Data Elements from Order Ids 58 through 60 from Table 2-8.

To test the submission of Direct Debit data in an tokenized environment, as well as the submission of tokens in Direct Debit transactions, do the following:

1. Verify that your cnpAPI template is coded correctly for these transaction types as applicable (refer to eCheck Sale Transactions on page 255 and eCheck Credit Transactions on page 244).
2. Submit the four transactions, Order Ids 61 through 64, using the Supplied Data Elements from Table 2-8.
3. Verify that your response values match those shown in the Key Response Elements section of Table 2-8 for Order Ids 61 through 64.

---

**TABLE 2-7  Register Token Test Data**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>52</td>
<td>&lt;accountNumber&gt;</td>
<td>4457119922390123</td>
</tr>
<tr>
<td></td>
<td>&lt;bin&gt;</td>
<td>445711</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt;</td>
<td>802</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt;</td>
<td>Account number was previously registered</td>
</tr>
<tr>
<td>53</td>
<td>&lt;accNum&gt;</td>
<td>1099999998</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011100012</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>EC</td>
</tr>
<tr>
<td></td>
<td>&lt;eCheckAccountSuffix&gt;</td>
<td>998</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt;</td>
<td>801</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt;</td>
<td>Account number was successfully registered</td>
</tr>
<tr>
<td>54</td>
<td>&lt;accNum&gt;</td>
<td>1022222102</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>1145_7895</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt;</td>
<td>Invalid bank routing number</td>
</tr>
<tr>
<td>orderId</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>55</td>
<td>&lt;amount&gt; 15000 &lt;type&gt; MC &lt;number&gt; 5435101234510196 &lt;expDate&gt; 1121 &lt;cardValidationNum&gt; 987</td>
<td>&lt;response&gt; 000 &lt;message&gt; Approved &lt;litleToken&gt; xxxxxxxxxxx0196 &lt;tokenResponseCode&gt; 801 &lt;tokenMessage&gt; Account number was successfully registered &lt;type&gt; MC &lt;bin&gt; 543510</td>
</tr>
<tr>
<td>56</td>
<td>&lt;amount&gt; 15000 &lt;type&gt; MC &lt;number&gt; 543510999999999 &lt;expDate&gt; 1121 &lt;cardValidationNum&gt; 987</td>
<td>&lt;response&gt; 301 &lt;message&gt; Invalid account number</td>
</tr>
<tr>
<td>57</td>
<td>&lt;amount&gt; 15000 &lt;type&gt; MC &lt;number&gt; 5435101234510196 &lt;expDate&gt; 1121 &lt;cardValidationNum&gt; 987</td>
<td>&lt;response&gt; 000 &lt;message&gt; Approved &lt;litleToken&gt; xxxxxxxxxxx0196 &lt;tokenResponseCode&gt; 802 &lt;tokenMessage&gt; Account number was previously registered &lt;type&gt; MC &lt;bin&gt; 543510</td>
</tr>
<tr>
<td>58</td>
<td>&lt;amount&gt; 15000 &lt;litleToken&gt; xxxxxxxxxxx0196 &lt;expDate&gt; 1121 &lt;cardValidationNum&gt; 987</td>
<td>&lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>59</td>
<td>&lt;amount&gt; 15000 &lt;litleToken&gt; 1111000100092332 &lt;expDate&gt; 1121</td>
<td>&lt;response&gt; 822 &lt;message&gt; Token was not found</td>
</tr>
</tbody>
</table>

*Note:* Use the token returned in Order Id 57.
### TABLE 2-8  Implicit Registration Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>60</td>
<td>&lt;amount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;littleToken&gt;</td>
<td>1112000100000085</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td>61</td>
<td>eCheck Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1099999903</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011100012</td>
</tr>
<tr>
<td></td>
<td>&lt;litleToken&gt;</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>eCheck Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1099999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011100012</td>
</tr>
<tr>
<td></td>
<td>&lt;litleToken&gt;</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>eCheck Credit:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;accType&gt;</td>
<td>Checking</td>
</tr>
<tr>
<td></td>
<td>&lt;accNum&gt;</td>
<td>1099999999</td>
</tr>
<tr>
<td></td>
<td>&lt;routingNum&gt;</td>
<td>011100012</td>
</tr>
<tr>
<td></td>
<td>&lt;litleToken&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### 2.4.5 Testing Query Transactions

You can use the following test scenarios to verify your coding of the `queryTransaction`, as well as various responses. To test the query transaction, please do the following:

**NOTE:** If you are enabled for the use of the `queryTransaction`, you will not be able to test for response code 153 - Query Transaction not enabled.

1. Submit a `queryTransaction` with the following elements:
   - `<origId>newTransaction</origId>` *(Note: You can use any value never used as an id attribute.)*
   - `<origActionType>A</origActionType>` *(Note: You can use any valid action type as detailed in the enumerations table of `origActionType` on page 599.)*

   Verify that you receive a response value of **151** and message of **Original transaction not found**.

2. Submit an authorization or sale transaction using a unique id attribute.

---

**TABLE 2-8** Implicit Registration Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
</table>
| 64      | eCheck Sale:  
  - `<accType>` Corporate  
  - `<accNum>` 6099999993  
  - `<routingNum>` 211370545  
|         |                        | <originalTokenInfo>  
  - `<accType>` Checking  
  - `<litleToken>` 1119000001003001  
  - `<routingNum>` 211370545  
  - `<newTokenInfo>`  
    - `<accType>` Checking  
    - `<litleToken>` 1119000000154101  
    - `<routingNum>` 211370545  
    - `<litleToken>` xxxxxxxxxxxxxxxx  
  - `<tokenResponseCode>` 801  
  - `<tokenMessage>` Account number was successfully registered  
  - `<type>` EC  
  - `<eCheckAccountSuffix>` 993  

**NOTE:** Order ID 64 returns accountUpdater information. This test allows you to test responses you might receive when a NOC exists against the Direct Debit account, but you submit the old account information. In this case, the system provides the old token information, but issues a new token based upon the new account information and provides it as well.
3. Submit a queryTransaction using the id attribute value from Step 2 as the value for the <origId> element and the <origActionType> element set to the transaction type you submitted in Step 2.
   Verify that you receive a response value of 150 and message of Original transaction found, a <matchCount> value of 1, and the response message for the transaction submitted in Step 2, as a child of the <result_Max10> element.

4. Submit a second authorization or sale transaction (use the same transaction type as Step 2), using the same value for the id attribute. Note: If you use a sale transaction for this test, you must change the credit card number from the one you used in Step 2 to avoid having the transaction flagged as a duplicate.

5. Submit a queryTransaction using the id attribute value from Step 2 as the value for the <origId> element and the <origActionType> element set to the transaction type you submitted in Step 2.
   Verify that you receive a response value of 150 and message of Original transaction found, a <matchCount> value of 2, and the response messages for the transactions submitted in Steps 2 and 4, as a children of the <result_Max10> element.

6. Submit an authorization or sale transaction using a value of error_id for the id attribute.

7. Submit a queryTransaction using error_id for the origId value.
   Verify that you receive a response value of 152 and message of Original transaction found but response not yet available, a <matchCount> value of 1, and the <queryTransactionUnavailableResponse> element as a child of the <result_Max10> element.

2.4.6 Testing Stored Credentials Processing

When you submit the first transaction in a recurring/installment stream, or when storing credentials for future purchases, you must set the <processingType> element to either initialRecurring, initialInstallment, or initialCOF (Card on File), as applicable. With the exception of an American Express transaction, the XML response message includes the <networkTransactionId> element. You must retain the value returned for use in future transactions. When you submit the next and all subsequent transactions in the recurring/installment stream, set the <orderSource> to recurring or installment as appropriate, and include the networkTransactionId value in the <originalNetworkTransactionId> element. For a CoF transaction, set the <orderSource> to ecommerce and the <processingType> element to either merchantInitiatedCOF, or cardholderInitiatedCOF (Card on File), as applicable.

To test recurring/installment and COF transactions, do the following:

1. Submit either Authorization or Sale transactions for orderIds Net_Id1, Net_Id2, and Net_Id3 using the data shown in Table 2-9. Verify that each transactions response indicates approval and that you parse and store the value of the networkTransactionId element for each transaction.

   NOTE: In the Pre-Live environment, the system may return all zeros for the networkTransactionId.

2. Submit transactions Net_Id1a, Net_Id2a, Net_Id3a and Net_Id3b using the supplied data. Verify that each transaction results in an approval.
NOTE: Each of the responses include a networkTransactionId value. You can either store this value for future use in subsequent transactions or continue to use the original value returned in the initial transactions.

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>Net_Id1</td>
<td>Authorization/Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>4999</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300011001</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0521</td>
</tr>
<tr>
<td></td>
<td>&lt;processingType&gt;</td>
<td>initialRecurring</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Net_Id1a</td>
<td>Authorization/Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>4999</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300011001</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0521</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>463</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;originalNetworkTransactionId&gt;</td>
<td>Value from Net_Id1 response</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Net_Id2</td>
<td>Authorization/Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5500</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John &amp; Mary Smith</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>1 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>Burlington</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>MA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>01803-3747</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>44570100000000009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0121</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>349</td>
</tr>
<tr>
<td></td>
<td>&lt;processingType&gt;</td>
<td>initialInstallment</td>
</tr>
<tr>
<td>orderId</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>Net_Id2a</td>
<td>Authorization/Sale: 5500 &lt;type&gt; VI &lt;number&gt; 4457010000000009 &lt;expDate&gt; 0121 &lt;orderSource&gt; installment &lt;originalNetworkTransactionId&gt; Value from Net_Id2 response</td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;authCode&gt; 11111 &lt;avsResult&gt; 00 &lt;cardValidationResult&gt; P</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;networkTransactionId&gt; Dynamically generated value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>Net_Id3</td>
<td>Authorization/Sale: 5500 &lt;name&gt; John Smith &lt;addressLine1&gt; 1 Main St. &lt;city&gt; Waltham &lt;state&gt; MA &lt;zip&gt; 02453 &lt;country&gt; US &lt;type&gt; VI &lt;number&gt; 4457000800000002 &lt;expDate&gt; 0121 &lt;cardValidationNum&gt; 349 &lt;processingType&gt; initialCOF</td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td>Net_Id3a</td>
<td>Authorization/Sale: 2500 &lt;type&gt; VI &lt;number&gt; 4457000800000002 &lt;expDate&gt; 0121 &lt;orderSource&gt; ecommerce &lt;processingType&gt; merchantInitiatedCOF &lt;originalNetworkTransactionId&gt; Value from Net_Id3 response</td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Authorization Response: &lt;response&gt; 000 &lt;message&gt; Approved</td>
</tr>
</tbody>
</table>

**TABLE 2-9** Stored Credentials Test Data (Continued)
2.4.7 Testing Online Duplicate Transaction Processing

When you submit certain Online transactions, the system acts to detect if it is a duplicate by comparing the id attribute and the credit card number against other successful Online transactions of the same type processed within the previous two days. The system performs this checking routine for the following transaction types: Capture, Force Capture, Capture Given Auth, Credit, Sales, eCheck Credit, eCheck Sales, eCheck Void, and Void, as well as Gift Card transactions.

If the system determines a transaction to be a duplicate, the transaction appears in the Declined Transaction report. This report is available in near real-time via Worldpay eComm iQ, and as an Secure Scheduled Report, which is generated daily for the previous days transactions. Please refer to Online Duplicate Checking on page 9 for additional information.

To test your handling of duplicate transactions:

1. Send any of the following Capture transactions more than once within a two day period: Order numbers 1A, 2A, 3A, 4A, or 5A. The second submission will appear in the Declined Transaction report with a response Reason Code of 251 - Duplicate Transaction. Note: You may have to submit the corresponding Authorization transaction prior to submitting the Capture transaction.

2. Send any of the following Credit transactions more than once within a two day period: Order numbers 1B, 2B, 3B, 4B, or 5B. The second submission will appear in the Declined Transaction report with a response Reason Code of 251 - Duplicate Transaction. Note: You may have to submit the corresponding Capture transaction prior to submitting the Credit transaction.

3. Send any of the following Void transactions more than once within a two day period: Order numbers 1C, 2C, 3C, 4C, or 5C. The second submission will appear in the Declined Transaction report with a response Reason Code of 251 - Duplicate Transaction. Note: You may have to submit the corresponding Sale transaction prior to submitting the Void transaction.

NOTE: When you submit the duplicate transaction, make sure that all information, including the id attribute, is identical.

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>Net_Id3b</td>
<td>Authorization/Sale:</td>
<td>4000</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>4457000800000002</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>0121</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>cardholderInitiatedC</td>
</tr>
<tr>
<td></td>
<td>&lt;processingType&gt;</td>
<td>OF</td>
</tr>
<tr>
<td></td>
<td>&lt;originalNetworkTrans</td>
<td>Value from Net_Id3</td>
</tr>
<tr>
<td></td>
<td>actionId&gt;</td>
<td>response</td>
</tr>
</tbody>
</table>

TABLE 2-9  Stored Credentials Test Data (Continued)
4. Send either of the following eCheck Sale transactions more than once within a two day period: Order numbers 42 or 43. The second submission will appear in the Declined Transaction report with a response Reason Code of 251 - Duplicate Transaction.

5. Send any of the following eCheck Credit transactions more than once within a two day period: Order numbers 46, 47, or 48. The second submission will appear in the Declined Transaction report with a response Reason Code of 251 - Duplicate Transaction.

### 2.4.8 Testing Refund Authorization

Beginning in April of 2019, Visa will require the submission of an Authorization for each Credit (Refund) transaction. When you submit a Credit transaction, Worldpay automatically generates the Authorization transaction for you. If the Authorization fails, the response message contains a response code normally used for Authorization declines. The test below illustrates this type of decline.

To test Refund Authorizations, do the following:

1. Submit the Sale transactions for orderId Sale11111 using the data shown in Table 2-10. Verify that each transactions response indicates approval and that you parse and store the value of the networkTransactionId element for each transaction.

**NOTE:** In the Pre-Live environment, the system may return all zeros for the networkTransactionId.

2. Submit a Credit transaction using the cnpTnxId returned in the Sale response. Verify that you handle the declined response correctly.

**TABLE 2-10** Credit Authorization Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>Sale111111</td>
<td>Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>111111</td>
</tr>
<tr>
<td></td>
<td>&lt;&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457002900000009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1223</td>
</tr>
<tr>
<td></td>
<td>Credit:</td>
<td>Use the value</td>
</tr>
<tr>
<td></td>
<td>&lt;cnpTnxId&gt;</td>
<td>returned from the above transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.5 Performing the Optional Tests

This section describes data that you can use to test different response codes, messages, and AVS response codes from the Worldpay system for American Express, Visa, MasterCard, Discover, and Diner’s Club cards. You can perform these tests after completing the certification testing.

This section contains the following topics:

- Testing AVS and Card Validation
- Testing Address Responses
- Testing Advanced AVS Response Codes
- Testing Response Reason Codes and Messages
- Testing 3DS Responses
- Testing the Prepaid Filtering Feature
- Testing the International Card Filter Feature
- Testing Security Code No-Match Filtering
- Testing Advanced Fraud Tools
- Testing Account Updater
- Testing Tax Billing
- Testing Convenience Fees
- Testing the Recycling Engine
- Testing Recurring Engine Transactions
- Testing Gift Card Transactions
- Testing MasterPass Transactions
- Testing Apple Pay Transaction Processing
- Testing Google Pay Transaction Processing
- Testing Amazon Pay
- Testing checkoutId
- Testing Transaction Volume Capacity
2.5.1 Testing AVS and Card Validation

Use the AVS tests to test all of the possible AVS response codes that the system can produce, including those response codes that cannot be produced by varying the address and ZIP code data. For these tests the AVS response codes are independent of any address or ZIP code data that you submit.

To test AVS response codes:

1. Submit transactions using the card data in **Table 2-11**. If you are using Card Validation, include the `cardValidationNum` element. The Card Validation test will return all possible Card Validation response codes. The response codes that are returned are independent of the card validation value that you submit.

2. Verify that the AVS tests return the following response:

   ```xml
   <response>000</response>
   <message>Approved</message>
   <authCode>654321</authCode>
   <avsresult>See Table 2-11</avsresult>
   <cardValidationResult>See Table 2-11</cardValidationResult>
   ```

**NOTE:** For a list of all possible AVS response codes, see **AVS Response Codes** on page 808.
For a list of all possible card validation response codes, see **Card Validation Response Codes** on page 812.

**TABLE 2-11** AVS and Card Validation Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>65</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000300000007</td>
</tr>
<tr>
<td>66</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000100000009</td>
</tr>
<tr>
<td>67</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457003100000003</td>
</tr>
<tr>
<td>68</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000400000006</td>
</tr>
<tr>
<td>69</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000200000008</td>
</tr>
<tr>
<td>70</td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112000100000003</td>
</tr>
</tbody>
</table>
2.5.2 Testing Address Responses

Use the address tests to test different AVS responses by varying the address and ZIP code data. The address tests are intended to return a realistic AVS response code.

To test address responses:

1. Submit Authorization or Sale transactions in Table 2-12. The AVS tests always return an `avsResult` of 00 when submitting 95 Main St. and 95022. If you extend the Zip Code to 9 digits, by appending 1111, the system returns an `avsResult` of 01, as shown in the second test.

2. The AVS response code depends on the Address Line 1 and ZIP Code that are passed in with the transaction. Submit additional transactions using the card data from the table, but varying the address/zip information to receive other `avsResult` codes in the response messages as shown in the tests for orderIds AVS3 and AVS4.

For a detailed list of all possible AVS response codes, see AVS Response Codes on page 808.

### TABLE 2-11 AVS and Card Validation Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>71</td>
<td>&lt;type&gt; MC</td>
<td>5112002100000009</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; M</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>&lt;type&gt; MC</td>
<td>5112002200000008</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; N</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>&lt;type&gt; MC</td>
<td>5112000200000002</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; N</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>&lt;type&gt; MC</td>
<td>5112000300000001</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; P</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>&lt;type&gt; MC</td>
<td>5112000400000000</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; U</td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>&lt;type&gt; MC</td>
<td>5112010400000009</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; S</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>&lt;type&gt; MC</td>
<td>5112000600000008</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationResult&gt; P</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>&lt;type&gt; AX</td>
<td>374313304211118</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt; 352</td>
<td>&lt;message&gt; Decline CVV2/CID Fail</td>
</tr>
</tbody>
</table>

**Note:** American Express CID failures are declined by American Express.
2.5.3 Testing Advanced AVS Response Codes

The Advanced AVS (AAVS) feature is an offering from American Express that allows you to check several parameters not normally covered by a standard AVS check, including name, phone, and email.

To test AAVS Response Codes:
1. Submit an Authorization transaction using the data supplied in Table 2-14.
2. Verify that you handle the response correctly.
3. Enter additional transaction varying the values for name, phone, and/or email to trigger other AAVS results (see AAVS Response Codes on page 809 for other result codes). To obtain a value of 3 in any position, use one of the following values in the appropriate position:
   - <name>Jane Doe</name>
   - <phone>555551234</phone>
   - <email>badtest@test.com</email>

For example, if you submit a second transaction using the name Jane Doe Instead of John Doe, the AAVS result would be 311 indicating No Match for name, but Match for phone and email.
### TABLE 2-13  Advanced AVS Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>81</td>
<td>&lt;amount&gt;</td>
<td>12523</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John Doe</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>95 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>Palo Alto</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>CA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>950221111</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt;</td>
<td><a href="mailto:test@test.com">test@test.com</a></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt;</td>
<td>6178675309</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>341234567890127</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
</tbody>
</table>

### 2.5.4 Testing Response Reason Codes and Messages

Use the data provided in this section to test Response Reason Codes and Messages.

**NOTE:** If you submit account numbers not specified in the tables, you will receive the following response:

```
<response>000</response>
<message>Approved</message>
<authCode>123457</authCode>
<avsResult>00</avsResult>
```

To test Response Codes and Messages:

1. Submit transactions using the data in **Table 2-14**. In each case use the supplied card number with a prefix of **RRC-** as the orderId.
2. Verify that you handle the response correctly.

**NOTE:** The messages listed are samples of messages that the system can return. Since the messages are subject to change at any time, you should use them only for human readability purposes and not for coding purposes. Always code to the response codes, since these do not change.
• For a list of all possible response reason codes, see Payment Transaction Response Codes on page 786.
• For a list of all possible AVS response codes, see AVS Response Codes on page 808.

### TABLE 2-14  Response Code Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRC-card #</td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457000800000002</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457000900000001</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457001000000008</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112000900000005</td>
<td>&lt;message&gt; Approved</td>
</tr>
</tbody>
</table>
### TABLE 2-14 Response Code Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; AX</td>
<td>375000030000001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td>601100040000000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>44570012000000006</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>44570013000000005</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457001400000004</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5112001000000002</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457001900000009</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457002000000006</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457002100000005</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457002200000004</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; AX</td>
<td>375000050000000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457002300000003</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>4457002500000001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5112001600000006</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-14  Response Code Test Data

<table>
<thead>
<tr>
<th>RRC-card #</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>orderId</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;response&gt; 301</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112001700000005</td>
<td>&lt;message&gt; Invalid Account Number</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;response&gt; 321</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112001800000004</td>
<td>&lt;message&gt; Invalid Merchant</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 101</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457002700000009</td>
<td>&lt;message&gt; Issuer Unavailable</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;response&gt; 305</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112001900000003</td>
<td>&lt;message&gt; Expired Card</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 322</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457002800000008</td>
<td>&lt;message&gt; Invalid Transaction</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 350</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457002900000007</td>
<td>&lt;message&gt; Generic Decline</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 101</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457003000000004</td>
<td>&lt;message&gt; Issuer Unavailable</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;response&gt; 101</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5112002000000000</td>
<td>&lt;message&gt; Issuer Unavailable</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt; 301</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457000100000000</td>
<td>&lt;message&gt; Invalid Account Number</td>
</tr>
</tbody>
</table>
2.5.5 Testing 3DS Responses

The cardholder authentication value should only be included by merchants who support 3DS (3 Domain Secure) electronic commerce transactions. Your systems must be in compliance with the Verified by Visa or MasterCard Secure Code implementations of 3DS.

To test 3DS responses:

1. Submit Authorization transactions or Sale transactions using the data in Table 2-15. For all tests except the last three Discover tests, set the orderSource element to either 3dsAuthenticated or 3dsAttempted and the cardholderAuthentication element to the following base64 encoded string:

   BwABBJQ1AgAAAAAgJDUCAAAAAA=

   The response from a 3DS test will be the same as an Authorization or Sale response, except the authenticationResult element will be included in the response as a child of the fraudResult element.

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>3DS1</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300000004</td>
</tr>
<tr>
<td>3DS2</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300000012</td>
</tr>
<tr>
<td>3DS3</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200300000103</td>
</tr>
<tr>
<td>3DS4</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>410020030001002</td>
</tr>
<tr>
<td>3DS5</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000020</td>
</tr>
<tr>
<td>3DS6</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000038</td>
</tr>
<tr>
<td>3DS7</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000046</td>
</tr>
<tr>
<td>3DS8</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000053</td>
</tr>
<tr>
<td>3DS9</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000061</td>
</tr>
<tr>
<td>3DS10</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>41002003000079</td>
</tr>
</tbody>
</table>
### TABLE 2-15 3DS Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>3DS11</td>
<td>&lt;type&gt; VI</td>
<td>4100200300000087</td>
</tr>
<tr>
<td>3DS12</td>
<td>&lt;type&gt; VI</td>
<td>4100200300000095</td>
</tr>
<tr>
<td>3DS13</td>
<td>&lt;type&gt; VI</td>
<td>4100200300000111</td>
</tr>
<tr>
<td>3DS14</td>
<td>&lt;type&gt; VI</td>
<td>4100200300000129</td>
</tr>
<tr>
<td>3DS15</td>
<td>&lt;orderSource&gt; 3dsAttempted</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5112010200000001</td>
</tr>
<tr>
<td>3DS16</td>
<td>&lt;orderSource&gt; 3dsAttempted</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5112010200000001</td>
</tr>
<tr>
<td>3DS17</td>
<td>&lt;orderSource&gt; 3dsAuthenticated</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5407102010000018</td>
</tr>
<tr>
<td>3DS18</td>
<td>&lt;orderSource&gt; 3dsAuthenticated</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>516701020236549</td>
</tr>
<tr>
<td>3DS19</td>
<td>&lt;orderSource&gt; 3dsAttempted</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>5154605300000121</td>
</tr>
<tr>
<td>DI3DS1</td>
<td>&lt;orderSource&gt; 3dsAuthenticated</td>
<td>DI</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td>6011000400001008</td>
</tr>
<tr>
<td></td>
<td>&lt;cardholderAuthentication&gt; BRICAIAISNBERERERERERARERE=</td>
<td></td>
</tr>
<tr>
<td>DI3DS2</td>
<td>&lt;orderSource&gt; 3dsAuthenticated</td>
<td>DI</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td>6011000400000018</td>
</tr>
<tr>
<td></td>
<td>&lt;cardholderAuthentication&gt; FRIBAIASNBERERERERARERE=</td>
<td></td>
</tr>
</tbody>
</table>
2.5.6 Testing the Prepaid Filtering Feature

Complete this test only if you are planning on using the Prepaid Filtering Feature and are using schema version 8.3 or above.

To test the Prepaid Filtering feature:
1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-16.
2. Verify that your response values match those shown in the Key Response Elements section of Table 2-16.
3. After you complete this test, go to the next test.

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>DI3DS3</td>
<td>&lt;orderSource&gt; 3dsAuthenticated</td>
<td>&lt;authenticationResult&gt; 2</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 6011000400000109</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;cardholderAuthentication&gt; JRIBAIASNBERERERERERERERE=</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>table 2-15 3DS Test Data</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterPrepaidMC</td>
<td>&lt;amount&gt; 9100 recurring</td>
<td>&lt;response&gt; 309</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt; recurring</td>
<td>&lt;message&gt; Restricted Card - Prepaid Card Filtering Service</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; John Doe</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt; 10 Main St.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt; San Jose</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt; CA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt; 95032</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt; US</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt; <a href="mailto:jdoe@phoenixProcessing.com">jdoe@phoenixProcessing.com</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt; 7812701111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5500000958501839</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1121</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;prepaid&gt; true</td>
<td></td>
</tr>
</tbody>
</table>
2.5.7 Testing the International Card Filter Feature

Complete this test only if you are planning on using the International Card Filtering Feature and are using schema version 8.3 or above.

To test the International Card Filtering feature:

1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-17.
2. Verify that your response values match those shown in Key Response Elements section of Table 2-17.
3. After you complete this test, go to the next test.

<table>
<thead>
<tr>
<th>orderID</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterPrepaidVI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>element</td>
<td>value</td>
<td>element</td>
</tr>
<tr>
<td>&lt;amount&gt;</td>
<td>9100</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td>&lt;orderSource&gt;</td>
<td>recurring</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td>&lt;name&gt;</td>
<td>John Doe</td>
<td></td>
</tr>
<tr>
<td>&lt;addressLine1&gt;</td>
<td>10 Main St.</td>
<td>&lt;authCode&gt;</td>
</tr>
<tr>
<td>&lt;city&gt;</td>
<td>San Jose</td>
<td>&lt;avsResult&gt;</td>
</tr>
<tr>
<td>&lt;state&gt;</td>
<td>CA</td>
<td></td>
</tr>
<tr>
<td>&lt;zip&gt;</td>
<td>95032</td>
<td></td>
</tr>
<tr>
<td>&lt;country&gt;</td>
<td>US</td>
<td></td>
</tr>
<tr>
<td>&lt;email&gt;</td>
<td><a href="mailto:jdoe@phoenixProcessing.com">jdoe@phoenixProcessing.com</a></td>
<td></td>
</tr>
<tr>
<td>&lt;phone&gt;</td>
<td>7812701111</td>
<td></td>
</tr>
<tr>
<td>&lt;type&gt;</td>
<td>VI</td>
<td></td>
</tr>
<tr>
<td>&lt;number&gt;</td>
<td>46500002010001478</td>
<td></td>
</tr>
<tr>
<td>&lt;expDate&gt;</td>
<td>1121</td>
<td></td>
</tr>
<tr>
<td>&lt;prepaid&gt;</td>
<td>true</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-17  International Filtering Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>filterInternational1</td>
<td>9100</td>
<td>recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John Doe</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>10 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>San Jose</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>CA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>95032</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt;</td>
<td><a href="mailto:jdoe@phoenixProcessing.com">jdoe@phoenixProcessing.com</a></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt;</td>
<td>7812701111</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200309950001</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td>filterInternational2</td>
<td>9100</td>
<td>recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John Doe</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>10 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>San Jose</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>CA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>95032</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt;</td>
<td><a href="mailto:jdoe@phoenixProcessing.com">jdoe@phoenixProcessing.com</a></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt;</td>
<td>7812701111</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200309950001</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;international&gt;</td>
<td>false</td>
</tr>
</tbody>
</table>

#### 2.5.8  Testing Security Code No-Match Filtering

Complete this test only if you are planning on using the Security Code No-Match Filtering Feature.

To test the Security Code No-Match feature:
1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-18.

2. Verify that your response values match those shown in Key Response Elements section of Table 2-18.

After you complete this test, go to the next test.

**TABLE 2-18  Security Code No-Match Filtering Test Data**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>9100</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John Doe</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>10 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>San Jose</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>CA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>95032</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>51120022000000008</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>&lt;international&gt;</td>
<td>false</td>
</tr>
<tr>
<td>securityCodeFilter1</td>
<td>&lt;amount&gt;</td>
<td>9100</td>
</tr>
<tr>
<td></td>
<td>&lt;orderSource&gt;</td>
<td>ecommerce</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Jane Doe</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>10 Main St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>San Jose</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>CA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>95032</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112000200000002</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>&lt;international&gt;</td>
<td>false</td>
</tr>
</tbody>
</table>
2.5.9 Testing Advanced Fraud Tools

Complete this test only if you are planning on using the Advanced Fraud Tools Feature.

To test the Advanced Fraud Tools feature:

1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-19. For each test, replace "Your Prefix-" with the prefix supplied by your Implementation Consultant.
2. Verify that your response values match those shown in Key Response Elements section of Table 2-19.

NOTE: You can submit these tests as sale transactions using the supplied data, or as standalone fraudCheck transactions using just the designated orderId and threatMetrixsessionId. The results for each test type will be as shown in the Key response elements section.

Also, please note that the third test, orderId = tmx_fail_order_id, has two possible results depending upon whether you are configured for Info Only or Auto-Decline.

After you complete this test, go to the next test.

TABLE 2-19 Advanced Fraud Tools Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>tmx_pass_order_id</td>
<td>&lt;amount&gt;</td>
<td>150</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
</tr>
<tr>
<td></td>
<td>&lt;threatMetrixSessionId&gt;</td>
<td>YourPrefix-A980A93LP2 O3-KNP0050</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>tmx_review_order_id</th>
<th>&lt;amount&gt;</th>
<th>150</th>
<th>Result if Info Only:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
<td>&lt;deviceReviewStatus&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;threatMetrixSessionId&gt;</td>
<td>YourPrefix-A0S9D8F7G 6H5J4-KMR-020</td>
<td>&lt;deviceReputationScore&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td>orderId</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
<td>Element</td>
</tr>
<tr>
<td>tmx_fail_order_id</td>
<td>&lt;amount&gt;</td>
<td>150</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
<td>&lt;deviceReviewStatus&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
<td>&lt;deviceReputationScore&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;threatMetrixSessionId&gt;</td>
<td>YourPrefix-Q1W2E3R4T5Y6U7I8-KHF-100</td>
<td>&lt;&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;deviceReviewStatus&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;deviceReputationScore&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;triggeredRule&gt;</td>
</tr>
<tr>
<td>tmx_unavail_order_id</td>
<td>&lt;amount&gt;</td>
<td>150</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
<td>&lt;deviceReviewStatus&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;threatMetrixSessionId&gt;</td>
<td>YourPrefix-Q1W2E3R4T5Y6U7I8-XLP0050</td>
<td></td>
</tr>
</tbody>
</table>
2.5.10 Testing Account Updater

To test Account Updater, you submit a normal Authorization transaction. The certification system returns an Authorization response that includes Account Update information. You should verify that you correctly parse the update information.

**NOTE:** You can also perform the tests in this section using Sale transactions instead of Authorization transactions.

To test the Account Updater service:

1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-20.
2. Verify that your response values match those shown in Key Response Elements section of Table 2-20.
3. If you have coded to receive Extended Response Codes, proceed to the next test.

**TABLE 2-20** Account Updater Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th></th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000300000007</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;newCardInfo&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;number&gt;</td>
<td>5112000100000003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0115</td>
</tr>
<tr>
<td>101</td>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;type&gt;</td>
<td>DI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;number&gt;</td>
<td>6500102087026221</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;newCardInfo&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;type&gt;</td>
<td>DI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;number&gt;</td>
<td>6011102077026225</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0115</td>
</tr>
</tbody>
</table>
2.5.10.1 Testing Account Updater Extended Response Codes

To test the Account Updater Extended Response Codes feature:

1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-21.

2. Verify that the response values match those shown in Key Response Elements section of Table 2-21 and that your systems parse the data correctly. The second test case does not include account repair information only the Extended Response Code.

### TABLE 2-21 Account Updater Extended Response Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>102</td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112000101110009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1199</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457000301100004</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1199</td>
</tr>
</tbody>
</table>

2.5.10.2 Testing Account Updater for Tokenized Merchants

If you are a tokenized merchant using the Account Updater service, you can test this service using the card information provided in Table 2-20. In this case you will receive an original and new token in the <accountUpdater> section of the Authorization response message (see accountUpdater Structure - Credit Cards (tokenized Merchant) on page 320).

2.5.11 Testing Tax Billing

This test applies only to merchants with MCC 9311.

To test Tax Billing and Convenience Fee transactions:
1. Submit authorization transactions using the values provided in Supplied Data Elements of Table 2-22. Note: The second transactions omits the <taxType> element.

2. Verify that the system returns a response code of 000 - Approved for the first transaction and response code of 851 - MCC 9311 requires taxType element for the second.

### TABLE 2-22  Tax Billing Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>MCC9311Test</td>
<td>&lt;amount&gt; 3000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457010200000247</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1121</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;taxType&gt; fee</td>
<td></td>
</tr>
<tr>
<td>MCC9311Test2</td>
<td>&lt;amount&gt; 3000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457010200000247</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1121</td>
<td></td>
</tr>
</tbody>
</table>

### 2.5.12 Testing Convenience Fees

You include Convenience Fees through the use of the secondaryAmount element.

To test the use of Convenience Fees submit the transactions detailed below:

1. Submit authorization or sale transactions for the first five transactions of Table 2-23 using the values provided in Supplied Data Elements columns.

2. Verify that the response values match those shown in Key Response Elements section of Table 2-23 for those transactions and that your systems parse the data correctly.

3. Submit sale transactions for Order Id SaleWOSecondary using the data provided.

4. After receiving an approval for the sale transaction, submit a credit transaction using the litleTnxId from the sale transaction and including the secondaryAmount element with the value provided.

5. Verify that the response values match those shown in Key Response Elements section for the credit transaction using Order Id SaleWOSecondary and that your systems parse the data correctly.
**TABLE 2-23**  Convenience Fee Test Data

<table>
<thead>
<tr>
<th>orderbyd</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>Visa_secondary Amount</td>
<td>&lt;amount&gt;</td>
<td>10500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457010200000247</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;secondaryAmount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td>SecondaryAmt_Higher</td>
<td>&lt;amount&gt;</td>
<td>2500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
</tr>
<tr>
<td></td>
<td>&lt;secondaryAmount&gt;</td>
<td>3000</td>
</tr>
<tr>
<td>MOP_Unsupported</td>
<td>&lt;amount&gt;</td>
<td>6002</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>375001010000003</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>0421</td>
</tr>
<tr>
<td></td>
<td>&lt;secondaryAmount&gt;</td>
<td>1100</td>
</tr>
<tr>
<td>Negative_Secondary</td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457010200000247</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
<tr>
<td></td>
<td>&lt;secondaryAmount&gt;</td>
<td>-500</td>
</tr>
<tr>
<td>Partial_Not_Alated</td>
<td>&lt;amount&gt;</td>
<td>2500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4111111111111111</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1230</td>
</tr>
<tr>
<td></td>
<td>&lt;secondaryAmount&gt;</td>
<td>3000</td>
</tr>
<tr>
<td></td>
<td>&lt;allowPartialAuth&gt;</td>
<td>true</td>
</tr>
<tr>
<td>SaleWOSecondary (Sale TXN)</td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5112010140000004</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1121</td>
</tr>
</tbody>
</table>
Testing Your cnpAPI Transactions

2.5.13 Testing the Recycling Engine

The Certification test cases for the Recycling Engine serve two purposes. First, you use the test transactions to verify your handling of the responses you receive if you submit additional Authorization transactions for a declined auth being handled by the engine. Second, you can verify your process for retrieving and processing recycling completion files via sFTP.

There are three test scenarios you can use to test the Recycling Engine and your ability to parse the response messages and/or result Batches. The particular data sets and scenarios you use depends upon the version of cnpAPI you use, as well as your plans for retrieving the response messages. Use the following to determine which tests you should run:

- If you plan to retrieve recycling results via the results Batches posted daily to the FTP site, perform the tests in Scenario 1.
- If you are using cnpAPI schema version V8.6 or above, perform the tests in Scenario 2.

Scenario 1

To test your handling of the Recycling Results Session file:

1. Submit authorization (or sale) transactions using the values provided in the Supplied Data Elements column of Table 2-24. Please use the same value for the orderId and if applicable, the recycleId elements.

   NOTE: If your configuration is set for Worldpay to recycle by default, you can omit the <recycleBy> element.

2. Wait a minimum of 2 hours after submitting the last transaction. After 2 hours, retrieve the Results Session file from the FTP site.

TABLE 2-23  Convenience Fee Test Data

<table>
<thead>
<tr>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>&lt;litleTxnId&gt;</td>
<td>Value from previous transaction</td>
</tr>
<tr>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td>&lt;secondaryAmount&gt;</td>
<td>400</td>
</tr>
</tbody>
</table>

SaleWOSSecond ary (Credit Txn)
### TABLE 2-24  Recycling Engine Test Data - Results Session File Pick-up

<table>
<thead>
<tr>
<th>orderID or recycleID (replace XXX with your merchantID)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>XXXCase1Order1</td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4457012400000027</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt;</td>
<td>Litle</td>
</tr>
<tr>
<td>XXXCase1Order2</td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5160124000000029</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt;</td>
<td>Litle</td>
</tr>
<tr>
<td>XXXCase1Order3</td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4100200700000059</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt;</td>
<td>Litle</td>
</tr>
</tbody>
</table>
### TABLE 2-24  Recycling Engine Test Data - Results Session File Pick-up

<table>
<thead>
<tr>
<th>orderId or recycleId (replace XXX with your merchantId)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase1Order4</td>
<td>10000</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>MC</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>55000100000000052</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>1220</td>
<td>&lt;recyclingEngineActive&gt;</td>
</tr>
<tr>
<td></td>
<td>Little</td>
<td>Final Response (in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true</td>
</tr>
<tr>
<td>XXXCase1Order5</td>
<td>10000</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>4457032800000047</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>1220</td>
<td>&lt;recyclingEngineActive&gt;</td>
</tr>
<tr>
<td></td>
<td>Little</td>
<td>Final Response (in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true</td>
</tr>
<tr>
<td>XXXCase1Order6</td>
<td>10000</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>MC</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>5160328000000042</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>1220</td>
<td>&lt;recyclingEngineActive&gt;</td>
</tr>
<tr>
<td></td>
<td>Little</td>
<td>Final Response (in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true</td>
</tr>
</tbody>
</table>

| Cardholder requests that recurring or installment payment be stopped | 120 | Call Issuer |
Testing Your cnpAPI Transactions

Scenario 2

to test your handling of the Intercept Response Codes/Messages, as well as the Recycling Results Session file and Normal Batch/Online responses:

1. Submit authorization (or sale) transactions using the values provided in the Supplied Data Elements column of Table 2-25. Please use the same value for the orderId and if applicable, the recycleId elements.

2. If you are using schema version V8.6 or above, resubmit any of the first four transactions within 36 hours to receive a response message containing the intercept Response Reason Code 372 - Soft Decline - Auto Recycling In Progress. If you are using schema version 8.5 or below, you will receive a response message with the same Response Reason Code as in the initial response message.

3. Wait a minimum of 36 hours after submitting the last of the initial transactions. After 36 hours, you can retrieve the Results Session file from the FTP site and/or resubmit the transaction. The responses will contain the data shown for the Final Response.

**NOTE:** If your configuration is set for Worldpay to recycle by default, you can omit the <recycleBy> element.

---

**TABLE 2-24** Recycling Engine Test Data - Results Session File Pick-up

<table>
<thead>
<tr>
<th>orderId or recycleId (replace XXX with your merchantId)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase1Order7</td>
<td>&lt;amount&gt; 10000</td>
<td>Initial Response: 302 Account Number Does Not Match Payment Type</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;response&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5160328000000042</td>
<td>&lt;message&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td>&lt;recyclingEngineActive&gt; false</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Litle</td>
<td>Final Response (in FTP Session File): None - This type of decline is not recycled.</td>
</tr>
<tr>
<td>orderId or recycleId (replace XXX with your merchantId)</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>XXXCase3Order1</td>
<td>&lt;amount&gt; 20000</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td>&lt;response&gt; 350</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 6223012400000025</td>
<td>&lt;message&gt; Generic Decline</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td>Intermediate Attempts (V8.6):</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Litle</td>
<td>&lt;response&gt; 372</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Soft decline - Recycling In Progress</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;recyclingEngineActive&gt; true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Final Response (Online/Normal Batch, or in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Approved</td>
</tr>
</tbody>
</table>
## TABLE 2-25  Recycling Engine Test Data - Intercept and Online or Results Session File Pick-up

<table>
<thead>
<tr>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>&lt;amount&gt;</td>
<td>20000</td>
</tr>
<tr>
<td>&lt;type&gt;</td>
<td>AX</td>
</tr>
<tr>
<td>&lt;number&gt;</td>
<td>377201240000025</td>
</tr>
<tr>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td>&lt;recycleBy&gt;</td>
<td>Little</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(orderId or recycleId (replace XXX with your merchantId))
TABLE 2-25  Recycling Engine Test Data - Intercept and Online or Results Session File Pick-up

<table>
<thead>
<tr>
<th>orderId or recycleId (replace XXX with your merchantId)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order3</td>
<td>&lt;amount&gt; 20000</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
<td>&lt;response&gt; 350</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 6223012400000033</td>
<td>&lt;message&gt; Generic Decline</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td>&lt;recyclingEngineActive&gt; true</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Little</td>
<td>Intermediate Attempts (V8.6):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 372</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Soft decline - Recycling In Progress true</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;recyclingEngineActive&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Final Response (Online/Normal Batch, or in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 373</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Hard Decline - Auto Recycling Complete</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order3</td>
<td>&lt;amount&gt; 20000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; DI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 6223012400000033</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Little</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order3</td>
<td>Initial Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt; 350</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt; Generic Decline</td>
</tr>
<tr>
<td></td>
<td>&lt;recyclingEngineActive&gt; true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order3</td>
<td>Intermediate Attempts (V8.6):</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt; 372</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt; Soft decline - Recycling In Progress true</td>
</tr>
<tr>
<td></td>
<td>&lt;recyclingEngineActive&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order3</td>
<td>Final Response (Online/Normal Batch, or in FTP Session File):</td>
</tr>
<tr>
<td></td>
<td>&lt;response&gt; 373</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt; Hard Decline - Auto Recycling Complete</td>
</tr>
</tbody>
</table>
### TABLE 2-25  Recycling Engine Test Data - Intercept and Online or Results Session File Pick-up

<table>
<thead>
<tr>
<th>orderId or recycleId (replace XXX with your merchantId)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXCase3Order4  &lt;amount&gt; 20000  &lt;type&gt; DI  &lt;number&gt; 6011002078551608  &lt;expDate&gt; 1220  &lt;recycleBy&gt; Litle</td>
<td>Initial Response: &lt;response&gt; &lt;message&gt; Issuer Unavailable true  &lt;recyclingEngineActive&gt; Intermediate Attempts (V8.6): &lt;response&gt; &lt;message&gt; 372 Soft decline - Recycling In Progress true &lt;recyclingEngineActive&gt; Final Response (Online/Normal Batch, or in FTP Session File): &lt;response&gt; &lt;message&gt; 373 Hard Decline - Auto Recycling Complete</td>
<td></td>
</tr>
<tr>
<td>XXXCase3Order5  &lt;amount&gt; 20000  &lt;type&gt; VI  &lt;number&gt; 377203280000048  &lt;expDate&gt; 1220  &lt;recycleBy&gt; Litle</td>
<td>Initial Response: &lt;response&gt; &lt;message&gt; Account Number Does Not Match Payment Type false &lt;recyclingEngineActive&gt; All Responses: None - This type of decline is not recycled.</td>
<td></td>
</tr>
</tbody>
</table>
2.5.13.1 Testing Recycling Engine Cancellation

You use an authReversal transaction to halt the automatic recycling of an authorization. For a sale transaction, use a void transaction to halt recycling.

NOTE: You can perform this test either after completing the Recycling Engine test or prior to starting that test.

To test recycling cancellation:

1. Submit a sale transaction using the values provided for Case1Order1a and an authorization transaction using the values provided for Case1Order2a in the Supplied Data Elements of Table 2-26.

2. Two (2) hours after receiving the decline message, submit a void transaction using the littleTxnId returned in the response message for Case1Order1a. If you are not enabled for Auto-refunding an approved Sales on Void, the Declined transaction report will contain a response code of 362 - Transaction not Voided - Already Settled.

NOTE: If you submit the Void transaction (Step 2) within 2 hours of the initial transaction submission, you will receive a voidResponse with a response code of 000 - Approved.

3. After receiving the decline messages, submit an authReversal transaction using the littleTxnId returned in the response message for Case1Order2a. This will halt the recycling of the order.
### TABLE 2-26  Recycling Engine Cancellation Test Data

<table>
<thead>
<tr>
<th>orderId or recycleId (replace XXX with your merchantId)</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>XXXCase1Order1a</strong></td>
<td></td>
<td><strong>Order 1</strong> (INSUFFICIENT FUNDS)</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 11000</td>
<td>&lt;response&gt; 110</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; VI</td>
<td>&lt;message&gt; Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 4457012400000027</td>
<td>&lt;recyclingEngineActive&gt; true</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Little</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>voidResponse</strong> (if enabled for Auto-Refund):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;creditLitleTxnId&gt; (Random Value)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>voidResponse</strong> (if not enabled for Auto-Refund):</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Declined Transaction report result = 362 - Transaction Not Voided - Already Settled</td>
</tr>
<tr>
<td><strong>XXXCase1Order2a</strong></td>
<td>&lt;amount&gt; 11000</td>
<td>&lt;response&gt; 110</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; MC</td>
<td>&lt;message&gt; Insufficient Funds</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; 5160124000000029</td>
<td>&lt;recyclingEngineActive&gt; true</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;recycleBy&gt; Little</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>authReversalResponse</strong>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Approved</td>
</tr>
</tbody>
</table>

Submit Void using litleTxnId from Initial Response

Submit authReversal using litleTxnId from Initial Response
2.5.14 Testing Recurring Engine Transactions

Use the following Certification tests to verify transactions associated with the Recurring Engine functionality. For testing purposes, the Certification environment will process the first recurring transaction within 2 hours. The system cancels subsequent recurring transactions in the payment stream. For example, if you submitted an Authorization at 9:00 AM that set-up a subscription for twelve monthly payments, the Cert Recurring Engine would process the first payment by 11:00 AM and cancel the remaining eleven payments.

**IMPORTANT:** The test data supplied does not necessarily account for all data fields/xml elements in a particular request. Where test data is not supplied, you should provide appropriate information. You should never override your own system to enter supplied data. If you are unable to enter the supplied data without overriding your system, please consult your Implementation Consultant concerning the test and how to proceed.

To test the Recurring Engine functionality using the data supplied in Table 2-27:

1. Submit `createPlan` transactions for Order Ids R1.1, R1.2, R1.3, R1.4, R1.5, and R1.6. These transactions establish Plans used in the remaining tests, as well as verify your cnpAPI messages used to create Plans. You can use whatever values you wish for the `<planCode>` and `<name>` elements. Verify that each transaction receives an approval code in the response message.

2. Submit `createPlan` transactions for Order Id R1.7 using the same `<planCode>` value you used in R1.3. This transaction is declined, because the Plan Code already exists. Verify that the transaction receives a response code of 487 - Plan code already exists.

3. Submit a `sale` transaction for Order Id R2.1. This transaction creates a subscription using the plan established in Order R1.1. The amount in the sale transaction represents a set-up fee and not the initial payment. Verify that the transaction receives an approval code in the response message.

4. Submit an `authorization` transaction for Order Id R2.2. This transaction creates a subscription using the plan established in Order R1.1. Because the Authorization did not specify a start date, the Recurring Engine schedules the first payment for the current date. Verify that the transaction receives an approval code in the response message.

**NOTE:** The transaction specified in Order 2.2 is a $0 Auth. If you include an amount when using an Auth to establish a subscription, you should plan on reversing the Auth to avoid a Misuse of Auth fee. The Recurring Engine obtains its own Auth for the first payment.

5. Submit an `updateSubscription` transaction for Order Id R2.3. This transaction updates the subscription initiated with Order Id 2.1, changing the Plan to the plan you created in R1.2.

6. Submit a `sale` transaction for Order Id R3.1. This transaction utilizes a Sale transaction to initialize a subscription based upon the plan created in R1.3, but overrides both the number of payments and the amount specified in the Plan. Verify that the transaction receives an approval code in the response message.

7. Submit an `authorization` transaction for Order Id R3.2. This transaction utilizes an Authorization transaction to initialize a subscription based upon the plan created in R1.3, but overrides both the number of payments and the amount specified in the Plan. Verify that the transaction receives an approval code in the response message.
8. Submit an `updateSubscription` transaction for Order Id R3.3. This transaction updates an existing subscription, changing the billing date.

9. Submit an `authorization` transaction for Order Id R4.1. This transaction utilizes an Authorization transaction to initialize a subscription that overrides the default amount in the Plan and has a trial period. Verify that the transaction receives an approval code in the response message.

10. Submit an `updateSubscription` transaction for Order Id R4.2. This transaction updates an existing subscription with a new credit card number.

11. Submit an `authorization` transaction for Order Id R5.1. This transaction utilizes an Authorization transaction to initialize a subscription based upon the SEMIANNUAL_PLAN, but overrides the number of payments. Verify that the transaction receives an approval code in the response message.

12. Submit an `updateSubscription` transaction for Order Id R5.2. This transaction updates an existing subscription with new billing information.

13. Submit an `updateSubscription` transaction for Order Id R5.3. Since the transaction does not include any updated information, the Declined Transaction report will contain a response code of 484 - Insufficient data to update subscription. Verify the response code by accessing the Declined Transaction report in Worldpay eComm iQ.

14. Submit an `updateSubscription` transaction for Order Id R5.4. Since the transaction specifies a new billing date in the past, the Declined Transaction report will contain a response code of 485 - Invalid billing date. Verify the response code by accessing the Declined Transaction report in Worldpay eComm iQ.

15. Submit an `authorization` transaction for Order Id R6.1. The system declines this transaction, since it uses an invalid Plan Code. Verify that the transaction receives a response code of 472 - Invalid plan code.

16. Submit an `updateSubscription` transaction for Order Id R6.2. Since this order uses an invalid subscription Id, The system declines this transaction, the Declined Transaction report will contain a response code of 482 - Invalid start date. Verify the response code by accessing the Declined Transaction report in Worldpay eComm iQ.

17. Submit an `authorization` transaction for Order Id R6.3. The system declines this transaction, since it uses a start date in the past. Verify that the transaction receives a response code of 475 - Invalid Subscription Id.

18. Submit an `authorization` transaction for Order Id R7.1. This transaction utilizes an Authorization transaction to initialize a subscription based upon the PLAN_WITH_PROMOTIONS Plan with an add On and a Discount applied. Verify that the transaction receives an approval code in the response message.

19. Submit an `updateSubscription` transaction for Order Id R7.2. This transaction attempts to update an existing subscription with a new Add On, but is declined because the Add On already exists. Verify that the Declined Transaction report contains a response code of 476 - Add On already exists.

20. Submit an `updateSubscription` transaction for Order Id R7.3. This transaction attempts to update an Add On for an existing subscription, but is declined because the specified Add On is not associated with the Subscription. Verify that the Declined Transaction report contains a response code of 478 - No matching Add On code for the subscription.

21. Submit an `updateSubscription` transaction for Order Id R7.4. This transaction attempts to update an Add On for an existing subscription, but is declined because the specified Add On is not associated with the Subscription. Verify that the Declined Transaction report contains a response code of 478 - No matching Add On code for the subscription.

22. Submit an `updateSubscription` transaction for Order Id R7.5. This transaction attempts to update a Discount for an existing subscription, but is declined because the specified Discount is not
associated with the Subscription. Verify that the Declined Transaction report contains a response code of 480 - No matching Discount code for the subscription.

23. Submit an authorization transaction for Order Id R8.1. This transaction utilizes an Authorization transaction to initialize a subscription, but includes duplicate Add Ons. Verify that the transaction receives a response code of 477 - Duplicate add-on codes in request.

24. Submit an authorization transaction for Order Id R8.2. This transaction utilizes an Authorization transaction to initialize a subscription, but includes duplicate Discounts. Verify that the transaction receives a response code of 481 - Duplicate discount codes in request.

25. Submit an authorization transaction for Order Id R9.1. In this case the parent Authorization transaction is declined with a response code of 301 - Invalid Account Number. The response code associated with the recurring request is 471 - Parent transaction declined - Recurring subscription not created.PREMIUM_MONTHLY

26. Submit an cancelSubscription transaction for Order Id R10.1. Verify that the transaction receives an approval code in the response message.

27. Submit an updatePlan transaction for Order Id R11.1. This transaction changes the Plan to an inactive state. Verify that the transaction receives an approval code in the response message.

**TABLE 2-27** Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createPlan&gt; (parent element)</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Your Value</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Your Value</td>
<td>&lt;planCode&gt; Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt; Basic monthly plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt; MONTHLY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 5000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;numberOfPayments&gt; 12</td>
<td></td>
</tr>
<tr>
<td>R1.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createPlan&gt; (parent element)</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Your Value</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt; Your Value</td>
<td>&lt;planCode&gt; Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt; Premium monthly plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt; MONTHLY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 7000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;numberOfPayments&gt; 12</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-27 Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderl</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>R1.3</td>
<td>&lt;createPlan&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt;</td>
<td>An annual plan</td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt;</td>
<td>ANNUAL</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>14000</td>
</tr>
<tr>
<td>R1.4</td>
<td>&lt;createPlan&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt;</td>
<td>A monthly plan with trial period</td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt;</td>
<td>MONTHLY</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>&lt;trialIntervalType&gt;</td>
<td>MONTH</td>
</tr>
<tr>
<td></td>
<td>&lt;trialNumberOfIntervals&gt;</td>
<td>2</td>
</tr>
<tr>
<td>R1.5</td>
<td>&lt;createPlan&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt;</td>
<td>A semi-annual plan</td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt;</td>
<td>SEMIANNUAL</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>7000</td>
</tr>
<tr>
<td>R1.6</td>
<td>&lt;createPlan&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Your Value</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt;</td>
<td>A plan with Add Ons and Discounts</td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt;</td>
<td>MONTHLY</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5000</td>
</tr>
<tr>
<td>R1.7</td>
<td>&lt;createPlan&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Value from R1.3</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Value from R1.3</td>
</tr>
<tr>
<td></td>
<td>&lt;description&gt;</td>
<td>An annual plan</td>
</tr>
<tr>
<td></td>
<td>&lt;intervalType&gt;</td>
<td>ANNUAL</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>5000</td>
</tr>
</tbody>
</table>
**TABLE 2-27**  Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>R2.1</strong></td>
<td>&lt;sale&gt; (parent element)</td>
<td>&lt;recurringResponse&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 1000 (parent element)</td>
<td>&lt;responseCode&gt; 470</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt; (parent element)</td>
<td>&lt;responseMessage&gt; Approved - Recurring subscription created</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt; (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Value from R1.1 (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt; Use any date (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;recurringResponse&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;responseCode&gt; 470</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;responseMessage&gt; Approved - Recurring subscription created</td>
</tr>
<tr>
<td><strong>R2.2</strong></td>
<td>&lt;authorization&gt; (parent element)</td>
<td>&lt;recurringResponse&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 000 (parent element)</td>
<td>&lt;responseCode&gt; 470</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt; (parent element)</td>
<td>&lt;responseMessage&gt; Approved - Recurring subscription created</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt; (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Value from R1.1 (parent element)</td>
<td></td>
</tr>
<tr>
<td><strong>R2.3</strong></td>
<td>&lt;updateSubscription&gt; (parent element)</td>
<td>&lt;subscriptionId&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt; Value returned in R2.1 response (parent element)</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Value from R1.2 (parent element)</td>
<td></td>
</tr>
<tr>
<td><strong>R3.1</strong></td>
<td>&lt;sale&gt; (parent element)</td>
<td>&lt;recurringResponse&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt; (parent element)</td>
<td>&lt;responseCode&gt; 470</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt; (parent element)</td>
<td>&lt;responseMessage&gt; Approved - Recurring subscription created</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Value from R1.3 (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt; Use any valid date in the future. (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;numberOfPayments&gt; 6 (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 4000 (parent element)</td>
<td></td>
</tr>
<tr>
<td><strong>R3.2</strong></td>
<td>&lt;authorization&gt; (parent element)</td>
<td>&lt;recurringResponse&gt; (parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt; (parent element)</td>
<td>&lt;responseCode&gt; 470</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt; (parent element)</td>
<td>&lt;responseMessage&gt; Approved - Recurring subscription created</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt; Value from R1.3 (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt; Use any valid date in the future. (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;numberOfPayments&gt; 6 (parent element)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 4000 (parent element)</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-27 Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>R3.3</td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td>Value returned in</td>
</tr>
<tr>
<td></td>
<td>&lt;billingDate&gt;</td>
<td>R3.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;billingDate&gt;</td>
<td>Use any valid date in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the future and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>different from date</td>
</tr>
<tr>
<td></td>
<td></td>
<td>used in R3.1.</td>
</tr>
<tr>
<td>R4.1</td>
<td>&lt;authorization&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Value from R1.4</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>Use any valid date in</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>the future.</td>
</tr>
<tr>
<td></td>
<td>4000</td>
<td></td>
</tr>
<tr>
<td>R4.2</td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td>Value returned in</td>
</tr>
<tr>
<td></td>
<td>&lt;card&gt;</td>
<td>R4.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>VI</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>44570100000000009</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1221</td>
</tr>
<tr>
<td>R5.1</td>
<td>&lt;authorization&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Value from R1.5</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>Use any valid date in</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>the future.</td>
</tr>
<tr>
<td></td>
<td>&lt;numberOfPayments&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-27 Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Element</strong></td>
<td><strong>Value</strong></td>
</tr>
<tr>
<td>R5.2</td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;billToAddress&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>John</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td>900 Chelmsford St.</td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td>Lowell</td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td>MA</td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td>01781</td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td>US</td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt;</td>
<td><a href="mailto:john@gmail.com">john@gmail.com</a></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt;</td>
<td>8559658965</td>
</tr>
<tr>
<td>R5.3</td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td></td>
</tr>
<tr>
<td>R5.4</td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;billingDate&gt;</td>
<td>2000-10-01</td>
</tr>
<tr>
<td>R6.1</td>
<td>&lt;authorization&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>INVALID_PLAN</td>
</tr>
</tbody>
</table>
### TABLE 2-27  Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>R6.2</td>
<td>&lt;updateSubscription&gt;</td>
<td>Declined Transaction</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td>report result = 475 -</td>
</tr>
<tr>
<td></td>
<td>&lt;billToAddress&gt;</td>
<td>Invalid Subscription</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Id</td>
</tr>
<tr>
<td></td>
<td>&lt;addressLine1&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;city&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;state&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;zip&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;country&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;email&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;phone&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1111111111</td>
<td></td>
</tr>
<tr>
<td></td>
<td>John</td>
<td></td>
</tr>
<tr>
<td></td>
<td>900 Chelmsford St.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lowell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>01781</td>
<td></td>
</tr>
<tr>
<td></td>
<td>US</td>
<td></td>
</tr>
<tr>
<td></td>
<td><a href="mailto:john@gmail.com">john@gmail.com</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8559658965</td>
<td></td>
</tr>
<tr>
<td>R6.3</td>
<td>&lt;authorization&gt;</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value from R1.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2000-04-04</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;recurringResponse&gt;</td>
<td>&lt;responseCode&gt; 482</td>
</tr>
<tr>
<td></td>
<td>&lt;responseMessage&gt;</td>
<td>Invalid start date</td>
</tr>
<tr>
<td>R7.1</td>
<td>&lt;authorization&gt;</td>
<td>&lt;response&gt; 470</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>Approved - Recurring</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>Subscription Created</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value from R1.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createAddOn&gt;</td>
<td>&lt;createDiscount&gt;</td>
</tr>
<tr>
<td></td>
<td>SSL_ADDON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Additional Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2013-08-30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2050-08-30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;addOnCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SSL_ADDON</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2013-08-30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2050-08-30</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-27  Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>R7.2</td>
<td>&lt;updateSubscription&gt; (parent element)</td>
<td>Value returned in R7.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createAddOn&gt; (parent element)</td>
<td>SSL_ADDON</td>
</tr>
<tr>
<td></td>
<td>&lt;addOnCode&gt;</td>
<td>Additional Service</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td>R7.3</td>
<td>&lt;updateSubscription&gt; (parent element)</td>
<td>Value returned in R7.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createDiscount&gt; (parent element)</td>
<td>PROMO_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>&lt;discountCode&gt;</td>
<td>Special Offer</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td>R7.4</td>
<td>&lt;updateSubscription&gt; (parent element)</td>
<td>Value returned in R7.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;updateAddOn&gt; (parent element)</td>
<td>INVALID_ADDON_CODE</td>
</tr>
<tr>
<td></td>
<td>&lt;addOnCode&gt;</td>
<td>Extra Features</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
</tbody>
</table>
**Testing Your cnpAPI Transactions**

**TABLE 2-27  Recurring Engine Test Data**

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td></td>
<td>&lt;updateSubscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td>R7.5</td>
<td>&lt;subscriptionId&gt;</td>
<td>Value returned in R7.1 response</td>
</tr>
<tr>
<td></td>
<td>&lt;updateDiscount&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;discountCode&gt;</td>
<td>INVALID_DISCOUNT_CODE</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>Special Offer</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td>R8.1</td>
<td>&lt;authorization&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>Value from R1.6</td>
</tr>
<tr>
<td></td>
<td>&lt;createAddOn&gt;</td>
<td>SSL_ADDON</td>
</tr>
<tr>
<td></td>
<td>&lt;addOnCode&gt;</td>
<td>Additional Service</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>(parent element)</td>
</tr>
<tr>
<td></td>
<td>&lt;createAddOn&gt;</td>
<td>SSL_ADDON</td>
</tr>
<tr>
<td></td>
<td>&lt;addOnCode&gt;</td>
<td>Additional Service</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>(parent element)</td>
</tr>
</tbody>
</table>
### TABLE 2-27  Recurring Engine Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>R8.2</td>
<td>&lt;authorization&gt;</td>
<td>&lt;recurringResponse&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>&lt;responseCode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>&lt;responseMessage&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;createDiscount&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;discountCode&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>PROMO_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>Discount</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;createDiscount&gt;</td>
<td>PROMO_DISCOUNT</td>
</tr>
<tr>
<td></td>
<td>&lt;discountCode&gt;</td>
<td>Discount</td>
</tr>
<tr>
<td></td>
<td>&lt;name&gt;</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;startDate&gt;</td>
<td>2013-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;endDate&gt;</td>
<td>2050-08-30</td>
</tr>
<tr>
<td></td>
<td>&lt;responseCode&gt;</td>
<td>481</td>
</tr>
<tr>
<td></td>
<td>&lt;responseMessage&gt;</td>
<td>Duplicate discount</td>
</tr>
<tr>
<td></td>
<td></td>
<td>codes in request</td>
</tr>
<tr>
<td>R9.1</td>
<td>&lt;authorization&gt;</td>
<td>&lt;recurringResponse&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;card&gt;</td>
<td>&lt;responseCode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>&lt;responseMessage&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>511201010000000002</td>
</tr>
<tr>
<td></td>
<td>&lt;recurringRequest&gt;</td>
<td>0721</td>
</tr>
<tr>
<td></td>
<td>&lt;subscription&gt;</td>
<td>&lt;responseCode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>&lt;responseMessage&gt;</td>
</tr>
<tr>
<td></td>
<td>Value from R1.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;responseCode&gt;</td>
<td>471</td>
</tr>
<tr>
<td></td>
<td>&lt;responseMessage&gt;</td>
<td>Parent transaction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>declined - Recurring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>subscription not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>created</td>
</tr>
<tr>
<td>R10.1</td>
<td>&lt;cancelSubscription&gt;</td>
<td>&lt;recurringResponse&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;subscriptionId&gt;</td>
<td>&lt;responseCode&gt;</td>
</tr>
<tr>
<td></td>
<td>Value returned in</td>
<td>&lt;responseMessage&gt;</td>
</tr>
<tr>
<td></td>
<td>R7.1 response</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;responseCode&gt;</td>
<td>000</td>
</tr>
<tr>
<td></td>
<td>&lt;message&gt;</td>
<td>Approved</td>
</tr>
<tr>
<td>R11.1</td>
<td>&lt;updatePlan&gt;</td>
<td>&lt;responseCode&gt;</td>
</tr>
<tr>
<td></td>
<td>&lt;planCode&gt;</td>
<td>&lt;responseMessage&gt;</td>
</tr>
<tr>
<td></td>
<td>Value from R1.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;active&gt;</td>
<td>000</td>
</tr>
<tr>
<td></td>
<td>false</td>
<td>Approved</td>
</tr>
</tbody>
</table>
### 2.5.15 Testing Gift Card Transactions

Use the following Certification tests to verify transactions associated with the Private Label Gift Card functionality. The tests are designed so that you can verify your XML structure for each transaction type, as well as your ability to parse the response data.

To test the Gift Card functionality using the data supplied in Table 2-28:

1. Submit an activate transaction for Order Id GC1. Verify that the data in the response message matches the Key Response Elements specified in the table.
2. Submit an activate transaction for Order Id GC1A. This transaction activates a Virtual Gift Card. Verify that the data in the response message matches the Key Response Elements specified in the table.
3. Submit an authorization transaction for Order Id GC2. Verify that the data in the response message matches the Key Response Elements specified in the table.
4. Submit a giftCardCapture transaction for Order Id GC2A. Verify that the data in the response message matches the Key Response Elements specified in the table.
5. Submit a giftCardCredit transaction for Order Id GC2B. Verify that the data in the response message matches the Key Response Elements specified in the table.
6. Submit a deactivate transaction for Order Id GC3. Verify that the data in the response message matches the Key Response Elements specified in the table.
7. Submit a load transaction for Order Id GC4. Verify that the data in the response message matches the Key Response Elements specified in the table.
8. Submit a unload transaction for Order Id GC5. Verify that the data in the response message matches the Key Response Elements specified in the table.
9. Submit a balanceInquiry transaction for Order Id GC6. Verify that the data in the response message matches the Key Response Elements specified in the table.
10. Submit an activate transaction for Order Id GC7. Verify that the data in the response message matches the Key Response Elements specified in the table.
11. Submit an activateReversal transaction for Order Id GC7A. Verify that the data in the response message matches the Key Response Elements specified in the table.
12. Submit an activateReversal transaction for Order Id GC1A. Verify that the data in the response message matches the Key Response Elements specified in the table.
13. Submit an authorization transaction for Order Id GC8. Verify that the data in the response message matches the Key Response Elements specified in the table.
14. Submit an authorizationReversal transaction for Order Id GC8A. Verify that the data in the response message matches the Key Response Elements specified in the table.
15. Submit a sale transaction for Order Id GC9. Verify that the data in the response message matches the Key Response Elements specified in the table.

**NOTE:** When processing gift cards, many transactions response messages include the `refCode`, `sequenceNumber`, `systemTraceId`, and `txnTime` elements. You should always store these values for possible use in subsequent transactions. For example, to perform the `giftCardCapture` in test GC2A, you must include the values returned for these elements in the associated `authorization` transaction, GC2.
16. Submit a depositReversal transaction for Order Id GC9A. Verify that the data in the response message matches the Key Response Elements specified in the table.

17. Submit a sale transaction for Order Id GC10. Verify that the data in the response message matches the Key Response Elements specified in the table.

18. Submit a giftCardCredit transaction for Order Id GC10A. Verify that the data in the response message matches the Key Response Elements specified in the table.

19. Submit a refundReversal transaction for Order Id GC10B. Verify that the data in the response message matches the Key Response Elements specified in the table.

20. Submit a deactivate transaction for Order Id GC11. Verify that the data in the response message matches the Key Response Elements specified in the table.

21. Submit a loadReversal transaction for Order Id GC12. Verify that the data in the response message matches the Key Response Elements specified in the table.

22. Submit an unload transaction for Order Id GC13. Verify that the data in the response message matches the Key Response Elements specified in the table.

23. Submit an unloadReversal transaction for Order Id GC13A. Verify that the data in the response message matches the Key Response Elements specified in the table.

24. Submit an activate transaction for Order Id GC14. This transaction is declined with a response code of 301 - Invalid Account Number. Verify that the data in the response message matches the Key Response Elements specified in the table.

25. Submit an activate transaction for Order Id GC15. This transaction is declined with a response code of 217 - Already active, because it attempts to activate the same card already activated in Order Id GC1. Verify that the data in the response message matches the Key Response Elements specified in the table.

26. Submit an authorization transaction for Order Id GC16. This transaction is declined with a response code of 110 - Insufficient Funds. Verify that the data in the response message matches the Key Response Elements specified in the table.

27. Submit an authorization transaction for Order Id GC17. This transaction is declined with a response code of 281 - Card not active. Verify that the data in the response message matches the Key Response Elements specified in the table.

28. Submit a sale transaction for Order Id GC19. Verify that the data in the response message matches the Key Response Elements specified in the table.

29. Submit a giftCardCredit transaction for Order Id GC19A. For this transaction the Declined Transaction report will contain response code of 304 - Lost/Stolen Card.

30. Submit a balanceInquiry transaction for Order Id GC20. This transaction is declined with a response code of 352 - Invalid CVV2. Verify that the data in the response message matches the Key Response Elements specified in the table.
<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td><strong>GC1</strong></td>
<td>Activate:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>15000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Supplied by Implementation Consultant</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td><strong>GC1A</strong></td>
<td>Virtual Activate:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>8000</td>
</tr>
<tr>
<td></td>
<td>&lt;accountNumberLength&gt;</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>&lt;giftCardBin&gt;</td>
<td>Supplied by Implementation Consultant</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>GC2</strong></td>
<td>Authorization:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>1500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Number from GC1</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td><strong>GC2A</strong></td>
<td>Gift Card Capture:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;littleTxnId&gt;</td>
<td>Value from GC2 resp.</td>
</tr>
<tr>
<td></td>
<td>&lt;captureAmount&gt;</td>
<td>1500</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Number from GC2</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC2</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC2</td>
</tr>
<tr>
<td></td>
<td>&lt;originalTxnTime&gt;</td>
<td>txnTime from GC2</td>
</tr>
</tbody>
</table>
### TABLE 2-28  Private Label Gift Card Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC2B</td>
<td>Gift Card Credit:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt; Value from GC2A resp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;creditAmount&gt; 500</td>
<td>Credit Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Number from GC2</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 1220</td>
<td></td>
</tr>
<tr>
<td>CG3</td>
<td>Deactivate:</td>
<td>Deactivate Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Supplied by Implementation Consultant</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;availableBalance&gt; 0</td>
</tr>
<tr>
<td>GC4</td>
<td>Load:</td>
<td>Load Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 5000</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Supplied by Implementation Consultant</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0121</td>
<td>&lt;availableBalance&gt; 5000</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 123</td>
<td></td>
</tr>
<tr>
<td>GC5</td>
<td>Unload:</td>
<td>Unload Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 2500</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Number from GC4</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0121</td>
<td>&lt;availableBalance&gt; 2500</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 123</td>
<td></td>
</tr>
<tr>
<td>GC6</td>
<td>Balance Inquiry:</td>
<td>Balance Inquiry Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Number from GC1</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0121</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 123</td>
<td>&lt;availableBalance&gt; 2500</td>
</tr>
</tbody>
</table>
### TABLE 2-28  Private Label Gift Card Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Element</strong></td>
<td><strong>Value</strong></td>
</tr>
<tr>
<td>GC7</td>
<td><strong>Activate:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>8000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Supplied by Implementation Consultant</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1215</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td>GC7A</td>
<td><strong>Activate Reversal:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;littleTxnId&gt;</td>
<td>Value from GC7 resp. from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt;</td>
<td>systemTraceId from GC7</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt;</td>
<td>sequenceNumber from GC7</td>
</tr>
<tr>
<td>GC7B</td>
<td><strong>Activate Reversal:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;littleTxnId&gt;</td>
<td>Value from GC1A resp. from GC1</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>from GC1</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>from GC1</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>from GC1</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC1A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC1A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt;</td>
<td>systemTraceId from GC1A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt;</td>
<td>sequenceNumber from GC1A</td>
</tr>
<tr>
<td>orderId</td>
<td>Supplied Data Elements</td>
<td>Key Response Elements</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>GC8</td>
<td>Authorization:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Supplied by Implementation Consultant</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td>GC8A</td>
<td>Gift Card Auth Reversal:</td>
<td>Value from GC8 resp.</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt;</td>
<td>systemTraceId from GC8</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt;</td>
<td>sequenceNumber from GC8</td>
</tr>
<tr>
<td>GC9</td>
<td>Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Supplied by Implementation Consultant</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
</tbody>
</table>
### TABLE 2-28  Private Label Gift Card Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>GC9A</td>
<td>Deposit Reversal:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value from GC9 resp.</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt;</td>
<td>systemTraceId from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt;</td>
<td>sequenceNumber from GC9</td>
</tr>
<tr>
<td></td>
<td>Sale:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Number from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt;</td>
<td>123</td>
</tr>
<tr>
<td>GC10A</td>
<td>Gift Card Credit:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value from GC10 resp.</td>
</tr>
<tr>
<td></td>
<td>&lt;CreditAmount&gt;</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>GC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>Number from GC9</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>1220</td>
</tr>
<tr>
<td>GC10B</td>
<td>Refund Reversal:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt;</td>
<td>Value from GC10A resp.</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt;</td>
<td>from GC10</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>form GC10</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt;</td>
<td>from GC10</td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt;</td>
<td>refCode from GC10A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt;</td>
<td>amount from GC10A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt;</td>
<td>systemTraceId from GC10A</td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt;</td>
<td>sequenceNumber from GC10A</td>
</tr>
</tbody>
</table>
### TABLE 2-28 Private Label Gift Card Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC11</td>
<td>Deactivate Reversal:</td>
<td>Deactivate Reversal Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt; Value from GC3 resp.</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; from GC3</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; from GC3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; from GC3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt; refCode from GC3 resp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt; amount from GC3 resp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt; systemTraceId from GC3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt; sequenceNumber from GC3 resp.</td>
<td></td>
</tr>
<tr>
<td>GC12</td>
<td>Load Reversal:</td>
<td>Load Reversal Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;litleTxnId&gt; Value from GC4 resp.</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; from GC4</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; from GC4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; form GC4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalRefCode&gt; refCode from GC4 resp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalAmount&gt; amount from GC4 resp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalSystemTraceId&gt; systemTraceId from GC4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;originalSequenceNumber&gt; sequenceNumber from GC4 resp.</td>
<td></td>
</tr>
<tr>
<td>GC13</td>
<td>Unload:</td>
<td>Unload Response:</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 2500</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;type&gt; GC</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt; Supplied by Implementation Consultant</td>
<td>&lt;cardValidationResult&gt; M</td>
</tr>
<tr>
<td></td>
<td>&lt;expDate&gt; 0121</td>
<td>&lt;availableBalance&gt; 1500</td>
</tr>
<tr>
<td></td>
<td>&lt;cardValidationNum&gt; 123</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-28  
Private Label Gift Card Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
</table>
| **GC13A** | **Unload Reversal:**  
- `<litleTxnId>` Value from GC13 resp.  
- `<type>` from GC13  
- `<number>` from GC13  
- `<expDate>` from GC13  
- `<originalRefCode>` refCode from GC13 resp.  
- `<originalAmount>` amount from GC13 resp.  
- `<originalSystemTraceId>` systemTraceId from GC13 resp.  
- `<originalSequenceNumber>` sequenceNumber from GC13 resp. | **Unload Reversal Response:**  
- `<response>`  
- `<message>` Approved |
| **GC14** | **Activate:**  
- `<amount>` 15000  
- `<type>` GC  
- `<number>` Supplied by Implementation Consultant  
- `<expDate>` 1221  
- `<cardValidationNum>` 123 | **Activate Response:**  
- `<response>` 301  
- `<message>` Invalid Account Number |
| **GC15** | **Activate:**  
- `<amount>` 10000  
- `<type>` GC  
- `<number>` Number from GC1  
- `<expDate>` 1221  
- `<cardValidationNum>` 123 | **Activate Response:**  
- `<response>` 217  
- `<message>` Already active  
- `<availableBalance>` 14000 |
| **GC16** | **Authorization:**  
- `<amount>` 1500000  
- `<type>` GC  
- `<number>` Number from GC1  
- `<expDate>` 1221  
- `<cardValidationNum>` 123 | **Authorization Response:**  
- `<response>` 352  
- `<message>` Decline CVV/CID Fail  
- `<cardValidationResult>` M  
- `<availableBalance>` 14000 |
2.5.16 Testing MasterPass Transactions

If you are planning to support MasterPass transaction, in addition to other required certification testing, you should submit the following transactions to test the use of the additional cnpAPI elements required for a MasterPass transaction.

To test the MasterPass functionality using the data supplied in Table 2-29:

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>GC17</td>
<td>Authorization:</td>
<td>Authorization Response:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;amount&gt;</code>: 15000</td>
<td><code>&lt;response&gt;</code>: 218</td>
</tr>
<tr>
<td></td>
<td><code>&lt;type&gt;</code>: GC</td>
<td><code>&lt;message&gt;</code>: Card not active</td>
</tr>
<tr>
<td></td>
<td><code>&lt;number&gt;</code>: Supplied by Implementation Consultant</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;expDate&gt;</code>: 1221</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;cardValidationNum&gt;</code>: 123</td>
<td></td>
</tr>
<tr>
<td>GC19</td>
<td>Sale:</td>
<td>Sale Response:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;amount&gt;</code>: 1000</td>
<td><code>&lt;response&gt;</code>: 000</td>
</tr>
<tr>
<td></td>
<td><code>&lt;type&gt;</code>: GC</td>
<td><code>&lt;message&gt;</code>: Approved</td>
</tr>
<tr>
<td></td>
<td><code>&lt;number&gt;</code>: Supplied by Implementation Consultant</td>
<td><code>&lt;authCode&gt;</code>: 11111</td>
</tr>
<tr>
<td></td>
<td><code>&lt;expDate&gt;</code>: 1220</td>
<td><code>&lt;cardValidationResult&gt;</code>: M</td>
</tr>
<tr>
<td></td>
<td><code>&lt;cardValidationNum&gt;</code>: 123</td>
<td><code>&lt;availableBalance&gt;</code>: 1500</td>
</tr>
<tr>
<td>GC19A</td>
<td>Gift Card Credit:</td>
<td>Credit Response:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;littleTxnId&gt;</code>: Value from GC19 resp.</td>
<td><code>&lt;response&gt;</code>: 000</td>
</tr>
<tr>
<td></td>
<td><code>&lt;creditAmount&gt;</code>: 10000</td>
<td><code>&lt;message&gt;</code>: Approved</td>
</tr>
<tr>
<td></td>
<td><code>&lt;type&gt;</code>: GC</td>
<td>Declined Transaction report result = 304 - Lost/Stolen Card</td>
</tr>
<tr>
<td></td>
<td><code>&lt;number&gt;</code>: Value from GC19</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;expDate&gt;</code>: 1220</td>
<td></td>
</tr>
<tr>
<td>GC20</td>
<td>Balance Inquiry:</td>
<td>Balance Inquiry Response:</td>
</tr>
<tr>
<td></td>
<td><code>&lt;type&gt;</code>: GC</td>
<td><code>&lt;response&gt;</code>: 352</td>
</tr>
<tr>
<td></td>
<td><code>&lt;number&gt;</code>: Number from GC4</td>
<td><code>&lt;message&gt;</code>: Invalid CVV2</td>
</tr>
<tr>
<td></td>
<td><code>&lt;expDate&gt;</code>: 0121</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;cardValidationNum&gt;</code>: 476</td>
<td></td>
</tr>
</tbody>
</table>
1. Submit an Authorization transaction using the data supplied for Order Id MCWalletAuth. Verify that the response data matches the values for the Key Response Elements for that order Id.

2. Submit a Sale transaction using the data supplied for Order Id MCWalletSale. Verify that the response data matches the values for the Key Response Elements for that order Id.

### TABLE 2-29 MasterPass Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCWalletAuth</td>
<td>&lt;wallet&gt; (parent element)</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;walletSourceType&gt; MasterPass</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;walletSourceTypeId&gt; 102</td>
<td></td>
</tr>
<tr>
<td>MCWalletSale</td>
<td>&lt;wallet&gt; (parent element)</td>
<td>&lt;response&gt; 000</td>
</tr>
<tr>
<td></td>
<td>&lt;walletSourceType&gt; MasterPass</td>
<td>&lt;message&gt; Approved</td>
</tr>
<tr>
<td></td>
<td>&lt;walletSourceTypeld&gt; 102</td>
<td></td>
</tr>
</tbody>
</table>

### 2.5.17 Testing Apple Pay Transaction Processing

The following Apple Pay tests allow you to verify your submission of valid XML for two Apple Pay scenarios. The first test scenario applies to the case where you decrypt the Apple Pay PKPaymentToken, while the second scenario applies to the Worldpay decryption PKPaymentToken components submitted in a cnpAPI transaction.

**NOTE:** For information about testing the submission of the PKPaymentToken using the Mobile eProtect, please refer to the *Worldpay eComm eProtect Integration Guide*.

#### 2.5.17.1 Testing the Submission of the Decrypted PKPaymentToken in cnpAPI

To test the submission of an Apple Pay transaction in cnpAPI when you decrypt the PKPaymentToken, you must include the information listed in the table below in an Authorization or Sale transaction. Assuming you submit valid XML with appropriate values, the test environment will return an approved response message.

### TABLE 2-30 cnpAPI Elements for Merchant Decrypted PKPaymentToken

<table>
<thead>
<tr>
<th>cnpAPI Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>Use any valid (Mod-10 compliant) card number. You can find test numbers in Chapter 2 of the <em>Worldpay eComm cnpAPI Reference Guide</em>.</td>
</tr>
<tr>
<td>expDate</td>
<td>Use any valid expiration date (i.e., a date in the future).</td>
</tr>
</tbody>
</table>
2.5.17.2 Testing the Submission of PKPaymentToken in cnpAPI

When you receive a PKPaymentToken, you must parse (not decrypt) the component parts and use the data to populate the cnpAPI <applepay> child elements. This test is designed to allow you to verify your ability to parse the PKPaymentToken data and construct well formed cnpAPI messages. The test uses the data from the example PKPaymentToken shown below (data element names in bold type) to construct an Authorization or Sale transaction (see [applepay]). The system returns a Response Code of 000 - Approved.

Example: PKPaymentToken

```json
"version":"EC_v1","data":"Ww9EI+10VVpyZrAb3nxu9c8PG4JEnIh4oTkDhZi4axj5Qq5SW1ir6TJcFmk/3wkrNL/KaRXz2aan4W0RC6cL+uCUTPRqU09EQcTBItmqBqJxBn42713yI+y97k3ms17bd5rJOM1opCtfua+31nhBhJGPnUzdCqx","signature":"MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgEFADCABgkqhkiG9w0BAQsBAggwJQYIKwYBBQUHMAGGKWh0dHA6Ly9jcmwuY29tL2NybC9jem8wDgYDVR0PAQH/BAQDAgEGCCsGAQUFBw modulus 512-bit prime, p; public exponent 65537, e; private exponent 316179803791905210321174519...""
```

Note: The authenticationValue is Any Base-64 encoded value between 40 and 56 characters in length. This value simulates the cryptogram extracted in the PKPaymentToken.

<table>
<thead>
<tr>
<th>cnpAPI Element</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>authenticationValue</td>
<td>Any Base-64 encoded value between 40 and 56 characters in length. This value simulates the cryptogram extracted in the PKPaymentToken.</td>
</tr>
<tr>
<td>orderSource</td>
<td>Set to applepay</td>
</tr>
</tbody>
</table>

TABLE 2-30 cnpAPI Elements for Merchant Decrypted PKPaymentToken
2.5.18 Testing Google Pay Transaction Processing

The certification testing you need to perform depends upon the integration method you implement. This section provides information about the required tests for the eProtect integration method.

2.5.18.1 Testing Google Pay using eProtect

In this integration method, your application submits information to the Google test environment, receives a low-value token, and then submits the low value token to Worldpay in a normal payment transaction.

The table below provides key data values you provide in the submitted transaction.

1. Perform an implicit token registration by submitting a Sale or Authorization transaction using the data shown for the androidpay_1 (orderId) transaction in Table 1.
2. Verify the response message includes the key Response values shown.
3. Perform an explicit token registration by submitting a Register Token Request using the data shown for the androidpay_2 (orderId) transaction in Table 1.
4. Verify the response message includes the key Response values shown.

| Table 2-31 Google Pay using eProtect Test Data |
|-----------------------------|-----------------------------|-----------------------------|
| **orderId** | **Supplied Data Elements** | **Key Response Elements** |
| | **Element** | **Value** | **Element** | **Value** |
| androidpay_1 | <orderSource> | androidpay | <response> | 000 |
| | <paypageRegistrationId> | Value returned from Google | <message> | Approved |
2.5.19 Testing Amazon Pay

This section provides information about required Worldpay certification testing for merchants using Amazon Pay as a method of payment. Prior to performing the certification tests detailed in this document, you should complete all required integration with Amazon Pay and have access to their test environment.
To test Amazon Pay transactions:

1. Obtain a valid Amazon Pay token and submit an authorization transaction using the token and data from the Supplied Data Elements for Order Id AmznPay_1. Use any valid token for the litleToken value. Verify the transaction approves.

2. Submit a capture transaction for Order Id AmznPay_1A using the litleTxnId value returned in the response messages for Authorization Order Id AmznPay_1. Verify the transaction approves.

3. Submit a credit transaction for Order Id AmznPay_1B using the litleTxnId value returned in the response messages for Authorization Order Id AmznPay_1A. Verify the transaction approves.

4. Submit a void transaction for Order Id AmznPay_1C using the litleTxnId value returned in the response messages for Authorization Order Id AmznPay_1B. Verify the transaction approves.

5. Obtain a valid Amazon Pay token and submit a sale transaction using the token and data from the Supplied Data Elements for Order Id AmznPay_2. Use any valid token for the litleToken value. Verify the transaction approves.

6. Obtain an Amazon Pay token (from Amazon Pay) that generates a 560 Response Code and submit either an authorization or a sale transaction using the token and data from the Supplied Data Elements for Order Id AmznPay_560. Verify your system handles the 560 response code correctly.

7. Obtain an Amazon Pay token (from Amazon Pay) that generates a 561 Response Code and submit either an authorization or a sale transaction using the token and data from the Supplied Data Elements for Order Id AmznPay_561. Verify your system handles the 561 response code correctly.

8. Submit either an authorization or a sale transaction using the data from the Supplied Data Elements for Order Id AmznPay_562. Verify your system handles the 562 response code correctly.

9. Submit either an authorization or a sale transaction using the data from the Supplied Data Elements for Order Id AmznPay_563. Verify your system handles the 563 response code correctly.

10. Submit either an authorization or a sale transaction using the data from the Supplied Data Elements for Order Id AmznPay_110. Verify your system handles the 110 response code correctly.

### Table 2-32 Amazon Pay Test Data

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>AmznPay_1</td>
<td>Authorization: &lt;amount&gt;</td>
<td>10100</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;littleToken&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>xxxxxxxxxxxxxxxAMZN</td>
</tr>
<tr>
<td>AmznPay_1A</td>
<td>Capture: &lt;littleTxnId&gt;</td>
<td>Value returned in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Auth response for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Order Id AmznPay_1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 2-32  Amazon Pay Test Data (Continued)

<table>
<thead>
<tr>
<th>orderId</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>AmznPay_1B</td>
<td>Credit:</td>
<td>Value returned in Capture response for Order Id Amzn_1A</td>
</tr>
<tr>
<td></td>
<td>Void:</td>
<td>Use either the value returned in the Credit response for Order Id AmznPay_1B (not Auth).</td>
</tr>
<tr>
<td>AmznPay_2</td>
<td>Sale:</td>
<td>&lt;amount&gt; 10100 &lt;number&gt; xxxxxxxxxxxxxAMZN</td>
</tr>
</tbody>
</table>
2.5.20 Testing checkoutId

The checkoutId element is a low value token issued by eProtect in replacement for the CVV value. You only need to perform these test if you use eProtect and plan to use the checkoutId in place of the CVV.

To test the use of the checkoutId element, do the following:

The first test provides a response of a matching CVV, while the second and third test return checkoutId failure messages allowing you to verify your handling of these conditions. All tests return as Approved, because the CVV check does not necessarily impact the approval/decline of a transaction.

1. Using your (test) Web site, the token value returned for orderId 1 (Table 2-1) and the CVV value shown for orderId 1 to initiate an eProtect transaction. Use the checkoutId value you receive for CVV to construct an Authorization transaction within 24 hours. Verify the response data matches the information shown for orderId checkoutId_1 in Table 2-33.

2. Submit an Authorization using the same token value from Step 1 and the checkoutId value from the Supplied Data Elements for orderId checkoutId_2. Verify that the response data matches the Key response Elements for orderId checkoutId_2 in Table 2-33.

3. Submit an Authorization using the same token value from Step 1 and the checkoutId value from the Supplied Data Elements for orderId checkoutId_3. Verify that the response data matches the Key response Elements for orderId checkoutId_3 in Table 2-33.

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmznPay_110</td>
<td>Authorization:</td>
<td>Authorization</td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt; 11000</td>
<td>Response:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;litleToken&gt; 110000000000AMZN</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>checkoutId value from orderId_1</td>
<td>message</td>
<td>Insufficient funds</td>
</tr>
</tbody>
</table>

TABLE 2-32 Amazon Pay Test Data (Continued)

<table>
<thead>
<tr>
<th>orderid</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>chkoutId_1</td>
<td>&lt;litleToken&gt; value from orderld_1</td>
<td>&lt;response&gt; 000</td>
<td>000</td>
</tr>
<tr>
<td></td>
<td>&lt;checkoutId&gt; value from eProtect</td>
<td>&lt;message&gt; Approved</td>
<td>Approved</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;cardValidationResult&gt;M</td>
<td>M</td>
</tr>
</tbody>
</table>
2.5.21 Testing Transaction Volume Capacity

Volume testing is useful if you plan to send large files. This is an optional test you can perform during certification testing. Volume testing enables you to verify how many transactions (the number of requests and responses) you can process within a specific time frame.

Worldpay recommends you submit transactions for a 15-minute time interval. Submit the approximate number of transactions that you anticipate to be normal volume for any 15-minute period. You can send in any valid transaction data; the actual data you send will not be verified.
cnpAPI Transaction Examples

This chapter contains information and examples concerning the structure of cnpAPI transaction messages. Where differences exist between the structure of Batch and Online transactions, the sections present examples of both structures.

**NOTE:** In the cnpAPI, Worldpay enforces the order of the elements. Failure to adhere to the element order, as defined in the schema, will result in XML validation errors.

This chapter does not include examples of the PayFac Instruction-Based Dynamic Payout transaction types. For additional information about these transactions, please refer to Appendix D, "PayFac Dynamic Payout".

This chapter discusses the following topics:

- Overview of Online and Batch Processing Formats
- Transaction Types and Examples
3.1 Overview of Online and Batch Processing Formats

There are two methods of submitting payment transactions using the cnpAPI format: Online (one transaction at a time), or Batch. This section provides a high level overview of the request and response structures used for each submission type.

3.1.1 Batch Process Format

We treat each Batch transmission you send to us as a single request. You can think of the entire Batch request as a session composed of one or more individual batches, each containing one or more transactions. You can also use a Batch as a retrieval request for the response to a previously processed session. We return a response transmission for each request.

Batch processing supports the following transaction types:

- Activate Transactions
- Authorization Transactions
- Authorization Reversal
- Balance Inquiry Transactions
- Cancel Subscription Transactions
- Capture Transactions
- Capture Given Auth Transactions
- Create Plan Transactions
- Credit Transactions
- Deactivate Transactions
- eCheck Credit Transactions
- eCheck Prenotification Credit
- eCheck Prenotification Sale
- eCheck Redeposit Transactions
- eCheck Sale Transactions
- eCheck Verification Transactions
- Force Capture Transactions
- Fraud Check Transaction
- Gift Card Auth Reversal Transaction
- Gift Card Capture Transaction
- Gift Card Credit Transaction
- Load Transactions
- Register Token Transactions
- RFR Batch Transactions (Batch Only)
- Sale Transactions
- Unload Transactions
- Update Card Validation Number Transactions
- Update Plan Transactions
- Update Subscription Transactions
- All PayFac Dynamic Payout Transactions

NOTE: For information and examples of Dynamic Payout Funding Instructions, please refer to Appendix D, "PayFac Dynamic Payout".

For more information about Batch processing, see Batch Transaction Processing on page 5.
3.1.1.1 Supported Communication Protocols

We supports the following communication protocols for Batch processing:

- Encrypted FTP (PGP or GPG key encryption)
- sFTP

For additional information concerning the recommended transmission methods, see Transferring Session Files on page 86.

3.1.1.2 Batch Processing Request Format

The Batch processing request is made up of the following elements:

- Header information - one <litleRequest> element
- Merchant authentication information - one <authentication> element
- Batch information - one or more <batchRequest> elements
- Payment transactions (for example, an Authorization, Capture, Credit, etc.) - at least one per <batchRequest> element

**NOTE:** Each of the num and amount attributes used in a batchRequest defaults to 0 (zero) if not specified in the request. In any of the amount fields (authAmount, captureAmount, etc.), the value is an implied decimal (for example 500=5.00).

3.1.1.3 Batch Processing Response Format

The Batch processing response is composed of the following elements:

- Header information - one <litleResponse> element
- One or more <batchResponse> elements - contains payment transactions response.
- At least one payment transaction response (for example, an Authorization response, Capture response, Credit response, etc.).

**NOTE:** For information on the XML Validation response and message attributes, please refer to XML Validation Error Messages on page 829.
3.2 Online Processing Format

Each Online request you send to us is a single transaction. We process Online transactions upon receipt and return a response file.

**CAUTION:** If you use Open Access (i.e., Transact) to submit Online transactions, you must limit the message size to a maximum of 20K characters. We reject any transaction containing more than 20K characters with a XML validation response value of 5 (see Additional Response Header Error Messages on page 831).

Online processing supports the following transaction types:

- Activate Transactions
- Activate Reversal Transactions (Online Only)
- Authorization Transactions
- Authorization Reversal
- Balance Inquiry Transactions
- Cancel Subscription Transactions
- Capture Transactions
- Capture Given Auth Transactions
- Create Plan Transactions
- Credit Transactions
- Deactivate Transactions
- Deactivate Reversal Transactions (Online Only)
- Deposit Reversal Transactions (Online Only)
- Force Capture Transactions
- Load Transactions
- Load Reversal Transactions (Online Only)
- eCheck Sale Transactions
- eCheck Credit Transactions
- eCheck Redeposit Transactions
- eCheck Verification Transactions
- eCheck Void Transactions (Online Only)
- Refund Reversal Transactions (Online Only)
- Sale Transactions
- Status Query Transaction (Online Only)
-Unload Transactions
- Unload Reversal Transactions (Online Only)
- Update Card Validation Number Transactions
- Update Plan Transactions
3.2.1 **Supported Communication Protocols**

Worldpay supports the HTTPS POST communication protocol for Online processing. For additional information concerning the recommended transmissions methods, see [Transferring Online Files](#) on page 87.

3.2.2 **Online Processing Request Format**

The Online processing request is made up of the following elements:

- Header information - one `<litleOnlineRequest>` element
- Merchant authentication information - one `<authentication>` element
- Payment transaction - one payment transaction

3.2.3 **Online Processing Response Format**

An Online processing response is composed of the following elements:

- Header information - one `<litleOnlineResponse>`
- Payment transaction - one payment transaction

**NOTE:** For information on the XML Validation response and message attributes, please refer to [XML Validation Error Messages](#) on page 829.
3.3 Transaction Types and Examples

This section presents structural information of each transaction type for both Online and Batch submission methods. The structural information is followed by one or more examples of the cnpAPI transaction. Each structural example shows the parent and all child elements, but does not show grandchildren. The cnpAPI examples do show child elements to multiple levels.

The element names in the structural examples provide links to the element definitions in Chapter 4.

**NOTE:** The XML examples in this section are intended to present typical cnpAPI transactions. The examples may not include every possible element for a particular transaction type. When coding your XML, always consult the cnpAPI schema files for information concerning all available elements.

This section contains examples of the following transaction types:

**NOTE:** In the cnpAPI, the order of the elements is enforced. Failure to adhere to the element order as defined in the schema will result in XML validation errors.

- Authorization Transactions
- Authorization Reversal Transactions
- Activate Transactions (Private Label Gift Card transaction)
- Activate Reversal Transactions (Online Only) (Private Label Gift Card transaction)
- Balance Inquiry Transactions (Private Label Gift Card transaction)
- Cancel Subscription Transactions (Recurring Engine transaction)
- Capture Transactions
- Capture Given Auth Transactions
- Create Plan Transactions (Recurring Engine transaction)
- Credit Transactions
- Deactivate Transactions (Private Label Gift Card transaction)
- Deactivate Reversal Transactions (Online Only) (Private Label Gift Card transaction)
- Deposit Reversal Transactions (Online Only) (Private Label Gift Card transaction)
- eCheck Credit Transactions
- eCheck Prenotification Credit Transactions (Batch Only)
- eCheck Prenotification Sale Transactions (Batch Only)
- eCheck Redeposit Transactions
- eCheck Sale Transactions
- eCheck Verification Transactions
- eCheck Void Transactions (Online Only)
- Force Capture Transactions
• Fraud Check Transaction
• Gift Card Auth Reversal Transactions (Private Label Gift Card transaction)
• Gift Card Capture Transactions (Private Label Gift Card transaction)
• Gift Card Credit Transactions (Private Label Gift Card transaction)
• Load Transactions (Private Label Gift Card transaction)
• Load Reversal Transactions (Online Only) (Private Label Gift Card transaction)
• Status Query Transactions (Online Only)
• Refund Reversal Transactions (Online Only) (Private Label Gift Card transaction)
• Register Token Transactions
• RFR Transactions (Batch Only)
• Sale Transactions
• Unload Transactions (Private Label Gift Card transaction)
• Unload Reversal Transactions (Online Only) (Private Label Gift Card transaction)
• Update Plan Transactions (Recurring Engine transaction)
• Update Subscription Transactions (Recurring Engine transaction)
• Update Card Validation Number Transactions
• Void Transactions (Online Only)
### 3.3.1 Authorization Transactions

The Authorization transaction enables you to confirm that a customer has submitted a valid payment method with their order and has sufficient funds to purchase the goods or services they ordered.

The lifespan of an authorization varies according to the payment type being used, as shown in Table 3-1. During the lifespan, you can use a valid authorization multiple times as needed.

**NOTE:** To submit an AVS Only request, submit an Authorization request with the `<amount>` element set to 000. The response is identical to an Authorization response message.

**Table 3-1** Lifespan of a Payment Authorization

<table>
<thead>
<tr>
<th>Payment Type</th>
<th>Lifespan of Authorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Express</td>
<td>7 days</td>
</tr>
<tr>
<td>Discover</td>
<td>10 days</td>
</tr>
<tr>
<td>MasterCard</td>
<td>7 days</td>
</tr>
<tr>
<td>PayPal</td>
<td>29 days total; Worldpay recommends three days. For more information about PayPal authorizations, see the Worldpay PayPal Integration Guide.</td>
</tr>
<tr>
<td>Visa</td>
<td>7 days</td>
</tr>
</tbody>
</table>

This section describes the format you must use for an Authorization request, as well as the format of the Authorization Response you receive from us.

#### 3.3.1.1 Authorization Request Structure

You must structure an Authorization request as shown in the following examples. The structure of an Authorization request is identical for either an Online or a Batch submission.

```xml
<authorization id="Authorization Id" reportGroup="iQ Report Group" customerId="CustomerId">
  <orderId>Order Id</orderId>
  <amount>Authorization Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <orderSource>Order Entry Source</orderSource>
  <customerInfo>
    <billToAddress>
    <shipToAddress>
      [<card> | <paypal> | <paypage> | <token> | <mpos> | <applepay>]
    <billMeLaterRequest>
```
Example: Batch Authorization Request - Card Not Present

The example below shows a batch request with a single card-not-present Authorization request. If the batch included additional Authorization requests, each would have its own <authorization> element with all applicable attributes and child elements. Also, the numAuths attribute of the <batchRequest> element would increment for each additional <authorization> element and the authAmount attribute would increase by the new amounts from each authorization.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" numBatchRequests = "1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest numAuths="1" authAmount="2500" merchantId="000902">
    <authorization id="test1" reportGroup="core" customerId="test1">
      <orderId>visa_test1</orderId>
      <amount>2500</amount>
      <orderSource>telephone</orderSource>
      <billToAddress>
<name>John Doe</name>
<addressLine1>15 Main Street</addressLine1>
<city>San Jose</city>
<state>CA</state>
<zip>95032-1234</zip>
<country>USA</country>
<phone>9782750000</phone>
<email>jdoe@vantiv.com</email>
</billToAddress>

<shipToAddress>
<name>Jane Doe</name>
<addressLine1>15 Main Street</addressLine1>
<city>San Jose</city>
<state>CA</state>
<zip>95032-1234</zip>
<country>USA</country>
<phone>9782750000</phone>
<email>jdoe@vantiv.com</email>
</shipToAddress>

<card>
<type>VI</type>
<number>400555000081019</number>
<expDate>1110</expDate>
</card>

<customBilling>
<phone>8009990001</phone>
<descriptor>bdi*001</descriptor>
</customBilling>

<allowPartialAuth>true</allowPartialAuth>
</authorization>
</batchRequest>
</litleRequest>
**Example:** Batch Authorization Request - Card Present

The following example contains two Authorization requests, each defined in its own `<authorization>` element. The first is a card present transaction, which uses the `<track>` child of the `<card>` element.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numAuths="2" authAmount="68336" merchantId="100">
    <authorization id="AX54321678" reportGroup="RG27">
      <orderId>12z58743y1</orderId>
      <amount>12522</amount>
      <orderSource>retail</orderSource>
      <billToAddress>
        <zip>95032</zip>
      </billToAddress>
      <card>
        <track>%B40000001^Doe/JohnP^06041...?;40001=0604101064200?</track>
      </card>
      <pos>
        <capability>magstripe</capability>
        <entryMode>completeread</entryMode>
        <cardholderId>signature</cardholderId>
      </pos>
    </authorization>
    <authorization id="AX54325432" reportGroup="RG12">
      <orderId>12z58743y7</orderId>
      <amount>55814</amount>
      <orderSource>retail</orderSource>
      <billToAddress>
        <zip>01854</zip>
      </billToAddress>
      <card>
        <type>VI</type>
        <number>4005550000081019</number>
        <expDate>0911</expDate>
      </card>
      <pos>
        <capability>keyedonly</capability>
        <entryMode>keyed</entryMode>
        <cardholderId>directmarket</cardholderId>
      </pos>
      <allowPartialAuth>true</allowPartialAuth>
    </authorization>
  </batchRequest>
</litleRequest>
```
Example: Online Authorization Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
    merchantId="100">
    <authentication>
        <user>User Name</user>
        <password>Password</password>
    </authentication>
    <authorization id="834262" reportGroup="ABC Division" customerId="038945">
        <orderId>65347567</orderId>
        <amount>40000</amount>
        <orderSource>3dsAuthenticated</orderSource>
        <billToAddress>
            <name>John Smith</name>
            <addressLine1>100 Main St</addressLine1>
            <city>Boston</city>
            <state>MA</state>
            <zip>12345</zip>
            <email>jsmith@someaddress.com</email>
            <phone>555-123-4567</phone>
        </billToAddress>
        <card>
            <type>VI</type>
            <number>4000000000000002</number>
            <expDate>1209</expDate>
            <cardValidationNum>555</cardValidationNum>
        </card>
        <cardholderAuthentication>
            <authenticationValue>BwABBQl1gJDUCAAAAAA=</authenticationValue>
            <authenticationTransactionId>gMV75TmjAgk=</authenticationTransactionId>
        </cardholderAuthentication>
    </authorization>
</litleOnlineRequest>

NOTE: The example below uses 3dsAuthenticated as the <orderSource> value. If you submit the wrong <orderSource> value, we return the response code 370 - Internal System Error - Contact Litle.

Also, the values for the <authenticationValue> and <authenticationTransactionId> elements in the example below have been truncated.
Example: Authorization Request using token Element

The example below uses the following token related elements (click name to jump to element definition): token and littleToken.

```xml
<authorization id="99999" customerId="444" reportGroup="RG1">
  <orderId>F12345</orderId>
  <amount>15000</amount>
  <orderSource>telephone</orderSource>
  <billToAddress>
    <name>John Doe</name>
    <addressLine1>15 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
    <phone>9782750000</phone>
    <email>jdoe@vantiv.com</email>
  </billToAddress>
  <shipToAddress>
    <name>Jane Doe</name>
    <addressLine1>15 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
    <phone>9782750000</phone>
    <email>jdoe@vantiv.com</email>
  </shipToAddress>
  <token>
    <cnpToken>1111000101039449</cnpToken>
    <expDate>1112</expDate>
    <cardValidationNum>987</cardValidationNum>
  </token>
</authorization>
```

**NOTE:** When you submit the CVV2/CVC2/CID in a registerTokenRequest, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. To use the store value when submitting an Auth/Sale transaction, set the cardValidationNum value to 000.
Example: Authorization Request with Recurring Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <authorization id="834262" reportGroup="ABC Division" customerId="038945">
    <orderSource>3dsAuthenticated</orderSource>
    <orderId>65347567</orderId>
    <amount>40000</amount>
    <billToAddress>
      <name>John Smith</name>
      <addressLine1>100 Main St</addressLine1>
      <city>Boston</city>
      <state>MA</state>
      <zip>12345</zip>
      <email>jsmith@someaddress.com</email>
      <phone>555-123-4567</phone>
    </billToAddress>
    <card>
      <type>VI</type>
      <number>4000000000000002</number>
      <expDate>1209</expDate>
      <cardValidationNum>555</cardValidationNum>
    </card>
    <cardholderAuthentication>
      <authenticationValue>BwABBgQ1gJDUCAAAAAAA=</authenticationValue>
      <authenticationTransactionId>gMV75TmjAgk=</authenticationTransactionId>
    </cardholderAuthentication>
    <recurringRequest>
      <subscription>
        <planCode>Gold_Monthly_1</planCode>
        <numberOfPayments>12</numberOfPayments>
        <startDate>2016-07-21</startDate>
        <amount>1500</amount>
        <createDiscount>
          <discountCode>New_Customer_Discount_1</discountCode>
          <name>New_Customer</name>
          <amount>750</amount>
          <startDate>2016-07-21</startDate>
          <endDate>2016-07-22</endDate>
        </createDiscount>
        <createAddOn>
          <addOnCode>Extra_Feature_1</addOnCode>
        </createAddOn>
      </subscription>
    </recurringRequest>
  </authorization>
</litleOnlineRequest>
```
3.3.1.2 Authorization Response Structure

An Authorization response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<authorizationResponse id="Authorization Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <orderId>Order Id</orderId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <authCode>Approval Code</authCode>
  <approvedAmount>Approved amount for partial Auth</approvedAmount>
  <accountInformation>
  <accountUpdater>
  <fraudResult>
  <billMeLaterResponseData>
  <tokenResponse> (for Tokenized merchants submitting card data)
  <enhancedAuthResponse>
  <recycling> (included for declined Auths if feature is enabled)
  <recurringResponse> (for Recurring Engine merchants)
  <giftCardResponse> (included if Gift Card is Method of Payment)
  <applepayResponse>
  <cardSuffix>Card Last 4</cardSuffix> (included for ApplePay using VI or MC)
  <androidpayResponse>
  <networkTransactionId>Txn ID returned from network</networkTransactionId>
</authorizationResponse>
```
**Example: Batch Authorization Response**

The example below shows a batch Authorization response that contains two transactions.

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <authorizationResponse id="AX54321678" reportGroup="RG27">
      <cnpTxnId>84568456</cnpTxnId>
      <orderId>12z58743y1</orderId>
      <response>00</response>
      <responseTime>2018-03-01T10:24:31</responseTime>
      <message>Approved</message>
      <authCode>123456</authCode>
      <fraudResult>
        <avsResult>00</avsResult>
      </fraudResult>
      <networkTransactionId>123456789012345</networkTransactionId>
    </authorizationResponse>
    <authorizationResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <orderId>12z58743y7</orderId>
      <response>00</response>
      <responseTime>2018-03-01T10:24:31</responseTime>
      <message>Approved</message>
      <authCode>123456</authCode>
      <fraudResult>
        <avsResult>00</avsResult>
        <authenticationResult>2</authenticationResult>
      </fraudResult>
      <enhancedAuthResponse>
        <fundingSource>
          <type>PREPAID</type>
          <availableBalance>5000</availableBalance>
          <reloadable>NO</reloadable>
          <prepaidCardType>GIFT</prepaidCardType>
        </fundingSource>
        <networkTransactionId>123456789012346</networkTransactionId>
      </enhancedAuthResponse>
    </authorizationResponse>
  </batchResponse>
</litleResponse>
```
Example: Online Authorization Response including Advanced Fraud Results

NOTE: The online response format contains a <postDate> element, which indicates the date the financial transaction will post.

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <authorizationResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>969506</cnpTxnId>
    <orderId>65347567</orderId>
    <response>000</response>
    <postDate>2018-07-25</postDate>
    <message>Approved</message>
    <authCode>123457</authCode>
    <fraudResult>
      <avsResult>00</avsResult>
      <cardValidationResult>N</cardValidationResult>
      <authenticationResult>2</authenticationResult>
      <advancedFraudResults>
        <deviceReviewStatus>pass</deviceReviewStatus>
        <deviceReputationScore>70</deviceReputationScore>
        <triggeredRule>FlashImagesCookiesDisabled</triggeredRule>
      </advancedFraudResults>
    </fraudResult>
    <enhancedAuthResponse>
      <fundingSource>
        <type>PREPAID</type>
        <availableBalance>0</availableBalance>
        <reloadable>YES</reloadable>
        <prepaidCardType>TEEN</prepaidCardType>
      </fundingSource>
    </enhancedAuthResponse>
  </authorizationResponse>
</litleOnlineResponse>
```

Example: Authorization Response for Tokenized Merchant Sending Card Data

If a tokenized merchant submits card data in the Authorization request, the response includes the tokenResponse element. The example below is a response for an Online request (litleOnlineresponse element not shown); therefore, it includes the postDate element.

```xml
<authorizationResponse id="99999" reportGroup="RG1" customerId="444">
  <cnpTxnId>21200000001108</cnpTxnId>
</authorizationResponse>
```
Example: Online Authorization Response with Account Updater Token Info

In this example, the <accountUpdater> contains both original and new card information as well as the <extendedCardResponse> element. This signifies that the card number changed from the original to the new and (from the extended response code) that the issuer reported the new account as closed. Although the Authorization was approved, this information allows you to make an informed decision about how to proceed with the sale.
<type>VI</type>
<expDate>1114</expDate>
<bin>400555</bin>
</newCardTokenInfo>
<extendedCardResponse>
<code>501</code>
<message>The account was closed</message>
</extendedCardResponse>
</accountUpdater>
<fraudResult>
<avsResult>00</avsResult>
<cardValidationResult>N</cardValidationResult>
<authenticationResult>2</authenticationResult>
</fraudResult>
<networkTransactionId>123456789012345</networkTransactionId>
</authorizationResponse>
</litleOnlineResponse>

**Example:** Online Authorization Response with Recurring Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
<authorizationResponse id="834262" reportGroup="ABC Division">
  <cnpTxnId>969506</cnpTxnId>
  <orderId>65347567</orderId>
  <response>000</response>
  <postDate>2016-07-25</postDate>
  <message>Approved</message>
  <authCode>123457</authCode>
  <fraudResult>
    <avsResult>00</avsResult>
    <cardValidationResult>N</cardValidationResult>
    <authenticationResult>2</authenticationResult>
  </fraudResult>
  <recurringResponse>
    <subscriptionId>1234567890</subscriptionId>
    <responseCode>000</responseCode>
    <responseMessage>Approved</responseMessage>
  </recurringResponse>
</authorizationResponse>
</litleOnlineResponse>
```
3.3.2 Authorization Reversal Transactions

The Authorization Reversal transaction enables you to remove the hold on any funds being held by an Authorization. The original Authorization transaction must have been processed within the system. For information on how to use the Authorization Reversal transaction, see Notes on the Use of Authorization Reversal Transactions on page 70. Also, if you use our Recycling Engine, you can use the authReversal transaction to halt the recycling of an authorization transaction.


3.3.2.1 Authorization Reversal Requests

You must structure an Authorization Reversal request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```xml
<authReversal id="Authorization Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <amount>Authorization Amount to Reverse</amount>
  <actionReason>SUSPECT_FRAUD</actionReason>
</authReversal>
```

Example: Batch Authorization Reversal Request

The following example contains three Authorization Reversal Requests.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest numAuthReversals="3" authReversalAmount="3005" merchantId="000057">
    <authReversal id="ID" customerId="customerId" reportGroup="000057">
      <cnpTxnId>12345678</cnpTxnId>
      <amount>1002</amount>
    </authReversal>
    <authReversal id="ID" customerId="customerId" reportGroup="000057">
      <cnpTxnId>81234567</cnpTxnId>
      <amount>1003</amount>
    </authReversal>
    <authReversal id="ID" customerId="customerId" reportGroup="000057">
      <cnpTxnId>78123456</cnpTxnId>
    </authReversal>
  </batchRequest>
</litleRequest>
```

Example: Online Authorization Reversal Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <authReversal id="12345" customerId="Customer Id" reportGroup="Auth Reversals">
    <cnpTxnId>12345678</cnpTxnId>
    <amount>2999</amount>
  </authReversal>
</litleOnlineRequest>
```

3.3.2.2 Authorization Reversal Responses

The basic structure of an Authorization Reversal response is identical for Batch and Online responses.

```xml
<authReversalResponse id="Authorization Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
</authReversalResponse>
```

Example: Batch Authorization Reversal Response

This example shows a Batch Response containing three Authorization Reversal responses.

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" response="0"
message="ValidFormat" cnpSessionId="21500040809">
  <batchResponse cnpBatchId="21500040908" merchantId="000902">
    <authReversalResponse id="ID" reportGroup="core" customerId="Rev1">
      <cnpTxnId>21200000002700</cnpTxnId>
      <response>000</response>
      <message>Approved</message>
    </authReversalResponse>
    <authReversalResponse id="ID" reportGroup="core" customerId="viRev2">
      <cnpTxnId>21200000002700</cnpTxnId>
      <response>000</response>
      <message>Approved</message>
    </authReversalResponse>
  </batchResponse>
</litleResponse>
```
Example: Online Authorization Reversal Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
    <authReversalResponse id="12345" customerId="Customer Id" reportGroup="Auth Reversals">
        <cnpTxnId>12345678</cnpTxnId>
        <response>000</response>
        <message>Approved</message>
    </authReversalResponse>
</litleOnlineResponse>
```

### 3.3.3 Activate Transactions

The Activate transaction changes the status of a (Closed Loop) Gift Card from inactive to active with a value as specified by the `amount` element. You can also use this transaction type to create a Virtual Gift Card.

**NOTE:** You must be enabled for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

#### 3.3.3.1 Activate Request

You must structure an Activate request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```
<activate id="Activate Id" reportGroup="iQ Report Group" customerId="Customer Id">
```

Example: Activate Request - Gift Card

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <activate id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>65347567</orderId>
    <amount>40000</amount>
    <orderSource>ecommerce</orderSource>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
    </card>
  </activate>
</litleOnlineRequest>
```

Example: Activate Request - Virtual Gift Card

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <activate id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>65347567</orderId>
    <amount>40000</amount>
    <orderSource>ecommerce</orderSource>
    <virtualGiftCard>
      <accountNumberLength>16</accountNumberLength>
      <giftCardBin>900000</giftCardBin>
    </virtualGiftCard>
  </activate>
</litleOnlineRequest>
```
### Activate Response

An Activate response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<activateResponse id="Activate Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <fraudResult>
    <giftCardResponse>
      <virtualGiftCardResponse>
      </virtualGiftCardResponse>
    </giftCardResponse>
  </fraudResult>
</activateResponse>
```

**Example: Activate Response - Gift Card**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <activateResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <postDate>2017-01-25</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2016-11-25T15:13:38</txnTime>
      <refCode>123456</refCode>
      <systemTraceId>123456</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>5000</availableBalance>
    </giftCardResponse>
  </activateResponse>
</litleOnlineResponse>
```

**Example: Activate Response - Virtual Gift Card**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <activateResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
  </activateResponse>
</litleOnlineResponse>
```
3.3.4 Activate Reversal Transactions (Online Only)

The Activate Reversal transaction changes the status of a newly activated Gift Card from active to inactive, thus reversing the operation of an Active transaction. The Activate Reversal references the associated Activate transaction by means of the littleTxnId element returned in the Activate response. This transaction type is available only for Online transactions.

**NOTE:** Worldpay must enable for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

3.3.4.1 Activate Reversal Request

You must structure an Activate Reversal request as shown in the following examples.

```xml
<activateReversal id="Activate Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id from Activate Response</litleTxnId>
  <card>
    <virtualGiftCardBin>
      <originalRefCode>Reference Code from Activate Response</originalRefCode>
      <originalAmount>Amount from Activate Transaction</originalAmount>
      <originalTxnTime>Transaction Time from Activate Response</originalTxnTime>
      <originalSystemTraceId>Trace Id from Activate Response</originalSystemTraceId>
      <originalSequenceNumber>Seq Num from Activate Rsp</originalSequenceNumber>
    </virtualGiftCardBin>
  </card>
</activateReversal>
```
Example: Activate Reversal Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.little.com/schema"
merchantId="100">
    <authentication>
        <user>User Name</user>
        <password>Password</password>
    </authentication>
    <activateReversal id="12345" customerId="Customer Id" reportGroup="Activate Reversals">
        <cnpTxnId>12345678901234567890</cnpTxnId>
        <card>
            <type>GC</type>
            <number>1234102000003558</number>
            <cardValidationNum>888</cardValidationNum>
            <pin>1234</pin>
        </card>
        <originalRefCode>123456</originalRefCode>
        <originalAmount>1900</originalAmount>
        <originalTxnTime>2017-03-21T10:02:46</originalTxnTime>
        <originalSystemTraceId>678901</originalSystemTraceId>
        <originalSequenceNumber>123456</originalSequenceNumber>
    </activateReversal>
</litleOnlineRequest>
```

3.3.4.2 Activate Reversal Response

An Activate Reversal response has the following structure.

```xml
<activateReversalResponse id="Activate Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <postDate>Date transaction posted</postDate> (Online Only)
    <message>Response Message</message>
    <giftCardResponse/>
</activateReversalResponse>
```
Example: Activate Reversal Response

```
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <activateReversalResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T12:00:00</txnTime>
      <refCode>03558</refCode>
      <systemTraceId>834528</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>0</availableBalance>
    </giftCardResponse>
  </activateReversalResponse>
</litleOnlineResponse>
```

3.3.5 Balance Inquiry Transactions
A Balance Inquiry transaction returns the available balance of a Gift Card.

NOTE: Worldpay must enable for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

3.3.5.1 Balance Inquiry Request
You must structure an Balance Inquiry request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```
<balanceInquiry id="Balance Inquiry Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <orderId>Order Id</orderId>
  <orderSource>Order Entry Source</orderSource>
</card>
</balanceInquiry>
```

Example: Balance Inquiry Request

```
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <!-- Authentication details -->
  </authentication>
</litleOnlineRequest>
```
<user>User Name</user>
<password>Password</password>
</authentication>

<balanceInquiry id="834262" reportGroup="ABC Division" customerId="038945">
  <orderId>65347567</orderId>
  <orderSource>ecommerce</orderSource>
  <card>
    <type>GC</type>
    <number>9000000000000001</number>
    <pin>1234</pin>
  </card>
</balanceInquiry>
</litleOnlineRequest>

### 3.3.5.2 Balance Inquiry Response

An Balance Inquiry response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<balanceInquiryResponse id="Balance Inquiry Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <orderId>Order Id</orderId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <fraudResult>
  </balanceInquiryResponse>
```

**Example: Balance Inquiry Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <balanceInquiryResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <orderId>65347567</orderId>
    <response>000</response>
    <responseTime>2017-03-21T15:13:43</responseTime>
    <postDate>2017-03-21</postDate>
    <message>Approved</message>
    <fraudResult>
  </balanceInquiryResponse>
```
3.3.6 Cancel Subscription Transactions

The Cancel Subscription transaction cancels the specified subscription, removing the transaction stream managed by the Recurring Engine.

3.3.6.1 Cancel Subscription Request

You must structure an Cancel Subscription request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```
<cancelSubscription>
  <subscriptionId>ID of Subscription to Cancel</subscriptionId>
</cancelSubscription>
```

Example: Cancel Subscription Request

```
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <cancelSubscription>
    <subscriptionId>1234</subscriptionId>
  </cancelSubscription>
</litleOnlineRequest>
```

3.3.6.2 Cancel Subscription Response

A Cancel Subscription response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```
<cancelSubscriptionResponse>
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <message>Response Message</message>
  <responseTime>Date and Time in GMT</responseTime>
</cancelSubscriptionResponse>
```
3.3.7 Capture Transactions

The Capture transaction transfers funds from the customer to the merchant. The Capture references the associated Authorization by means of the litleTxnId element returned in the Authorization response.

NOTE: To perform a capture on a Closed-loop Gift Card, submit a Gift Card Capture transaction, not an Capture transaction. See Gift Card Capture Transactions on page 272 for additional information.

You send a Capture after the order has been fulfilled. In some cases, it is not possible to fulfill a customer’s entire order in one shipment (for example, if some items are backordered, or some shipped from an off-site DCS). In this situation, you can send a Partial Capture transaction by setting the partial attribute to true. A Partial Capture contains only the data relevant to the items that were actually shipped, enabling you to settle the funds related to those items.

NOTE: If you settle in a currency other than US or Canada, you cannot use partial captures.

3.3.7.1 Capture Request

You must structure a Capture request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```xml
<capture id="Capture Id" reportGroup="iQ Report Group" customerId="Customer Id" partial="false">
  <litleTxnId>Transaction Id</litleTxnId>
  <amount>Authorization Amount</amount>
</capture>
```
Example: Batch Capture Request - Full Capture

The following Capture example is for a full capture. Although the `<capture>` element includes an `<amount>` child, it is not required for a full Capture. If you omit the `<amount>` child element, the capture amount defaults to the full amount from the associated Authorization.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numAuths="0" authAmount="0" numCaptures="1" captureAmount="55814" numCredits="0" creditAmount="0" numSales="0" saleAmount="0" merchantId="100">
    <capture id="AX54325432" reportGroup="RG12" partial="false">
      <cnpTxnId>84568457</cnpTxnId>
      <amount>55814</amount>
      <enhancedData>
        <customerReference>P012346</customerReference>
        <salesTax>1500</salesTax>
        <taxExempt>false</taxExempt>
        <discountAmount>0</discountAmount>
        <shippingAmount>3714</shippingAmount>
        <dutyAmount>0</dutyAmount>
        <shipFromPostalCode>01851</shipFromPostalCode>
        <destinationPostalCode>01851</destinationPostalCode>
        <destinationCountryCode>USA</destinationCountryCode>
        <invoiceReferenceNumber>123456</invoiceReferenceNumber>
        <orderDate>2011-09-14</orderDate>
        <detailTax>
          <taxIncludedInTotal>true</taxIncludedInTotal>
          <taxAmount>500</taxAmount>
          <taxRate>0.01667</taxRate>
          <taxTypeIdentifier>00</taxTypeIdentifier>
          <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
        </detailTax>
        <lineItemData>
          <itemSequenceNumber>1</itemSequenceNumber>
        </lineItemData>
      </enhancedData>
      <payPalOrderComplete>Set to true for final Capture</payPalOrderComplete>
      <pin>Gift Card Pin Number</pin>
    </capture>
  </batchRequest>
</litleRequest>
```
Example: Batch Capture Request - Partial Capture

A partial Capture has the partial attribute set to true and must include an amount child element.

```xml
<litRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numAuths="0" authAmount="0" numCaptures="1">
    <itemDescription>table</itemDescription>
    <productCode>TB123</productCode>
    <quantity>1</quantity>
    <unitOfMeasure>EACH</unitOfMeasure>
    <taxAmount>1500</taxAmount>
    <lineItemTotal>30000</lineItemTotal>
    <lineItemTotalWithTax>31500</lineItemTotalWithTax>
    <itemDiscountAmount>0</itemDiscountAmount>
    <commodityCode>301</commodityCode>
    <unitCost>300.00</unitCost>
    <detailTax>
      <taxIncludedInTotal>true</taxIncludedInTotal>
      <taxAmount>500</taxAmount>
      <taxRate>0.01667</taxRate>
      <taxTypeIdentifier>03</taxTypeIdentifier>
      <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
    </detailTax>
  </itemItemData>
  <itemItemData>
    <itemSequenceNumber>2</itemSequenceNumber>
    <itemDescription>chair</itemDescription>
    <productCode>CH123</productCode>
    <quantity>1</quantity>
    <unitOfMeasure>EACH</unitOfMeasure>
    <lineItemTotal>20000</lineItemTotal>
    <itemDiscountAmount>0</itemDiscountAmount>
    <commodityCode>301</commodityCode>
    <unitCost>200.00</unitCost>
  </itemItemData>
  </enhancedData>
</batchRequest>
</litRequest>
```
captureAmount="45814" numCredits="0" creditAmount="0" numSales="0"
saleAmount="0" merchantId="100">
  <capture id="AX54325432" reportGroup="RG12" partial="true">
    <cnpTxnId>84568457</cnpTxnId>
    <amount>45814</amount>
    <enhancedData>
      <customerReference>PO12346</customerReference>
      <salesTax>2100</salesTax>
      <taxExempt>false</taxExempt>
      <discountAmount>0</discountAmount>
      <shippingAmount>3714</shippingAmount>
      <dutyAmount>0</dutyAmount>
      <shipFromPostalCode>01851</shipFromPostalCode>
      <destinationPostalCode>01851</destinationPostalCode>
      <destinationCountryCode>USA</destinationCountryCode>
      <invoiceReferenceNumber>123456</invoiceReferenceNumber>
      <orderDate>2016-09-14</orderDate>
      <detailTax>
        <taxIncludedInTotal>true</taxIncludedInTotal>
        <taxAmount>500</taxAmount>
        <taxRate>0.01667</taxRate>
        <taxTypeIdentifier>00</taxTypeIdentifier>
        <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
      </detailTax>
      <lineItemData>
        <itemSequenceNumber>1</itemSequenceNumber>
        <itemDescription>table</itemDescription>
        <productCode>TB123</productCode>
        <quantity>1</quantity>
        <unitOfMeasure>EACH</unitOfMeasure>
        <taxAmount>1500</taxAmount>
        <lineItemTotal>30000</lineItemTotal>
        <lineItemTotalWithTax>31500</lineItemTotalWithTax>
        <itemDiscountAmount>0</itemDiscountAmount>
        <commodityCode>301</commodityCode>
        <unitCost>300.00</unitCost>
        <detailTax>
          <taxIncludedInTotal>true</taxIncludedInTotal>
          <taxAmount>500</taxAmount>
          <taxRate>0.01667</taxRate>
          <taxTypeIdentifier>03</taxTypeIdentifier>
          <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
        </detailTax>
      </lineItemData>
      <lineItemData>
        <itemDiscountAmount>0</itemDiscountAmount>
      </lineItemData>
    </enhancedData>
  </capture>
</cnpTxnId>
Example: Online Capture Request - Full Capture

The following Capture example is for a full capture. Although the `<capture>` element includes an `<amount>` child, it is not required for a full Capture. If you omit the `<amount>` child element, the capture amount defaults to the full amount from the associated Authorization.

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <capture id="2" reportGroup="ABC Division" customerId="038945" partial="false">
    <cnpTxnId>13254123434</cnpTxnId>
    <enhancedData>
      <customerReference>PO12345</customerReference>
      <salesTax>125</salesTax>
      <taxExempt>false</taxExempt>
      <discountAmount>0</discountAmount>
      <shippingAmount>495</shippingAmount>
      <dutyAmount>0</dutyAmount>
      <shipFromPostalCode>01851</shipFromPostalCode>
      <destinationPostalCode>01851</destinationPostalCode>
      <destinationCountryCode>USA</destinationCountryCode>
      <invoiceReferenceNumber>123456</invoiceReferenceNumber>
      <orderDate>2011-08-14</orderDate>
      <detailTax>
        <taxIncludedInTotal>true</taxIncludedInTotal>
        <taxAmount>55</taxAmount>
        <taxRate>0.0059</taxRate>
      </detailTax>
    </enhancedData>
  </capture>
</litleOnlineRequest>
```
<taxTypeIdentifier>00</taxTypeIdentifier>
<cardAcceptortaxId>011234567</cardAcceptortaxId>
</detailTax>

<lineItemData>
    <itemSequenceNumber>1</itemSequenceNumber>
    <itemDescription>chair</itemDescription>
    <productCode>CH123</productCode>
    <quantity>1</quantity>
    <unitOfMeasure>EACH</unitOfMeasure>
    <taxAmount>125</taxAmount>
    <lineItemTotal>9380</lineItemTotal>
    <lineItemTotalWithTax>9505</lineItemTotalWithTax>
    <itemDiscountAmount>0</itemDiscountAmount>
    <commodityCode>300</commodityCode>
    <unitCost>93.80</unitCost>
    <detailTax>
        <taxIncludedInTotal>true</taxIncludedInTotal>
        <taxAmount>55</taxAmount>
        <taxRate>0.0059</taxRate>
        <taxTypeIdentifier>03</taxTypeIdentifier>
        <cardAcceptortaxId>011234567</cardAcceptortaxId>
    </detailTax>
</lineItemData>

<lineItemData>
    <itemSequenceNumber>2</itemSequenceNumber>
    <itemDescription>table</itemDescription>
    <productCode>TB123</productCode>
    <quantity>1</quantity>
    <unitOfMeasure>EACH</unitOfMeasure>
    <lineItemTotal>30000</lineItemTotal>
    <itemDiscountAmount>0</itemDiscountAmount>
    <commodityCode>300</commodityCode>
    <unitCost>300.00</unitCost>
</lineItemData>
</enhancedData>
</capture>
</litleOnlineRequest>

**Example:** Online Capture Request - Partial Capture

A partial Capture has the partial attribute set to true and must include an <amount> child element.

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
    merchantId="100">
    <authentication>
```

```xml
</litleOnlineRequest>
```
<user>User Name</user>
<password>password</password>
</authentication>

<capture id="2" reportGroup="ABC Division" customerId="038945" partial="true">
  <cnpTxnId>13254123434</cnpTxnId>
  <amount>100</amount>
  <enhancedData>
    <customerReference>PO12345</customerReference>
    <salesTax>125</salesTax>
    <taxExempt>false</taxExempt>
    <discountAmount>0</discountAmount>
    <shippingAmount>495</shippingAmount>
    <dutyAmount>0</dutyAmount>
    <shipFromPostalCode>01851</shipFromPostalCode>
    <destinationPostalCode>01851</destinationPostalCode>
    <destinationCountryCode>USA</destinationCountryCode>
    <invoiceReferenceNumber>123456</invoiceReferenceNumber>
    <orderDate>2011-08-14</orderDate>
    <detailTax>
      <taxIncludedInTotal>true</taxIncludedInTotal>
      <taxAmount>55</taxAmount>
      <taxRate>0.0059</taxRate>
      <taxTypeIdentifier>00</taxTypeIdentifier>
      <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
    </detailTax>
    <lineItemData>
      <itemSequenceNumber>1</itemSequenceNumber>
      <itemDescription>chair</itemDescription>
      <productCode>CH123</productCode>
      <quantity>1</quantity>
      <unitOfMeasure>EACH</unitOfMeasure>
      <taxAmount>125</taxAmount>
      <lineItemTotal>9380</lineItemTotal>
      <lineItemTotalWithTax>9505</lineItemTotalWithTax>
      <itemDiscountAmount>0</itemDiscountAmount>
      <commodityCode>300</commodityCode>
      <unitCost>93.80</unitCost>
      <detailTax>
        <taxIncludedInTotal>true</taxIncludedInTotal>
        <taxAmount>55</taxAmount>
        <taxRate>0.0059</taxRate>
        <taxTypeIdentifier>03</taxTypeIdentifier>
        <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
      </detailTax>
    </lineItemData>
  </enhancedData>
</capture>
3.3.7.2 Capture Response

A Capture response has the following structure. The response message is identical for Online and Batch transactions except Batch always includes the <orderId> element, while Online includes the <postDate> element.

```xml
<captureResponse id="Authorization Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <postDate>Date Of Posting</postDate> (Online Only)
    <message>Response Message</message>
    <accountUpdater>
</captureResponse>
```

**Example: Batch Capture Response**

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema"  id="123" cnpSessionId="987654321" response="0" message="Valid Format">
    <batchResponse id="01234567" cnpBatchId="4455667788"  merchantId="100">
        <captureResponse id="AX54321678" reportGroup="RG27">
            <cnpTxnId>84568456</cnpTxnId>
            <response>000</response>
            <responseTime>2017-04-01T10:24:31</responseTime>
            <message>Approved</message>
        </captureResponse>
        <captureResponse id="AX54325432" reportGroup="RG12">
```
Example:  Online Capture Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <captureResponse id="2" reportGroup="ABC Division" customerId="038945">
    <cnpTxnId>1100030204</cnpTxnId>
    <response>000</response>
    <postDate>2016-07-11</postDate>
    <message>Approved</message>
  </captureResponse>
</litleOnlineResponse>
```

### 3.3.8 Capture Given Auth Transactions

You typically use a Capture Given Auth transaction when the associated Authorization occurred outside of the system (for example, if you received a telephone Authorization). Another possible use for a Capture Given Auth transaction is if the Authorization transaction occurred within the system, but the `<litleTxnId>` is unknown by the submitting party (for example, if the Auth was submitted by a merchant, but a fulfiller submits a Capture Given Auth).

#### 3.3.8.1 Capture Given Auth Request

You must specify the Capture Given Auth request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<captureGivenAuth id="Capture Given Auth Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <authInformation>
    <amount>Authorization Amount</amount>
    <secondaryAmount>Secondary Amount</secondaryAmount>
    <orderSource>Order Entry Source</orderSource>
  </billToAddress>
  <shipToAddress>
```
Example: Batch Capture Given Auth Request

The following example shows a single Capture Given Auth request. The example uses the `<card>` child element, but a tokenized merchant could use the `<token>` child element in its place.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema"
numBatchRequests="1">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <batchRequest id="batchId" numCaptureGivenAuths="1"
captureGivenAuthAmount="10000" merchantId="100">
    <captureGivenAuth id="AX54321678" reportGroup="RG27">
      <orderId>orderId</orderId>
      <authInformation>
        <authDate>2016-09-21</authDate>
        <authCode>123456</authCode>
      </authInformation>
      <amount>10000</amount>
      <orderSource>ecommerce</orderSource>
      <card>
        <type>VI</type>
        <number>4005550000081019</number>
        <expDate>0910</expDate>
      </card>
      <enhancedData>
        <customerReference>PO12345</customerReference>
      </enhancedData>
    </captureGivenAuth>
  </batchRequest>
</litleRequest>
```
Example: Online Capture Given Auth Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <captureGivenAuth id="AX54321678" reportGroup="RG27">
    <orderId>orderId</orderId>
  </captureGivenAuth>
</litleRequest>
```
<authInformation>
  <authDate>2016-08-24</authDate>
  <authCode>123456</authCode>
</authInformation>

<amount>10000</amount>
<orderSource>ecommerce</orderSource>
<card>
  <type>VI</type>
  <number>400555000081019</number>
  <expDate>0910</expDate>
</card>
<enhancedData>
  <customerReference>PO12345</customerReference>
  <salesTax>125</salesTax>
  <deliveryType>TBD</deliveryType>
  <taxExempt>false</taxExempt>
  <discountAmount>0</discountAmount>
  <shippingAmount>495</shippingAmount>
  <dutyAmount>0</dutyAmount>
  <shipFromPostalCode>01851</shipFromPostalCode>
  <destinationPostalCode>01851</destinationPostalCode>
  <destinationCountryCode>USA</destinationCountryCode>
  <invoiceReferenceNumber>123456</invoiceReferenceNumber>
  <orderDate>2016-08-14</orderDate>
  <detailTax>
    <taxIncludedInTotal>true</taxIncludedInTotal>
    <taxAmount>55</taxAmount>
    <taxRate>0.0059</taxRate>
    <taxTypeIdentifier>00</taxTypeIdentifier>
    <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
  </detailTax>
  <lineItemData>
    <itemSequenceNumber>1</itemSequenceNumber>
    <itemDescription>chair</itemDescription>
    <productCode>CH123</productCode>
    <quantity>1</quantity>
    <unitOfMeasure>EACH</unitOfMeasure>
    <taxAmount>125</taxAmount>
    <lineItemTotal>9380</lineItemTotal>
    <lineItemTotalWithTax>9505</lineItemTotalWithTax>
    <itemDiscountAmount>0</itemDiscountAmount>
    <commodityCode>300</commodityCode>
    <unitCost>93.80</unitCost>
  </lineItemData>
</enhancedData>
Example: Capture Given Auth Request using token Element

The example below uses the following token related elements (click name to jump to element definition): token and litleToken.

```xml
<captureGivenAuth id="99999" customerId="444" reportGroup="RG1">
  <orderId>F12345</orderId>
  <authInformation>
    <authDate>2011-10-25</authDate>
    <authCode>500005</authCode>
  </authInformation>
  <amount>15000</amount>
  <orderSource>ecommerce</orderSource>
  <billToAddress>
    <name>John Doe</name>
    <addressLine1>10 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
  </billToAddress>
  <token>
    <cnpToken>1112000100010085</cnpToken>
    <expDate>1112</expDate>
    <cardValidationNum>987</cardValidationNum>
  </token>
</captureGivenAuth>
```

### 3.3.8.2 Capture Given Auth Response

A Capture Given Auth response has the following structure. The response message is identical for Online and Batch transactions except Online includes the <postDate> element.

```xml
<captureGivenAuthResponse id="Capture Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date of Posting</postDate> (Online Only)
  <message>Response Message</message>
</captureGivenAuthResponse>
```
<tokenResponse>  (for Tokenized merchants submitting card data)
</captureGivenAuthResponse>

**Example:** Batch Capture Given Auth Response

```xml
<ltleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123"
    cnpSessionId="987654321" response="0" message="Valid Format">
    <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
        <captureGivenAuthResponse id="AX54325432" reportGroup="RG12">
            <cnpTxnId>84568457</cnpTxnId>
            <response>000</response>
            <responseTime>2017-04-01T10:24:31</responseTime>
            <message>Approved</message>
        </captureGivenAuthResponse>
    </batchResponse>
</ltleResponse>
```

**Example:** Online Capture Given Auth Response

```xml
<ltleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
    response="0" message="Valid Format">
    <captureGivenAuthResponse id="2" reportGroup="ABC Division" customerId="038945">
        <cnpTxnId>1100030204</cnpTxnId>
        <response>000</response>
        <postDate>2017-04-11</postDate>
        <message>Approved</message>
    </captureGivenAuthResponse>
</ltleOnlineResponse>
```

**Example:** Capture Given Auth Response for Tokenized Merchant Sending Card Data

```xml
<captureGivenAuthResponse id="99999" reportGroup="RG1" customerId="444">
    <cnpTxnId>21200000022005</cnpTxnId>
    <response>000</response>
    <responseTime>2017-04-25T04:00:00</responseTime>
    <postDate>2017-04-26</postDate>
    <message>Approved</message>
    <tokenResponse>
        <cnpToken>1111000100409510</cnpToken>
        <tokenResponseCode>801</tokenResponseCode>
        <tokenMessage>Account number was successfully registered</tokenMessage>
        <type>VI</type>
        <bin>432610</bin>
    </tokenResponse>
</captureGivenAuthResponse>
```
3.3.9 Create Plan Transactions

The Create Plan transaction allows you to define several attributes of a recurring payment schedule. Later, you can associate existing, active plans with subscriptions, which establishes a recurring payment schedule managed by the Recurring Engine.

3.3.9.1 Create Plan Request

You must specify the Create Plan transaction as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<createPlan>
  <planCode>Plan Reference Code</planCode>
  <name>Name of Plan</name>
  <description>Description of Plan</description>
  <intervalType>The Type of Interval</intervalType>
  <amount>Amount of Recurring Payment</amount>
  <numberOfPayments>1 to 99</numberOfPayments>
  <trialNumberOfIntervals>Number of Trial Intervals</trialNumberOfIntervals>
  <trialIntervalType>Type of Trial Period Interval</trialIntervalType>
  <active>true or false</active>
</createPlan>
```

Example: Online Create Plan Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <createPlan>
    <planCode>Gold12</planCode>
    <name>Gold_Monthly</name>
    <description>Gold Level with Monthly Payments</description>
    <intervalType>MONTHLY</intervalType>
    <amount>5000</amount>
    <numberOfPayments>4</numberOfPayments>
    <trialNumberOfIntervals>1</trialNumberOfIntervals>
    <trialIntervalType>MONTH</trialIntervalType>
    <active>true</active>
</createPlan>
```

3.3.9.2 Create Plan Response

The Create Plan response message is identical for Online and Batch transactions.

```xml
<createPlanResponse>
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <message>Response Message</message>
  <responseTime>Date and Time in GMT</responseTime>
  <planCode>Plan Reference Code</subscriptionId>
</createPlanResponse>
```

**Example: Online Create Plan Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <createPlanResponse>
    <cnpTxnId>1100030055</cnpTxnId>
    <response>000</response>
    <message>Approved</message>
    <planCode>Gold12</planCode>
  </createPlanResponse>
</litleOnlineResponse>
```

3.3.10 Credit Transactions

The Credit transaction enables you to refund money to a customer, even if the original transaction occurred outside of our system. You can submit refunds against any of the following payment transactions:

- Capture Transactions
- Capture Given Auth Transactions
- Force Capture Transactions
- Sale Transactions

**NOTE:** To perform a credit on a Closed-loop Gift Card, submit a Gift Card Credit transaction, not an Credit transaction. See Gift Card Credit Transactions on page 273 for additional information.
3.3.10.1 Credit Request for a Worldpay Processed Transaction

You must specify the Credit request for a Worldpay processed transaction as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<credit id="Credit Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <amount>Authorization Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <customBilling>
    <enhancedData>
      <processingInstructions>
        <actionReason>SUSPECT_FRAUD</actionReason>
      </processingInstructions>
    </enhancedData>
  </customBilling>
</credit>
```

**Example:** Batch Credit Request for a Worldpay Processed Transaction

To request a Credit against a sale settled by us, you only need to specify the `<litleTxnId>` element. The application uses the `<litleTxnId>` to look-up the Capture referenced and obtain all the necessary information including the amount.

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numAuths="0" authAmount="0" numCaptures="0" captureAmount="0" numCredits="1" creditAmount="10000" numSales="0" saleAmount="0" merchantId="100">
    <credit id="AX54321678" reportGroup="RG27">
      <cnpTxnId>84568456</cnpTxnId>
      <amount>10000</amount>
      <enhancedData>
      </enhancedData>
    </credit>
  </batchRequest>
</litleRequest>
```

**NOTE:** Although it is not required, if you choose to include `<amount>` elements in your Credit transaction, you must include the total amount in the `creditAmount` attribute of the `<batchRequest>`. If you do not specify amounts, set the `creditAmount` attribute to 0.
<customerReference>PO12345</customerReference>
<salesTax>125</salesTax>
<taxExempt>false</taxExempt>
<discountAmount>0</discountAmount>
<shippingAmount>3017</shippingAmount>
<dutyAmount>0</dutyAmount>
<shipFromPostalCode>01851</shipFromPostalCode>
<destinationPostalCode>01851</destinationPostalCode>
<destinationCountryCode>USA</destinationCountryCode>
<invoiceReferenceNumber>123456</invoiceReferenceNumber>
<orderDate>2016-09-14</orderDate>
<detailTax>
<taxIncludedInTotal>true</taxIncludedInTotal>
<taxAmount>55</taxAmount>
<taxRate>0.0059</taxRate>
<taxTypeIdentifier>00</taxTypeIdentifier>
<cardAcceptorTaxId>011234567</cardAcceptorTaxId>
</detailTax>
<lineItemData>
<ItemSequenceNumber>1</ItemSequenceNumber>
<ItemDescription>chair</ItemDescription>
<ProductCode>CH123</ProductCode>
<Quantity>1</Quantity>
<UnitOfMeasure>EACH</UnitOfMeasure>
<TaxAmount>125</TaxAmount>
<LineItemTotal>9380</LineItemTotal>
<LineItemTotalWithTax>9505</LineItemTotalWithTax>
<ItemDiscountAmount>0</ItemDiscountAmount>
<CommodityCode>300</CommodityCode>
<UnitCost>93.80</UnitCost>
<detailTax>
<TaxIncludedInTotal>true</TaxIncludedInTotal>
<TaxAmount>55</TaxAmount>
<TaxRate>0.0059</TaxRate>
<TaxTypeIdentifier>03</TaxTypeIdentifier>
<CardAcceptorTaxId>011234567</CardAcceptorTaxId>
</detailTax>
</lineItemData>
</enhancedData>
</credit>
</batchRequest>
</litleRequest>
**Example:** Online Credit Request for a Worldpay Processed Transaction

To request a Credit against a sale settled by us, you need only specify the `<litleTxnId>` element. The application uses the `<litleTxnId>` to look up the Capture referenced and obtain all the necessary information including the amount. The example below includes the optional `<customBilling>` and `<enhancedData>` elements.

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <credit id="2" reportGroup="ABC Division" customerId="038945">
    <cnpTxnId>13254123434</cnpTxnId>
    <customBilling>
      <phone>8888888888</phone>
      <descriptor>descriptor</descriptor>
    </customBilling>
    <enhancedData>
      <customerReference>PO12345</customerReference>
      <salesTax>125</salesTax>
      <taxExempt>false</taxExempt>
      <discountAmount>0</discountAmount>
      <shippingAmount>495</shippingAmount>
      <dutyAmount>0</dutyAmount>
      <shipFromPostalCode>01851</shipFromPostalCode>
      <destinationPostalCode>01851</destinationPostalCode>
      <destinationCountryCode>USA</destinationCountryCode>
      <invoiceReferenceNumber>123456</invoiceReferenceNumber>
      <orderDate>2016-07-14</orderDate>
      <detailTax>
        <taxIncludedInTotal>true</taxIncludedInTotal>
        <taxAmount>55</taxAmount>
        <taxRate>0.0059</taxRate>
        <taxTypeIdentifier>00</taxTypeIdentifier>
        <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
      </detailTax>
      <lineItemData>
        <itemSequenceNumber>1</itemSequenceNumber>
        <itemDescription>chair</itemDescription>
        <productCode>CH123</productCode>
        <quantity>1</quantity>
        <unitOfMeasure>EACH</unitOfMeasure>
        <taxAmount>125</taxAmount>
        <lineItemTotal>9380</lineItemTotal>
        <lineItemTotalWithTax>9505</lineItemTotalWithTax>
      </lineItemData>
    </enhancedData>
  </credit>
</litleOnlineRequest>
```
3.3.10.2 Credit Request for a Non-Worldpay Processed Transaction

You must specify the Credit request for a Non-Worldpay processed transaction as follows. The structure of the request is identical for either an Online or a Batch submission.

```
<credit id="Credit Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Authorization Amount</amount>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
```

**NOTE:** Although the schema shows `<paypal>` as an optional element for a Credit against a non-Worldpay processed transaction, refunds of this type are not supported for PayPal. If you need to refund non-Worldpay processed transactions and have not maintained a temporary relationship with your former processor for this purpose, please ask your Relationship Manager for alternative options.
Example: Batch Credit Request for a Non-Worldpay Processed Transaction

If the sale occurred outside of our system, you must specify the following elements in your Credit request: <orderId>, <amount>, and <card>, or <token> (<paypal> not supported for this transaction type).

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numCredits="1" creditAmount="10000"
               merchantId="100">
    <credit id="AX54321678" reportGroup="RG27">
      <orderId>12z58743y1</orderId>
      <amount>10000</amount>
      <orderSource>ecommerce</orderSource>
      <billToAddress>
        <name>John Doe</name>
        <addressLine1>123 4th street</addressLine1>
        <addressLine2>Apt. 20</addressLine2>
        <addressLine3>second floor</addressLine3>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95032</zip>
        <country>USA</country>
        <email>jdoe@isp.com</email>
        <phone>408-555-1212</phone>
      </billToAddress>
      <card>
        <type>MC</type>
      </card>
    </credit>
  </batchRequest>
</litleRequest>
```
<number>518600580001012</number>
<expDate>1110</expDate>
</card>
<enhancedData>
<customerReference>PO12345</customerReference>
<salesTax>125</salesTax>
<taxExempt>false</taxExempt>
<discountAmount>0</discountAmount>
<shippingAmount>495</shippingAmount>
<dutyAmount>0</dutyAmount>
<shipFromPostalCode>01851</shipFromPostalCode>
<destinationPostalCode>01851</destinationPostalCode>
<destinationCountryCode>USA</destinationCountryCode>
<invoiceReferenceNumber>123456</invoiceReferenceNumber>
<orderDate>2016-09-14</orderDate>
<detailTax>
<taxIncludedInTotal>true</taxIncludedInTotal>
<taxAmount>55</taxAmount>
<taxRate>0.0059</taxRate>
<taxTypeIdentifier>00</taxTypeIdentifier>
<cardAcceptorTaxId>011234567</cardAcceptorTaxId>
</detailTax>
<lineItemData>
(itemSequenceNumber>1</itemSequenceNumber>
(itemDescription>chair</itemDescription>
(productCode>CH123</productCode>
(quantity>1</quantity>
(unitOfMeasure>EACH</unitOfMeasure>
<taxAmount>125</taxAmount>
(lineItemTotal>9380</lineItemTotal>
(lineItemTotalWithTax>9505</lineItemTotalWithTax>
(itemDiscountAmount>0</itemDiscountAmount>
(commodityCode>300</commodityCode>
(unitCost>93.80</unitCost>
</lineItemData>
</enhancedData>
</credit>
</batchRequest>
</litleRequest>
Example: Online Credit Request for a Non-Worldpay Processed Transaction

If the sale occurred outside of our system, you must specify the following elements in your Credit request: <orderId>, <amount>, and <card>, or <token> (<paypal> not supported for this transaction type).

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
    merchantId="100">
    <authentication>
        <user>User Name</user>
        <password>password</password>
    </authentication>
    <credit id="2" reportGroup="ABC Division" customerId="038945">
        <orderId>56789</orderId>
        <amount>10000</amount>
        <orderSource>ecommerce</orderSource>
        <billToAddress>
            <name>Mike J. Hammer</name>
            <addressLine1>Two Main Street</addressLine1>
            <addressLine2>Apartement 222</addressLine2>
            <addressLine3></addressLine3>
            <city>Riverside</city>
            <state>RI</state>
            <zip>02915</zip>
            <country>US</country>
            <email>mike@sample.com</email>
            <phone>5555555555</phone>
        </billToAddress>
        <card>
            <type>VI</type>
            <number>400555000081019</number>
            <expDate>0907</expDate>
        </card>
        <customBilling>
            <phone>5555555555</phone>
            <descriptor>descriptor</descriptor>
        </customBilling>
        <enhancedData>
            <customerReference>PO12345</customerReference>
            <salesTax>125</salesTax>
            <taxExempt>false</taxExempt>
            <discountAmount>0</discountAmount>
            <shippingAmount>495</shippingAmount>
            <dutyAmount>0</dutyAmount>
            <shipFromPostalCode>01851</shipFromPostalCode>
            <destinationPostalCode>01851</destinationPostalCode>
        </enhancedData>
    </credit>
</litleOnlineRequest>
```
<destinationCountryCode>USA</destinationCountryCode>
<invoiceReferenceNumber>123456</invoiceReferenceNumber>
<orderDate>2011-08-14</orderDate>
<detailTax>
<taxIncludedInTotal>true</taxIncludedInTotal>
<taxAmount>55</taxAmount>
<taxRate>0.0059</taxRate>
<taxTypeIdentifier>00</taxTypeIdentifier>
<cardAcceptorTaxId>011234567</cardAcceptorTaxId>
</detailTax>
<lineItemData>
(itemSequenceNumber>1</itemSequenceNumber>
(itemDescription>chair</itemDescription>
(productCode>CH123</productCode>
(quantity>1</quantity>
(unitOfMeasure>EACH</unitOfMeasure>
<taxAmount>125</taxAmount>
<lineItemTotal>9380</lineItemTotal>
<lineItemTotalWithTax>9505</lineItemTotalWithTax>
<itemDiscountAmount>0</itemDiscountAmount>
<commodityCode>300</commodityCode>
<unitCost>93.80</unitCost>
</lineItemData>
<lineItemData>
(itemSequenceNumber>2</itemSequenceNumber>
(itemDescription>table</itemDescription>
(productCode>TB123</productCode>
(quantity>1</quantity>
(unitOfMeasure>EACH</unitOfMeasure>
<lineItemTotal>30000</lineItemTotal>
<itemDiscountAmount>0</itemDiscountAmount>
<commodityCode>300</commodityCode>
<unitCost>300.00</unitCost>
</lineItemData>
</enhancedData>
</credit>
</litleOnlineRequest>

### 3.3.10.3 Credit Response

The Credit response message is identical for Online and Batch transactions except Online includes the postDate element.

<creditResponse id="Credit Id" reportGroup="iQ Report Group" customerId="Customer Id">
Example: Credit Response

The example below illustrates a Batch Credit response. A response for an Online transaction uses a litleOnlineResponse element as the parent.

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" cnpSessionId="987654321" response="0" message="Valid Format">
    <batchResponse id="01234567" cnpBatchId="445567788" merchantId="100">
        <creditResponse id="AX54325432" reportGroup="RG12">
            <cnpTxnId>84568457</cnpTxnId>
            <response>000</response>
            <responseTime>2017-04-01T10:24:31</responseTime>
            <message>Approved</message>
        </creditResponse>
    </batchResponse>
</litleResponse>
```

Example: Credit Response for a Tokenized Merchant Sending Card Data

When a tokenized merchant submits card data in the Credit request, the response includes the tokenResponse element. The example below is a response for an Online request (<litleOnlineResponse> element not shown), so it includes the <postDate> element.

```xml
<creditResponse id="99999" reportGroup="RG1" customerId="444">
    <cnpTxnId>21200000473208</cnpTxnId>
    <response>000</response>
    <responseTime>2017-04-19T19:29:45</responseTime>
    <postDate>2017-04-19</postDate>
    <message>Approved</message>
    <tokenResponse>
        <cnpToken>1111102200202001</cnpToken>
        <tokenResponseCode>801</tokenResponseCode>
        <tokenMessage>Account number was successfully registered</tokenMessage>
        <type>VI</type>
        <bin>402410</bin>
    </tokenResponse>
</creditResponse>
```
3.3.11 Deactivate Transactions

The Deactivate transaction changes the status of a (Closed Loop) Gift Card from active to inactive.

NOTE: Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

3.3.11.1 Deactivate Request

You must structure a Deactivate request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```
<deactivate id="Activate Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <orderSource>Order Entry Source</orderSource>
</deactivate>
```

Example: Online Deactivate Request

```
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <deactivate id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>65347567</orderId>
    <orderSource>ecommerce</orderSource>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
      <pin>1234</pin>
    </card>
  </deactivate>
</litleOnlineRequest>
```
3.3.11.2 Deactivate Response

A Deactivate response has the following structure. The response message is identical for Online and Batch transactions except Online includes the <postDate> element.

```xml
<deactivateResponse id="Deactivate Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <fraudResult>
    <approvedAmount>5000</approvedAmount>
  </fraudResult>
</deactivateResponse>
```

**Example:** Online Deactivate Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <deactivateResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <postDate>2016-07-25</postDate>
    <message>Approved</message>
    <approvedAmount>5000</approvedAmount>
    <giftCardResponse>
      <availableBalance>0</availableBalance>
      <beginningBalance>5000</beginningBalance>
      <endingBalance>0</endingBalance>
    </giftCardResponse>
  </deactivateResponse>
</litleOnlineResponse>
```

3.3.12 Deactivate Reversal Transactions (Online Only)

The Deactivate Reversal transaction changes the status of a newly deactivated Gift Card from inactive to active, thus reversing the operation of an Deactivate transaction. The Deactivate Reversal references the associated Deactivate transaction by means of the litleTxnId element returned in the Deactivate response. This transaction type is available only for Online transactions.
3.3.12.1 Deactivate Reversal Request

You must structure an Deactivate Reversal request as shown in the following examples.

```
<deactivateReversal id="Deactivate Reversal Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id from Load Response</litleTxnId>
  <card>
    <originalRefCode>Reference Code from Deactivate Response</originalRefCode>
    <originalAmount>Amount from Deactivate Transaction</originalAmount>
    <originalTxnTime>Transaction Time from Deactivate Response</originalTxnTime>
    <originalSystemTraceId>Trace Id from Deactivate Resp</originalSystemTraceId>
    <originalSequenceNumber>Seq Num from Deactivate Resp</originalSequenceNumber>
  </card>
</deactivateReversal>
```

**Example:** Deactivate Reversal Request

```
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <deactivateReversal id="12345" customerId="Customer Id" reportGroup="Deactivate Reversals">
    <cnpTxnId>1234567890123456789</cnpTxnId>
    <card>
      <type>GC</type>
      <number>1234102000003558</number>
      <cardValidationNum>888</cardValidationNum>
      <pin>1234</pin>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>1900</originalAmount>
    <originalTxnTime>2017-03-21T10:02:46</originalTxnTime>
    <originalSystemTraceId>678901</originalSystemTraceId>
    <originalSequenceNumber>123456</originalSequenceNumber>
  </deactivateReversal>
</litleOnlineRequest>
```

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.
### 3.3.12.2 Deactivate Reversal Response

An Deactivate Reversal response has the following structure.

```xml
<deactivateReversalResponse id="Deactivate Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <postDate>Date transaction posted</postDate>
    <message>Response Message</message>
</deactivateReversalResponse>
```

**Example: Online Deactivate Reversal Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
    <deactivateReversalResponse id="834262" reportGroup="ABC Division">
        <cnpTxnId>9695064321</cnpTxnId>
        <response>000</response>
        <responseTime>2017-03-22T15:13:43</responseTime>
        <postDate>2017-03-22</postDate>
        <message>Approved</message>
        <giftCardResponse>
            <txnTime>2017-03-22T12:00:00</txnTime>
            <refCode>003558</refCode>
            <systemTraceId>834528</systemTraceId>
            <sequenceNumber>123456</sequenceNumber>
            <availableBalance>5000</availableBalance>
        </giftCardResponse>
    </deactivateReversalResponse>
</litleOnlineResponse>
```

### 3.3.13 Deposit Reversal Transactions (Online Only)

The Deposit Reversal transaction is a Gift Card transaction that reverses the deposit initiated by either a Capture or Sale transaction. The Deposit Reversal references the associated Capture/Sale transaction by means of the litleTxnId element returned in the Capture/Sale response. This transaction type is available only for Online transactions.
3.3.13.1 Deposit Reversal Request

You must structure a Deposit Reversal request as shown in the following examples.

```xml
<depositReversal id="Deposit Reversal Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id from Load Response</litleTxnId>
  <card>
    <originalRefCode>Reference Code from Gift Card Capture Resp</originalRefCode>
    <originalAmount>Amount from Gift Card Capture Resp</originalAmount>
    <originalTxnTime>Transaction Time from GC Capture Response</originalTxnTime>
    <originalSystemTraceId>Trace Id from GC Capture Resp</originalSystemTraceId>
    <originalSequenceNumber>Seq Num from GC Capture Resp</originalSequenceNumber>
  </card>
</depositReversal>
```

**Example: Deposit Reversal Request**

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <depositReversal id="12345" customerId="Customer Id" reportGroup="Deposit Reversals">
    <cnpTxnId>1234567890123456789</cnpTxnId>
    <card>
      <type>GC</type>
      <number>1234102000003558</number>
      <cardValidationNum>888</cardValidationNum>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>1900</originalAmount>
    <originalTxnTime>2017-03-21T10:02:46</originalTxnTime>
    <originalSystemTraceId>678901</originalSystemTraceId>
    <originalSequenceNumber>123456</originalSequenceNumber>
  </depositReversal>
</litleOnlineRequest>
```

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.
3.3.13.2 Deposit Reversal Response

An Deposit Reversal response has the following structure.

```xml
<depositReversalResponse id="Deactivate Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate>
  <message>Response Message</message>
  <giftCardResponse>
    <txnTime>2017-03-22T12:00:00</txnTime>
    <refCode>003558</refCode>
    <systemTraceId>834528</systemTraceId>
    <sequenceNumber>123456</sequenceNumber>
    <availableBalance>5000</availableBalance>
  </giftCardResponse>
</depositReversalResponse>
```

Example: Online Deposit Reversal Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <depositReversalResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T12:00:00</txnTime>
      <refCode>003558</refCode>
      <systemTraceId>834528</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>5000</availableBalance>
    </giftCardResponse>
  </depositReversalResponse>
</litleOnlineResponse>
```

3.3.14 eCheck Credit Transactions

The eCheck Credit transaction enables you to refund a previous eCheck Sale. Merchants can submit an eCheck Credit transaction for a sale regardless or whether the original transaction was settled by our system, although the requests are structured differently. This section contains the following:

- eCheck Credit Request Against a Worldpay Transaction
- eCheck Credit Request for a Non-Worldpay Processed Sale
- eCheck Credit Response
3.3.14.1 eCheck Credit Request Against a Worldpay Transaction

To request an eCheck Credit against an eCheck Sale that had been settled by us, you only need to specify the `<litleTxnId>` element. When you specify this element, the application uses the `<litleTxnId>` to look up the referenced echeckSale transaction and obtain the necessary information. In this case, the `<amount>` element is optional, but should be included if the credit amount is less than the captured amount. If you do not include the `<amount>` element, the system assumes the credit to be for the total amount of the referenced transaction.

When requesting a echeckCredit against an echeckSale that occurred within our system, specify the Credit request as follows:

```xml
<echeckCredit id="Credit Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <amount>Credit Amount</amount>
    <secondaryAmount>Secondary Amount</secondaryAmount>
    <customBilling> (Do not use)</customBilling>
    <customIdentifier>
</echeckCredit>
```

Example: eCheck Credit Request

The eCheck Credit batch request shown below contains three `<echeckCredit>` elements. The first two use a Transaction Id as a reference. The third, which you would use for a sale occurring outside of our system, uses the `<orderId>`, `<amount>`, `<billToAddress>`, and `<echeck>` elements to provide the required information.

```xml
<litleRequest version="11.3" xmlns="http://www.litle.com/schema" numBatchRequests="1">
    <authentication>
        <user>userName</user>
        <password>password</password>
    </authentication>
    <batchRequest id="xmlbat01" numEcheckCredit="3" echeckCreditAmount="12100" merchantId="000053">
        <echeckCredit id="credit1" reportGroup="new53" customerId="53">
            <cnpTxnId>4455667788</cnpTxnId>
            <amount>1000</amount>
        </echeckCredit>
        <echeckCredit reportGroup="new53">
            <cnpTxnId>4455667789</cnpTxnId>
            <amount>1100</amount>
        </echeckCredit>
        <echeckCredit reportGroup="new53">
            <cnpTxnId>4455667789</cnpTxnId>
            <amount>1100</amount>
        </echeckCredit>
    </batchRequest>
</litleRequest>
```

**NOTE:** Although there are two different scenarios for eCheck Credit requests, the response message uses the same structure.
<echeckCredit reportGroup="new53">
  <orderId>12z58743y1</orderId>
  <amount>10000</amount>
  <orderSource>ecommerce</orderSource>
  <billToAddress>
    <name>John Doe</name>
    <addressLine1>123 4th street</addressLine1>
    <addressLine2>Apt. 20</addressLine2>
    <addressLine3>second floor</addressLine3>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032</zip>
    <country>USA</country>
    <email>jdoe@isp.com</email>
    <phone>408-555-1212</phone>
  </billToAddress>
  <echeck>
    <accType>Checking</accType>
    <accNum>5186005800001012</accNum>
    <routingNum>00010101</routingNum>
    <checkNum>1104</checkNum>
  </echeck>
</echeckCredit>

### 3.3.14.2 eCheck Credit Request for a Non-Worldpay Processed Sale

If the original eCheck Sale transaction was not processed via our system, or if the `<litleTxnId>` for the original eCheck Sale transaction is not available, specify eCheck Credit request as follows:

```xml
<echeckCredit id="eCheckCredit Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Credit Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
    <echeck> or <echeckToken>
      <customBilling> (Do not use)
  <merchantData>
</echeckCredit>
```
The third transaction shown in the eCheck Credit Request example on page 245 shows an example of a Credit request against a non-Worldpay processed transaction.

### 3.3.14.3 eCheck Credit Response

The eCheck Credit message is identical for either type of eCheck Credit request. The `<accountUpdater>` element is included only if you submit account information in the request transaction for which a NOC exists. In this case the system automatically updates the information sent to the ACH network and includes the change information in the response.

The eCheck Credit response has the following structure:

```xml
<echeckCreditResponse id="eCheck Credit Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <message>Response Message</message>
    <postDate>Date of Posting</postDate> (Online Only)
    <accountUpdater>Account Change Info</accountUpdater>
    <tokenResponse> (for Tokenized merchants submitting account data)
</echeckCreditResponse>
```

**Example: eCheck Credit Response**

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
    <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
        <echeckCreditResponse id="AX54321678" reportGroup="RG27" customerId="53">
            <cnpTxnId>84568456</cnpTxnId>
            <response>000</response>
            <responseTime>2017-04-01T10:24:31</responseTime>
            <message>Approved</message>
        </echeckCreditResponse>
    </batchResponse>
</litleResponse>
```

### 3.3.15 eCheck Prenotification Credit Transactions (Batch Only)

You use this non-monetary transaction to verify the consumer's account information prior to submitting an eCheck Credit transaction (also see eCheck Prenotification on page 48). This transaction type is only supported for US transactions.
3.3.15.1 eCheck Prenotification Credit Request

You must specify the eCheck Prenotification Credit request using the following format:

```xml
<echeckPreNoteCredit id="eCheck PreNote Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
    <echeck>
      <merchantData>
      </echeckPreNoteCredit>
  </echeck>
</echeckPreNoteCredit>
```

**Example: eCheck Prenotification Credit Request**

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>

  <batchRequest id="654321" numEcheckPreNoteCredit="1" merchantId="100">
    <echeckPreNoteCredit id="AX54321678" reportGroup="RG27" customerId="53">
      <orderId>12z58743y1</orderId>
      <orderSource>telephone</orderSource>
      <billToAddress>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <addressLine1>123 4th street</addressLine1>
        <addressLine2>Apt. 20</addressLine2>
        <addressLine3>second floor</addressLine3>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95032</zip>
        <country>USA</country>
        <email>jdoe@isp.com</email>
        <phone>408-555-1212</phone>
      </billToAddress>
      <echeck>
        <accType>Checking</accType>
        <accNum>5186005800001012</accNum>
        <routingNum>000010101</routingNum>
        <checkNum>1104</checkNum>
      </echeck>
    </echeckPreNoteCredit>
  </batchRequest>
</litleRequest>
```
3.3.15.2 eCheck Prenotification Credit Response

The eCheck Prenotification Credit response has the following structure:

```xml
<echeckPreNoteCreditResponse id="eCheckSale Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <orderId>Order Id</orderId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
</echeckPreNoteCreditResponse>
```

**Example: eCheck Prenotification Credit Response**

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <echeckPreNoteCreditResponse id="AX54321678" reportGroup="RG27" customerId="53">
      <cnpTxnId>84568456</cnpTxnId>
      <orderId>12z58743y1</orderId>
      <response>000</response>
      <responseTime>2017-03-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckPreNoteCreditResponse>
    <echeckPreNoteCreditResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <orderId>12z58743y7</orderId>
      <response>000</response>
      <responseTime>2017-03-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckPreNoteCreditResponse>
  </batchResponse>
</litleResponse>
```
3.3.16 eCheck Prenotification Sale Transactions (Batch Only)

You use this non-monetary transaction to verify the consumer’s account information prior to submitting an eCheck Sale transaction (also see eCheck Prenotification on page 48). This transaction type is only supported for US transactions.

3.3.16.1 eCheck Prenotification Sale Request

You must specify the eCheck Prenotification Credit request using the following format:

```xml
<echeckPreNoteSale id="eCheck PreNote Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <orderId>Order Id</orderId>
    <orderSource>Order Entry Source</orderSource>
    <billToAddress>
        <echeck>
            <merchantData>
            </echeck>
        </billToAddress>
</echeckPreNotesale>
```

**Example:** eCheck Prenotification Sale Request

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
    <authentication>
        <user>userName</user>
        <password>password</password>
    </authentication>
    <batchRequest id="654321" numEcheckPreNoteSale="1" merchantId="100">
        <echeckPreNoteSale id="AX54321678" reportGroup="RG27" customerId="53">
            <orderId>12z58743y1</orderId>
            <orderSource>telephone</orderSource>
            <billToAddress>
                <firstName>John</firstName>
                <lastName>Doe</lastName>
                <addressLine1>123 4th street</addressLine1>
                <addressLine2>Apt. 20</addressLine2>
                <addressLine3>second floor</addressLine3>
                <city>San Jose</city>
                <state>CA</state>
                <zip>95032</zip>
                <country>USA</country>
                <email>jdoe@isp.com</email>
                <phone>408-555-1212</phone>
            </billToAddress>
        </echeck>
    </batchRequest>
</litleRequest>
```
3.3.16.2 eCheck Prenotification Sale Response

The eCheck Prenotification Sale response has the following structure:

```xml
<echeckPreNoteSaleResponse id="eCheckSale Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <orderId>Order Id</orderId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
</echeckPreNoteSaleResponse>
```

**Example: eCheck Prenotification Sale Response**

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="445567788" merchantId="100">
    <echeckPreNoteSaleResponse id="AX54321678" reportGroup="RG27" customerId="53">
      <cnpTxnId>84568456</cnpTxnId>
      <orderId>12z58743y1</orderId>
      <response>000</response>
      <responseTime>2017-03-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckPreNoteSaleResponse>
    <echeckPreNoteSaleResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <orderId>12z58743y7</orderId>
      <response>000</response>
      <responseTime>2017-03-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckPreNoteSaleResponse>
  </batchResponse>
</litleResponse>
```
3.3.17 eCheck Redeposit Transactions

You use this transaction type to manually attempt redeposits of Direct Debits returned for either Insufficient Funds or Uncollected Funds. You can use this element in either Batch or Online transactions.

**NOTE:** Do not use this transaction type if you are enabled for the Auto Redeposit feature. If you are enabled for the Auto Redeposit feature, the system will reject any echeckRedeposit transaction you submit.

### 3.3.17.1 eCheck Redeposit Request

You must specify the eCheck Redeposit request using the following format:

```xml
<echeckRedeposit id="eCheck Redeposit Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <echeck> or <echeckToken>
  <merchantData>
    <customIdentifier>
    </echeckredeposit>
</echeckRedeposit>
```

**NOTE:** If you include the echeck element, the values submitted for accType, accNum, and routingNum children must match those submitted in the original echeckSale transaction.

**Example:** Online eCheck Redeposit Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="81603">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <echeckRedeposit reportGroup="001603">
    <cnpTxnId>345454444</cnpTxnId>
    <echeck>
      <accType>Checking</accType>
      <accNum>1099999903</accNum>
      <routingNum>114567895</routingNum>
    </echeck>
    <merchantData>
      <campaign>New Marketing Campaign</campaign>
    </merchantData>
  </echeckRedeposit>
</litleOnlineRequest>
```
Example: Batch eCheck Redeposit Request

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema"
numBatchRequests="1">
<authentication>
    <user>User Name</user>
    <password>Password</password>
</authentication>
=batchRequest id="uniqueId" numEcheckRedeposit="4" merchantId="1603">
    <echeckRedeposit reportGroup="001603">
        <cnpTxnId>3456456444</cnpTxnId>
        <merchantdata>
            <affiliate>ABC Marketing</affiliate>
        </merchantdata>
    </echeckRedeposit>
    <echeckRedeposit reportGroup="001603">
        <cnpTxnId>3456456449</cnpTxnId>
        <echeck>
            <accType>Checking</accType>
            <accNum>1099999903</accNum>
            <routingNum>114567895</routingNum>
        </echeck>
        <merchantdata>
            <affiliate>ABC Marketing</affiliate>
        </merchantdata>
    </echeckRedeposit>
    <echeckRedeposit reportGroup="001603">
        <cnpTxnId>3456557123</cnpTxnId>
        <echeck>
            <accType>Savings</accType>
            <accNum>10999999444</accNum>
            <routingNum>114567895</routingNum>
        </echeck>
        <merchantdata>
            <affiliate>ABC Marketing</affiliate>
        </merchantdata>
    </echeckRedeposit>
    <echeckRedeposit reportGroup="001603">
        <cnpTxnId>123456789</cnpTxnId>
    </echeckRedeposit>
</batchRequest>
</litleRequest>
```
3.3.17.2 eCheck Redeposit Response

The eCheck Redeposit response indicates that we have received your eCheck Redeposit request. This does not indicate when funds will be transferred. The <accountUpdater> element is included only if the account information submitted in the request transaction has changed (NOC exists). In this case the system automatically updates the information sent to the ACH network and includes the change information in the response.

The eCheck Redeposit response has the following structure:

```
<echeckRedepositResponse id="eCheckRedeposit Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
  <postDate>Date of Posting</postDate> (Online Only)
  <accountUpdater>Account Change Info</accountUpdater>
</echeckRedepositResponse>
```

**Example: Batch eCheck Redeposit Response**

```
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123"
response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <echeckRedepositResponse id="AX54321678" reportGroup="RG27" customerId="53">
      <cnpTxnId>84568456</cnpTxnId>
      <response>000</response>
      <responseTime>2017-06-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckRedepositResponse>

    <echeckRedepositResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <response>000</response>
      <responseTime>2017-06-01T10:24:31</responseTime>
      <message>Approved</message>
      <accountUpdater>
        <originalAccountInfo>
          <accType>Checking</accType>
          <accNum>5186005800001012</accNum>
          <routingNum>000010101</routingNum>
        </originalAccountInfo>
        <newAccountInfo>
          <accType>Checking</accType>
        </newAccountInfo>
      </accountUpdater>
    </echeckRedepositResponse>
  </batchResponse>
</litleResponse>
```
3.3.18 eCheck Sale Transactions

You use the eCheck Sale transaction to capture funds from a customer paying via electronic checks. It is the Direct Debit equivalent of a Capture transaction. Setting the `<verify>` element to `true` triggers an eCheck Verification operation prior to the capture. If the verification fails, the system does not process the capture operation.

**NOTE:** To perform a verification you must include the following optional children of the `billToAddress` element in your request: `firstName`, `lastName`, `companyName` (if accType = Corporate or Corp Savings), `address1` (address 2 and 3 if needed), `city`, `state`, `zip`, and `phone`.

The value for the `orderSource` element must be one of the following: telephone, ecommerce, or recurringtel.

### 3.3.18.1 eCheck Sale Request

You must specify the eCheck Sale request using the following format:

```xml
<echeckSale id="eCheckSale Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <verify>true or false</verify>
  <amount>Authorization Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
  <shipToAddress>
  <echeck> or <echeckToken>
  <customBilling> (Do not use)
  <merchantData>
  <customIdentifier>
</echeckSale>
```
Example: Batch eCheck Sale Request

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="654321" numEcheckSales="1" echeckSalesAmount="10000" merchantId="100">
    <echeckSale id="AX54321678" reportGroup="RG27" customerId="53">
      <orderId>12z58743y1</orderId>
      <verify>true</verify>
      <amount>10000</amount>
      <orderSource>telephone</orderSource>
      <billToAddress>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <addressLine1>123 4th street</addressLine1>
        <addressLine2>Apt. 20</addressLine2>
        <addressLine3>second floor</addressLine3>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95032</zip>
        <country>USA</country>
        <email>jdoe@isp.com</email>
        <phone>408-555-1212</phone>
      </billToAddress>
      <echeck>
        <accType>Checking</accType>
        <accNum>5186005800001012</accNum>
        <routingNum>0001010101</routingNum>
        <checkNum>1104</checkNum>
      </echeck>
    </echeckSale>
  </batchRequest>
</litleRequest>
```

3.3.18.2 eCheck Sale Response

The eCheck Sale response indicates that we have received your eCheck Sale request. This does not indicate when funds will be transferred. The `<accountUpdater>` element is included only if the account information submitted in the request transaction has changed (NOC exists). In this case the system automatically updates the information sent to the ACH network and includes the change information in the response.
The eCheck Sale response has the following structure:

```xml
<echeckSalesResponse id="eCheckSale Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
  <postDate>Date of Posting</postDate> (Online Only)
  <accountUpdater>Account Change Info</accountUpdater>
  <tokenResponse>(for Tokenized merchants submitting account data)</tokenResponse>
</echeckSaleResponse>
```

**Example:** Batch eCheck Sale Response

The response example below includes the accountUpdater element, which indicates that there is a NOC against the account and provides the new account information. If the request used a token, the accountUpdater element would have children providing the original and new token information.

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123"
response="0" message="Valid Format"
cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <echeckSalesResponse id="AX54321678" reportGroup="RG27" customerId="53">
      <cnpTxnId>84568456</cnpTxnId>
      <response>000</response>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckSalesResponse>
    <echeckSalesResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <response>000</response>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
      <accountUpdater>
        <originalAccountInfo>
          <accType>Checking</accType>
          <accNum>5186005800001012</accNum>
          <routingNum>000010101</routingNum>
        </originalAccountInfo>
        <newAccountInfo>
          <accType>Checking</accType>
        </newAccountInfo>
      </accountUpdater>
    </echeckSalesResponse>
  </batchResponse>
</litleResponse>
```

**NOTE:** The schema for `echeckSalesResponse` includes a `verificationCode` child element. This element is not used at this time.
3.3.19 eCheck Verification Transactions

You use an eCheck Verification transaction to initiate a comparison to a database containing information about checking accounts. The database may include information as to whether the account has been closed, as well as whether there is a history of undesirable behavior associated with the account/account holder. This transaction type is only supported for US transactions.

**NOTE:** While eCheck Verification is a valuable tool that you can use to reduce possible fraud and loss, unlike a credit card authorization, it does not check for the availability of funds, nor does it place a hold on any funds.

### 3.3.19.1 eCheck Verification Request

You must specify the eCheck Verification request using the following format:

**IMPORTANT:** For an `eCheckVerification` transaction, you must submit the `firstName` and `lastName` elements instead of the `name` element *(middleInitial is optional)*. For a corporate account you must include the `companyName` element in addition to the `firstName` and `lastName` elements. In both cases, you also must include the `address, city, state, zip` and `phone` information.

For a corporate account, if you do not have the name of the check issuer, you can use a value of “unavailable” for the `firstName` and `lastName` elements.

```xml
<echeckVerification id="echeckVerification Id" reportGroup="iQ Report Group"
customerId="Customer Id"
  <orderId>Order Id</orderId>
  <amount>Authorization Amount</amount>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
    <echeck> or <echeckToken>
    <merchantData>
  </echeckVerification>
```
Example: eCheck Verification Request - Personal Checking

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123"
numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>

  <batchRequest id="654321" numEcheckVerification="1"
echeckVerificationAmount="10000" merchantId="100">
    <echeckVerification id="AX54321678" reportGroup="RG27" customerId="53">
      <orderId>12z58743y1</orderId>
      <amount>10000</amount>
      <orderSource>telephone</orderSource>
      <billToAddress>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
        <addressLine1>123 4th street</addressLine1>
        <addressLine2>Apt. 20</addressLine2>
        <addressLine3>second floor</addressLine3>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95032</zip>
        <country>USA</country>
        <email>jdoe@isp.com</email>
        <phone>408-555-1212</phone>
      </billToAddress>
      <echeck>
        <accType>Checking</accType>
        <accNum>5186005800001012</accNum>
        <routingNum>000010101</routingNum>
        <checkNum>1104</checkNum>
      </echeck>
      <merchantData>
        <campaign>New Campaign</campaign>
      </merchantData>
    </echeckVerification>
  </batchRequest>
</litleRequest>
```

Example: eCheck Verification Request - Corporate Account

NOTE: If you do not have the name of the check issuer, you can use a value of "unavailable" for the `firstName` and `lastName` elements.
3.3.19.2 eCheck Verification Response

The `<accountUpdater>` element is included only if the account information submitted in the request transaction has changed (NOC exists). In this case the system automatically updates the information sent to the ACH network and includes the change information in the response.

The eCheck Verification response has the following structure:

```xml
<echeckVerificationResponse id="echeckVerificationId" reportGroup="iQ Report Group"
```

```xml
<batchRequest id="654321" numEcheckVerification="1" echeckVerificationAmount="10000" merchantId="100">
    <echeckVerification id="AX54321678" reportGroup="RG27" customerId="53">
        <orderId>12z58743y1</orderId>
        <amount>10000</amount>
        <orderSource>telephone</orderSource>
        <billToAddress>
            <firstName>Paul</firstName>
            <lastName>Jones</lastName>
            <companyName>Widget Company</companyName>
            <addressLine1>123 4th street</addressLine1>
            <addressLine2>Apt. 20</addressLine2>
            <addressLine3>second floor</addressLine3>
            <city>San Jose</city>
            <state>CA</state>
            <zip>95032</zip>
            <country>USA</country>
            <email>pjones@isp.com</email>
            <phone>408-555-1212</phone>
        </billToAddress>
        <echeck>
            <accType>Corporate</accType>
            <accNum>5186005800001012</accNum>
            <routingNum>000010101</routingNum>
            <checkNum>1104</checkNum>
        </echeck>
    </echeckVerification>
</batchRequest>
```

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123"
numBatchRequests="1">
    <authentication>
        <user>userName</user>
        <password>password</password>
    </authentication>
    <batchRequest id="654321" numEcheckVerification="1"
    echeckVerificationAmount="10000" merchantId="100">
        <echeckVerification id="AX54321678" reportGroup="RG27" customerId="53">
            <orderId>12z58743y1</orderId>
            <amount>10000</amount>
            <orderSource>telephone</orderSource>
            <billToAddress>
                <firstName>Paul</firstName>
                <lastName>Jones</lastName>
                <companyName>Widget Company</companyName>
                <addressLine1>123 4th street</addressLine1>
                <addressLine2>Apt. 20</addressLine2>
                <addressLine3>second floor</addressLine3>
                <city>San Jose</city>
                <state>CA</state>
                <zip>95032</zip>
                <country>USA</country>
                <email>pjones@isp.com</email>
                <phone>408-555-1212</phone>
            </billToAddress>
            <echeck>
                <accType>Corporate</accType>
                <accNum>5186005800001012</accNum>
                <routingNum>000010101</routingNum>
                <checkNum>1104</checkNum>
            </echeck>
        </echeckVerification>
    </batchRequest>
</litleRequest>
```
Example: eCheck Verification Response

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <echeckVerificationResponse id="AX54321678" reportGroup="RG27" customerId="53">
      <cnpTxnId>84568456</cnpTxnId>
      <response>000</response>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
    </echeckVerificationResponse>
    <echeckVerificationResponse id="AX54325432" reportGroup="RG12">
      <cnpTxnId>84568457</cnpTxnId>
      <response>000</response>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
      <accountUpdater>
        <originalAccountInfo>
          <accType>Checking</accType>
          <accNum>5186005800001012</accNum>
          <routingNum>000010101</routingNum>
        </originalAccountInfo>
        <newAccountInfo>
          <accType>Checking</accType>
          <accNum>5499576040500006</accNum>
          <routingNum>000010102</routingNum>
        </newAccountInfo>
      </accountUpdater>
    </echeckVerificationResponse>
  </batchResponse>
</litleResponse>
```
3.3.20 eCheck Void Transactions (Online Only)

You use an eCheck Void transaction to either halt automatic redeposit attempts of Direct Debits returned for either Insufficient Funds or Uncollected Funds, or cancel an eCheck Sale transaction, as long as the transaction has not yet settled. This also applies to merchant initiated redeposits. You can use this element only in Online transactions.

3.3.20.1 eCheck Void Request

The eCheck Void request references the <litleTxnId> of the previously approved transaction. You must structure an eCheck Void request as follows.

```xml
<echeckVoid id = "echeckVoid Id" reportGroup="iQ Report Group">
  <litleTxnId>Transaction Id</litleTxnId>
</echeckVoid>
```

**Example: eCheck Void Request**

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="81601">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <echeckVoid id="101" reportGroup="001601">
    <cnpTxnId>345454444</cnpTxnId>
  </echeckVoid>
</litleOnlineRequest>
```

3.3.20.2 eCheck Void Response

The eCheck Void response has the following structure.

```xml
<echeckVoidResponse id="eCheck Void Id" reportGroup="iQ Report Group" numDeposits="1">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date of Posting</postDate>
  <message>Response Message</message>
</echeckVoidResponse>
```

**Example: Online eCheck Void Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
  response="0" message="Valid Format">
  <echeckVoidResponse id="101" reportGroup="001601">
  </echeckVoidResponse>
</litleOnlineResponse>
```
3.3.21 Force Capture Transactions

A Force Capture transaction is a Capture transaction used when you do not have a valid Authorization for the order, but have fulfilled the order and wish to transfer funds.

NOTE: Worldpay must authorize you to use this type of transaction. In some instances, using a Force Capture transaction can lead to chargebacks and fines.

3.3.21.1 Force Capture Request

You must specify the Force Capture request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<forceCapture id="Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Force Capture Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <orderSource>Order Entry Source</orderSource>
  <billToAddress>
    [ <card | token> | <paypage> | <mpos> ]
  </billToAddress>
  <customBilling>
    <taxType>payment or fee</taxType>
  </customBilling>
  <enhancedData>
  </enhancedData>
  <processingInstructions>
  </processingInstructions>
  <pos>
    <amexAggregatorData>
    </amexAggregatorData>
    <merchantData>
    </merchantData>
    <debtRepayment>true or false</debtRepayment>
    <processingType>processingType Enum</processingType>
  </pos>
</forceCapture>
```
**Example: Batch Force Capture Request**

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123"
    numBatchRequests="1">
    <authentication>
        <user>userName</user>
        <password>password</password>
    </authentication>
    <batchRequest id="01234567" numForceCaptures="1" forceCaptureAmount="10000"
        merchantId="100">
        <forceCapture id="AX54321678" reportGroup="RG27" customerId="038945">
            <orderId>orderId</orderId>
            <amount>10000</amount>
            <orderSource>ecommerce</orderSource>
            <card>
                <type>VI</type>
                <number>4005550000081019</number>
                <expDate>0910</expDate>
            </card>
            <enhancedData>
                <customerReference>PO12345</customerReference>
                <salesTax>125</salesTax>
                <taxExempt>false</taxExempt>
                <discountAmount>0</discountAmount>
                <shippingAmount>495</shippingAmount>
                <dutyAmount>0</dutyAmount>
                <shipFromPostalCode>01851</shipFromPostalCode>
                <destinationPostalCode>01851</destinationPostalCode>
                <destinationCountryCode>USA</destinationCountryCode>
                <invoiceReferenceNumber>123456</invoiceReferenceNumber>
                <orderDate>2016-09-14</orderDate>
                <detailTax>
                    <taxIncludedInTotal>true</taxIncludedInTotal>
                    <taxAmount>55</taxAmount>
                    <taxRate>0.0059</taxRate>
                    <taxTypeIdentifier>00</taxTypeIdentifier>
                    <cardAcceptorTaxId>011234567</cardAcceptorTaxId>
                </detailTax>
                <lineItemData>
                    <itemSequenceNumber>1</itemSequenceNumber>
                    <itemDescription>chair</itemDescription>
                    <productCode>CH123</productCode>
                    <quantity>1</quantity>
                    <unitOfMeasure>EACH</unitOfMeasure>
                    <taxAmount>125</taxAmount>
                    <lineItemTotal>9380</lineItemTotal>
                </lineItemData>
            </enhancedData>
        </forceCapture>
    </batchRequest>
</litleRequest>
```
Example: Online Force Capture Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema" merchantId="123">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <forceCapture id="AX54321678" reportGroup="RG27" customerId="038945">
    <orderId>orderId</orderId>
    <amount>10000</amount>
    <orderSource>ecommerce</orderSource>
    <card>
      <type>VI</type>
      <number>400555000081019</number>
      <expDate>0907</expDate>
    </card>
    <enhancedData>
      <customerReference>PO12345</customerReference>
      <salesTax>125</salesTax>
      <taxExempt>false</taxExempt>
      <discountAmount>0</discountAmount>
      <shippingAmount>495</shippingAmount>
      <dutyAmount>0</dutyAmount>
      <shipFromPostalCode>01851</shipFromPostalCode>
      <destinationPostalCode>01851</destinationPostalCode>
      <destinationCountryCode>USA</destinationCountryCode>
      <invoiceReferenceNumber>123456</invoiceReferenceNumber>
    </enhancedData>
  </forceCapture>
</litleOnlineRequest>
```
3.3.21.2 Force Capture Response

The Force Capture response message is identical for Online and Batch transactions, except Online includes the <postDate> element. The Force Capture response has the following structure:

```xml
<forceCaptureResponse id="Capture Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date of Posting</postDate> (Online Only)
  <message>Response Message</message>
  <tokenResponse> (for Tokenized merchants submitting card data)
  <accountUpdater>
</forceCaptureResponse>
```
Example: Batch Force Capture Response

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123"
   cnpSessionId="987654321" response="0" message="Valid Format">
  <batchResponse id="01234567" cnpBatchId="445667788" merchantId="100">
    <forceCaptureResponse id="AX54325432" reportGroup="RG12" customerId="038945">
      <cnpTxnId>84568457</cnpTxnId>
      <response>000</response>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
    </forceCaptureResponse>
  </batchResponse>
</litleResponse>
```

Example: Online Force Capture Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
   response="0" message="Valid Format">
  <forceCaptureResponse id="2" reportGroup="ABC Division" customerId="038945">
    <cnpTxnId>1100030204</cnpTxnId>
    <response>000</response>
    <postDate>2016-07-11</postDate>
    <message>Approved</message>
  </forceCaptureResponse>
</litleOnlineResponse>
```

Example: Force Capture Response for Tokenized Merchant Sending Card Data

A tokenized merchant that includes card information in the request receives a response message that
includes the token element. The example below is an Online response.

```xml
<forceCaptureResponse id="99999" reportGroup="RG1" customerId="444">
  <cnpTxnId>2120000039504</cnpTxnId>
  <response>000</response>
  <responseTime>2016-10-20T18:25:38</responseTime>
  <postDate>2016-10-20</postDate>
  <message>Approved</message>
  <tokenResponse>
    <cnpToken>111310880008000</cnpToken>
    <tokenResponseCode>801</tokenResponseCode>
    <tokenMessage>Account number was successfully registered</tokenMessage>
    <type>AX</type>
    <bin></bin>
  </tokenResponse>
</forceCaptureResponse>
```
3.3.22 Fraud Check Transaction

If you wish to retrieve the Advanced Fraud results without introducing a Authorization or Sale transactions, use a Fraud Check transaction, as shown in the example below. Fraud Check transactions are only supported as Online transactions.

3.3.22.1 Fraud Check Request

You must specify the Fraud Check request as follows.

```
<fraudCheck id="Fraud Check Id" reportGroup="iQ Report Group">
  <advancedFraudChecks>
    <billToAddress>
    <shipToAddress>
  </advancedFraudChecks>
</fraudCheck>
```

**Example: Fraud Check Request**

```
<cnponlineRequest version="12.4" xmlns="http://www.vantivcnp.com/schema"
merchantId="81601">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <fraudCheck id="002" reportGroup="001601">
    <advancedFraudChecks>
      <webSessionId>ASDFG-AXXXXAB999</webSessionId>
      <customAttribute1>Attribute passed to Vantiv</customAttribute1>
      <customAttribute2>Attribute passed to Vantiv</customAttribute2>
      <customAttribute3>Attribute passed to Vantiv</customAttribute3>
      <customAttribute4>Attribute passed to Vantiv</customAttribute4>
      <customAttribute5>Attribute passed to Vantiv</customAttribute5>
    </advancedFraudChecks>
    <billToAddress>
      <name>John Doe</name>
      <addressLine1>15 Main Street</addressLine1>
      <city>San Jose</city>
      <state>CA</state>
      <zip>95032-1234</zip>
      <country>USA</country>
      <phone>9782750000</phone>
      <email>jdoe@Worldpay.com</email>
  </billToAddress>
</cnponlineRequest>
```
<billToAddress>
  <shipToAddress>
    <name>Jane Doe</name>
    <addressLine1>15 Main Street</addressLine1>
    <city>San Jose</city>
    <state>CA</state>
    <zip>95032-1234</zip>
    <country>USA</country>
    <phone>9782750000</phone>
    <email>jdoe@vantiv.com</email>
  </shipToAddress>
  <amount>20000</amount>
</billToAddress>

3.3.22.2 Fraud Check Response

A Fraud Check response has the following structure.

```xml
<fraudCheckResponse id="Fraud Check Id" reportGroup="iQ Report Group">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <message>Response Message</message>
  <responseTime>Date and Time in GMT</responseTime>
  <advancedFraudResults>
    <deviceReviewStatus>pass</deviceReviewStatus>
    <deviceReputationScore>50</deviceReputationScore>
    <triggeredRule>FlashImagesCookiesDisabled</triggeredRule>
  </advancedFraudResults>
</fraudCheckResponse>
```

Example: Fraud Check Response

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema"
  response="0" message="Valid Format">
  <fraudCheckResponse id="002" reportGroup="001601">
    <litleTxnId>8282353416454639</litleTxnId>
    <response>000</response>
    <message>Approved</message>
    <responseTime>2016-11-08T21:36:50</responseTime>
    <advancedFraudResults>
      <deviceReviewStatus>pass</deviceReviewStatus>
      <deviceReputationScore>50</deviceReputationScore>
      <triggeredRule>FlashImagesCookiesDisabled</triggeredRule>
    </advancedFraudResults>
  </fraudCheckResponse>
</litleOnlineResponse>
```
3.3.23 Gift Card Auth Reversal Transactions

The Gift Card Auth Reversal transaction allows you to reverse an authorization against a closed-loop gift card; thereby, removing the funds hold established by the authorization.

3.3.23.1 Gift Card Auth Reversal Request

You must specify the Gift Card Auth Reversal request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<giftCardAuthReversal id="Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id from Auth Response</litleTxnId>
  <card>
    <originalRefCode>Auth Code from Auth Response</originalRefCode>
    <originalAmount>Amount from Auth Response</originalAmount>
    <originalTxnTime>txnTime from Auth Response</originalTxnTime>
    <originalSystemTraceId>systemTraceId from Auth Rsp</originalSystemTraceId>
    <originalSequenceNumber>sequenceNumber from Auth Rsp</originalSequenceNumber>
  </card>
</giftCardAuthReversal>
```

**Example:** Online Gift Card Auth Reversal

```xml
<litleOnlineRequest version="11.1" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <giftCardAuthReversal id="834262" reportGroup="ABC Division" customerId="038945">
    <litleTxnId>82823534116454002</litleTxnId>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
      <PIN>1234</PIN>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>40000</originalAmount>
    <originalTxnTime>2016-07-25T15:13:43</originalTxnTime>
    <originalSystemTraceId>654321</originalSystemTraceId>
  </giftCardAuthReversal>
</litleOnlineRequest>
```
<originalSequenceNumber>456789</originalSequenceNumber>
</giftCardAuthReversal>
</litleOnlineRequest>

### 3.3.23.2 Gift Card Auth Reversal Response

A Gift Card Auth Reversal response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<giftCardAuthReversalResponse id="Load Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <giftCardResponse>
    <txnTime>2017-03-25T15:13:38</txnTime>
    <refCode>123456</refCode>
    <systemTraceId>123456</systemTraceId>
    <sequenceNumber>123456</sequenceNumber>
    <availableBalance>5000</availableBalance>
  </giftCardResponse>
</giftCardAuthReversalResponse>
```

**Example:**

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <giftCardAuthReversalResponse id="834262" reportGroup="ABC Division">
    <litleTxnId>9695064321</litleTxnId>
    <response>000</response>
    <postDate>2017-03-25</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-25T15:13:38</txnTime>
      <refCode>123456</refCode>
      <systemTraceId>123456</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>5000</availableBalance>
    </giftCardResponse>
  </giftCardAuthReversalResponse>
</litleOnlineResponse>
```
3.3.24 Gift Card Capture Transactions

The Gift Card Capture transaction allows you to capture (settle) funds previously authorized on a Closed-loop gift card. Unlike the Capture transaction used for a credit card transaction, a Gift Card Capture requires you to provide both the capture amount and the original (authorized) amount. If the transaction is a partial capture, the capture amount will be less than the original amount, with the remainder available for future capture.

3.3.24.1 Gift Card Capture Request

You must specify the Gift Card Capture request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<giftCardCapture id="Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id from Auth Response</litleTxnId>
  <captureAmount>Amt of Capture</captureAmount>
  <card>
    <originalRefCode>Auth Code from Auth Response</originalRefCode>
    <originalAmount>Amount from Auth Request</originalAmount>
    <originalTxnTime>txnTime from Auth Response</originalTxnTime>
  </card>
</giftCardCapture>
```

**Example: Gift Card Capture Request**

```xml
<litleOnlineRequest version="11.1" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <giftCardCapture id="834262" reportGroup="ABC Division" customerId="038945">
    <litleTxnId>82823534116454002</litleTnxId>
    <captureAmount>40000</captureAmount>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>40000</originalAmount>
    <originalTxnTime>2016-07-25T15:13:43</originalTxnTime>
  </giftCardCapture>
</litleOnlineRequest>
```
### 3.3.24.2 Gift Card Capture Response

A Gift Card Capture response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<giftCardCaptureResponse id="Load Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <accountUpdater>
    <fraudResult>
      <giftCardResponse>
      </giftCardResponse>
    </fraudResult>
  </accountUpdater>
</giftCardCaptureResponse>
```

**Example:** Gift Card Capture Response

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <giftCardCaptureResponse id="834262" reportGroup="ABC Division">
    <litleTxnId>9695064321</litleTxnId>
    <response>000</response>
    <postDate>2017-03-25</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-25T15:13:38</txnTime>
      <refCode>123456</refCode>
      <systemTraceId>123456</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>5000</availableBalance>
    </giftCardResponse>
  </giftCardCaptureResponse>
</litleOnlineResponse>
```

### 3.3.25 Gift Card Credit Transactions

The Gift Card Credit transaction allows you to refund funds previously captured on a Closed-loop gift card.
3.3.25.1 Gift Card Credit Request

You must specify the Gift Card Credit request as follows. There are two possible structures for this transaction type. You use the structure containing the litleTxnId element, when the Worldpay processed the capture transaction. You use the structure containing the orderId, when the processor of the capture transaction was not Worldpay. Typically, this occurs when you first migrate your processing to Worldpay. The structures of the requests are identical for either an Online or a Batch submission.

```xml
<giftCardCapture id="Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id from Auth Response</litleTxnId>
  <creditAmount>Amt of Credit</creditAmount>
  <card/>
</giftCardCapture>
```

or

```xml
<giftCardCapture id="Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id from Capture</orderId>
  <creditAmount>Amt of Credit</creditAmount>
  <orderSource/></orderSource>
  <card/>
</giftCardCapture>
```

**Example: Online Gift Card Credit Transaction (Worldpay Processed Capture)**

```xml
<litleOnlineRequest version="11.1" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <giftCardCredit id="834262" reportGroup="ABC Division" customerId="038945">
    <litleTxnId>82823534116454002</litleTnxId>
    <creditAmount>40000</creditAmount>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
    </card>
  </giftCardCredit>
</litleOnlineRequest>
```

**Example: Online Gift Card Credit Transaction (Non-Worldpay Processed Capture)**

```xml
<litleOnlineRequest version="11.1" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
```
<password>Password</password>
</authentication>
<giftCardCredit id="834262" reportGroup="ABC Division" customerId="038945">
  <orderId>116454002</orderId>
  <creditAmount>4000</creditAmount>
  <orderSource>ecommerce</orderSource>
  <card>
    <type>GC</type>
    <number>9000000000000001</number>
  </card>
</giftCardCredit>
</litleOnlineRequest>

### 3.3.25.2 Gift Card Credit Response

A Gift Card Credit response has the following structure. The response message is identical for either request structure, as well as Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<giftCardCaptureResponse id="Load Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <fraudResult>
    <giftCardResponse>
      <txnTime>Transaction Time</txnTime>
      <refCode>Reference Code</refCode>
    </giftCardResponse>
  </fraudResult>
</giftCardCaptureResponse>
```

**Example:** Gift Card Credit Response

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <giftCardCreditResponse id="834262" reportGroup="ABC Division">
    <litleTxnId>9695064321</litleTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T12:00:00</txnTime>
      <refCode>003558</refCode>
    </giftCardResponse>
  </giftCardCreditResponse>
</litleOnlineResponse>
```
3.3.26 Load Transactions

The Load transaction adds funds to an active Gift Card. The load amount cannot exceed the maximum allowed amount for the Gift Card. If you attempt to load more than the maximum amount, the transaction will be declined with a response Code of 221 - Over Max Balance.

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

### 3.3.26.1 Load Request

You must specify the Load request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<load id="Load Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Amount to Load</amount>
  <orderSource>Order Entry Source</orderSource>
</load>
```

**Example:** Online Load Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <load id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>65347567</orderId>
    <amount>40000</amount>
    <orderSource>ecommerce</orderSource>
  </load>
</litleOnlineRequest>
```
3.3.26.2 Load Response

A Load response has the following structure. The response message is identical for Online and Batch transactions except Online includes the <postDate> element.

```xml
<loadResponse id="Load Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate> (Online Only)
  <message>Response Message</message>
  <fraudResult>
    <giftCardResponse>
      <txnTime>Transaction Time</txnTime>
      <refCode>Reference Code</refCode>
      <systemTraceId>System Trace ID</systemTraceId>
      <sequenceNumber>Sequence Number</sequenceNumber>
      <availableBalance>Available Balance</availableBalance>
    </giftCardResponse>
  </fraudResult>
</loadResponse>
```

**Example: Load Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <loadResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T15:13:38</txnTime>
      <refCode>123456</refCode>
      <systemTraceId>123456</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>5000</availableBalance>
    </giftCardResponse>
  </loadResponse>
</litleOnlineResponse>
```
3.3.27 Load Reversal Transactions (Online Only)

The Load Reversal transaction reverses the operation of a Load transaction, removing the newly loaded amount from the Gift Card. The Load Reversal references the associated Load transaction by means of the litleTxnId element returned in the Load response. You cannot perform a partial Load Reversal. This transaction always reverses the full amount of the referenced Load transaction. This transaction type is available only for Online transactions.

NOTE: Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

3.3.27.1 Load Reversal Request

You must structure a Load Reversal request as shown in the following examples.

```xml
<loadReversal id="Load Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id from Load Response</litleTxnId>
  <card>
    <originalRefCode>Reference Code from Load Response</originalRefCode>
    <originalAmount>Amount from Load Transaction</originalAmount>
    <originalTxnTime>Transaction Time from Load Response</originalTxnTime>
    <originalSystemTraceId>Trace Id from Load Response</originalSystemTraceId>
    <originalSequenceNumber>Seq Num from Load Rsp</originalSequenceNumber>
  </card>
</loadReversal>
```

Example: Online Load Reversal Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <loadReversal id="Load Reversal Id" reportGroup="Load Reversals">
    <litleTxnId>1234567890123456789</litleTxnId>
    <cnpTxnId>1234567890123456789</cnpTxnId>
    <card>
      <type>GC</type>
      <number>123410200003558</number>
      <cardValidationNum>888</cardValidationNum>
      <pin>1234</pin>
  </loadReversal>
</litleOnlineRequest>
```
3.3.27.2 Load Reversal Response

An Load Reversal response has the following structure.

```xml
<loadReversalResponse id="Load Reversal Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate>
  <message>Response Message</message>
  <giftCardResponse>
    <txnTime>Date transaction posted</txnTime>
    <refCode>Ref Code</refCode>
    <systemTraceId>System Trace ID</systemTraceId>
    <sequenceNumber>Sequence Number</sequenceNumber>
    <availableBalance>Available Balance</availableBalance>
  </giftCardResponse>
</loadReversalResponse>
```

**Example: Online Load Reversal Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <loadReversalResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T12:00:00</txnTime>
      <refCode>003558</refCode>
      <systemTraceId>834528</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>0</availableBalance>
    </giftCardResponse>
  </loadReversalResponse>
</litleOnlineResponse>
```
3.3.28 Status Query Transactions (Online Only)

You use a Status Query Transaction to verify the status of an Online transaction submitted within the prior 24 hours. As search criteria, you must submit, at a minimum, the id (the id attribute) and transaction type (i.e., authorization, deposit, void, etc.) of the original transaction, but to narrow the search, you can also include the transaction id from the original transaction. This transaction type is available only for Online transactions.

NOTE: You must be specifically enabled to use this transaction type. Please speak to your Implementation Consultant or Relationship Manager for additional information.

3.3.28.1 Query Transaction Request

You must structure your Query request as shown below. The origId and origActionType elements are required.

NOTE: If submitting a query for an Direct Debit transaction, do not include the account number. Use the origAccountNumber element for Credit/Debit/Gift cards only.

```xml
<queryTransaction id="GCQueryAuth" reportGroup = "Mer5PM1" customerId="1">
  <origId>id Attribute for Original Transaction</origId>
  <origActionType>Code for type of Original Transaction</origActionType>
  <origLitleTxnId>litleTxnId from Original Transaction</origLitleTxnId>
</queryTransaction>

Example: Query Transaction Request

```xml
<litleOnlineRequest version="11.1" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <queryTransaction id="query1" reportGroup="Some RG" customerId="038945">
    <origId>834262</origId>
    <origActionType>A</origActionType>
    <origLitleTxnId>9695061110103040608</origLitleTxnId>
  </queryTransaction>
</litleOnlineRequest>
```

3.3.28.2 Query Transaction Response

The structure of the Query Transaction response message can vary according to the results of the query. The results can include a single or multiple transactions that meet the query criteria, no results, if nothing
If no transaction was found, or a limited response, if a transaction was found, but processing was not complete. The structure of the response message will be as follows:

```xml
<queryTransactionResponse>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <message>Response Message</message>
  <matchCount>Number of Matches Found</matchCount>
  <results_max10>
    or
    One or more (10 max) found transaction responses of type specified in the queryTransaction
  </results_max10>
</queryTransactionResponse>
```

**Example: Query Transaction Response with One Found Transaction**

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <queryTransactionResponse id="GCQueryAuth" reportGroup="Mer5PM1" customerId="1">
    <response>150</response>
    <responseTime>2017-04-06T16:40:24</responseTime>
    <message>Original transaction found</message>
    <matchCount>1</matchCount>
    <results_max10>
      <authorizationResponse id="GiftCardAuth" reportGroup="Mer5PM1" customerId="1">
        <litleTxnId>82827170811986124</litleTxnId>
        <orderId>150330_GCAuth</orderId>
        <response>000</response>
        <responseTime>2017-04-06T16:40:04</responseTime>
        <postDate>2017-04-06</postDate>
        <message>Approved</message>
        <authCode>111115</authCode>
        <fraudResult>
          <avsResult>30</avsResult>
          <cardValidationResult>M</cardValidationResult>
        </fraudResult>
        <giftCardResponse>
          <availableBalance>125</availableBalance>
        </giftCardResponse>
      </authorizationResponse>
    </results_max10>
  </queryTransactionResponse>
</litleOnlineResponse>
```
Example: Query Transaction Response with Multiple Found Transactions

NOTE: This response only occurs if you fail to use unique values for the id attribute, when submitting transactions.

```xml
<queryTransactionResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <response>150</response>
  <responseTime>2017-04-06T16:40:24</responseTime>
  <message>Original transaction found</message>
  <matchCount>2</matchCount>
  <results_max10>
    <authorizationResponse id="DupeId" reportGroup="Mer5PM1">
      <litleTxnId>82827170811986215</litleTxnId>
      <orderId>150331_DupeAuth2</orderId>
      <response>000</response>
      <responseTime>2017-04-06T16:40:12</responseTime>
      <postDate>2017-04-06</postDate>
      <message>Approved</message>
      <authCode>055858</authCode>
      <fraudResult>
        <avsResult>32</avsResult>
        <cardValidationResult>M</cardValidationResult>
      </fraudResult>
    </authorizationResponse>
    <authorizationResponse id="DupeId">
      <litleTxnId>82827170811986207</litleTxnId>
      <orderId>150331_DupeAuth1</orderId>
      <response>000</response>
      <responseTime>2017-04-06T16:40:11</responseTime>
      <postDate>2017-04-06</postDate>
      <message>Approved</message>
      <authCode>111111</authCode>
      <fraudResult>
        <avsResult>00</avsResult>
        <cardValidationResult>M</cardValidationResult>
      </fraudResult>
    </authorizationResponse>
  </results_max10>
</queryTransactionResponse>
```
Example: Query Transaction Response with No Found Transaction

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
    <queryTransactionResponse id="AuthFromSale" reportGroup="Mer5PM1">
        <response>151</response>
        <responseTime>2017-04-06T16:40:30</responseTime>
        <message>Original transaction not found</message>
        <matchCount>0</matchCount>
        <results_max10 />
    </queryTransactionResponse>
</litleOnlineResponse>
```

Example: Query Transaction Response with Transaction Found, but not Complete

```xml
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
    <queryTransactionResponse id="someAuth" reportGroup="Mer5PM1">
        <response>152</response>
        <responseTime>2016-04-06T16:40:30</responseTime>
        <message>Original transaction found but response not yet available</message>
        <matchCount>1</matchCount>
        <results_max10>
            <queryTransactionUnavailableResponse>
                <litleTxnId>82827170811986124</litleTxnId>
                <response>152</response>
                <message>Original transaction found but response not yet available</message>
            </queryTransactionUnavailableResponse>
        </results_max10>
    </queryTransactionResponse>
</litleOnlineResponse>
```

### 3.3.29 Refund Reversal Transactions (Online Only)

The Refund Reversal transaction is a Gift Card only transaction that reverses the operation of a Refund transaction on the Gift Card. The Refund Reversal references the associated Credit transaction by means of the `litleTxnId` element returned in the Credit response. You cannot perform a partial Refund Reversal. This transaction always reverses the full amount of the referenced Refund transaction. This transaction type is available only for Online transactions.
### 3.3.29.1 Refund Reversal Request

You must structure a Refund Reversal request as shown in the following examples.

```xml
<refundReversal id="Refund Reversal Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id from CGCredit Response</litleTxnId>
  <card>
    <originalRefCode>Reference Code from CGCredit Response</originalRefCode>
    <originalAmount>Amount from CGCredit Transaction</originalAmount>
    <originalTxnTime>Transaction Time from CGCredit Response</originalTxnTime>
    <originalSystemTraceId>Trace Id from CGCredit Response</originalSystemTraceId>
    <originalSequenceNumber>Seq Num from CGCredit Rsp</originalSequenceNumber>
  </card>
</refundReversal>
```

**Example: Online Refund Reversal Request**

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <refundReversal id="12345" customerId="Customer Id" reportGroup="Refund Reversals">
    <cnpTxnId>1234567890123456789</cnpTxnId>
    <card>
      <type>GC</type>
      <number>1234102000003558</number>
      <cardValidationNum>888</cardValidationNum>
      <pin>1234</pin>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>1900</originalAmount>
    <originalTxnTime>2017-03-21T10:02:46</originalTxnTime>
    <originalSystemTraceId>678901</originalSystemTraceId>
  </refundReversal>
</litleOnlineRequest>
```

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.
3.3.29.2 Refund Reversal Response

An Refund Reversal response has the following structure.

```xml
<refundReversalResponse id="Refund Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date transaction posted</postDate>
  <message>Response Message</message>
  <giftCardResponse>
    <txnTime>2017-03-22T12:00:00</txnTime>
    <refCode>003558</refCode>
    <systemTraceId>834528</systemTraceId>
    <sequenceNumber>123456</sequenceNumber>
    <availableBalance>0</availableBalance>
  </giftCardResponse>
</refundReversalResponse>
```

Example: Online Refund Reversal Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <refundReversalResponse id="834262" reportGroup="ABC Division">
    <cnpTxnId>9695064321</cnpTxnId>
    <response>000</response>
    <responseTime>2017-03-22T15:13:43</responseTime>
    <postDate>2017-03-22</postDate>
    <message>Approved</message>
    <giftCardResponse>
      <txnTime>2017-03-22T12:00:00</txnTime>
      <refCode>003558</refCode>
      <systemTraceId>834528</systemTraceId>
      <sequenceNumber>123456</sequenceNumber>
      <availableBalance>0</availableBalance>
    </giftCardResponse>
  </refundReversalResponse>
</litleOnlineResponse>
```
### 3.3.30 Register Token Transactions

The Register Token transaction enables you to submit a credit card number, Direct Debit account number, or Registration Id to us and receive a token in return. While you can submit Register Token transactions at any time, typically, you would make use of this transactions when initially converting to the use of tokens. In this case you would submit large quantities of credit cards/Direct Debit account numbers in Batches and replace them in your database with the tokens returned.

**NOTE:** When initially tokenizing your customer database, Worldpay recommends that you collect all distinct credit card numbers and submit the information in one or more large Session files. When you receive the response file, parse the returned token information to your database, replacing the card numbers.

### 3.3.30.1 Register Token Request

You must specify the Register Token request as follows. The structure of the request is identical for either an Online or a Batch submission. The child elements used differ depending upon whether you are registering a credit card account, an Direct Debit account, or submitting a Registration Id.

When you submit the CVV2/CVC2/CID in a `registerTokenRequest`, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. This is done to accommodate merchant systems/workflows where the security code is available at the time of token registration, but not at the time of the Auth/Sale. If for some reason you need to change the value of the security code supplied at the time of the token registration, use an `updateCardValidationNumOnToken` transaction. To use the store value when submitting an Auth/Sale transaction, set the `cardValidationNum` value to 000.

**NOTE:** The use of the `cardValidationNum` element in the `registerTokenRequest` only applies when you submit an `accountNumber` element.

For credit cards:

```xml
<registerTokenRequest id="Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <accountNumber>Card Account Number</accountNumber>
  <cardValidationNum>CVV2/CVC2/CID</cardValidationNum>
</registerTokenRequest>
```

For Direct Debit accounts:

```xml
<registerTokenRequest id="Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <echeckForToken>
    <accNum>Account Number</accNum>
    <routingNum>Routing Number</routingNum>
  </echeckForToken>
</registerTokenRequest>
```
For Registration Ids:

```xml
<registerTokenRequest id="Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <paypageRegistrationId>
</registerTokenRequest>
```

For Mobile POS transactions:

```xml
<registerTokenRequest id="Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <mpos>
    <ksn>Key Serial Number</ksn>
    <formatId>Format of Encrypted Track</formatId>
    <encryptedTrack>Encrypted Track Data</encryptedTrack>
    <track1Status>Track Read Status - 0 or 1</track1Status>
    <track2Status>Track Read Status - 0 or 1</track2Status>
  </mpos>
</registerTokenRequest>
```

For Apple Pay transactions:

```xml
<registerTokenRequest id="Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <applepay>
    <data>Encrypted Payment Data</data>
    <header>
      <applicationData>Hash of Application Data Property</applicationData>
      <ephemeralPublicKey>Ephemeral Public Key</ephemeralPublicKey>
      <publicKeyHash>Merchant Cert Encoded Public Key</publicKeyHash>
      <transactionId>Transaction Id from Device</transactionId>
    </header>
    <signature>Signature of Payment and Header Data</signature>
    <version>Payment Token Version</version>
  </applepay>
</registerTokenRequest>
```

**Example:** Batch Register Token Request - Credit Card

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
```

---

編集者：グローバル決済 LLC および/または関連会社。全著作権を保持します。
Example: Batch Register Token Request - Direct Debit

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numTokenRegistrations="1" merchantId="000902">
    <registerTokenRequest id="99999" reportGroup="RG1">
      <orderId>F12345</orderId>
      <echeckForToken>
        <accNum>12345678901234567</accNum>
        <routingNum>000010101</routingNum>
      </echeckForToken>
    </registerTokenRequest>
  </batchRequest>
</litleRequest>
```

Example: Batch Register Token Request - paypageRegistrationId

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numTokenRegistrations="1" merchantId="000902">
    <registerTokenRequest id="99999" reportGroup="RG1">
      <orderId>F12345</orderId>
      <paypageRegistrationId>12345678901234567</paypageRegistrationId>
    </registerTokenRequest>
  </batchRequest>
</litleRequest>
```
3.3.30.2 Register Token Response

There is no structural difference an Online and Batch response; however, some child elements change depending upon whether the token is for a credit card account or an Direct Debit account. The response will have one of the following structures.

Register Token response for Credit Cards:

```xml
<registerTokenResponse id="99999" reportGroup="RGI">
  <litleTxnId>Transaction Id</litleTxnId>
  <litleToken>Token</litleToken>
  <bin>BIN</bin>
  <type>Method of Payment</type>
  <response>Response Code</response>
  <message>Response Message</message>
  <responseTime>Response Time</responseTime>
</registerTokenResponse>
```

Register Token response for Direct Debits:

```xml
<registerTokenResponse id="99999" reportGroup="RGI">
  <litleTxnId>Transaction Id</litleTxnId>
  <litleToken>Token</litleToken>
  <type>Method of Payment</type>
  <eCheckAccountSuffix>Last 3 of Acct Number</eCheckAccountSuffix>
  <response>Response Code</response>
  <responseTime>Response Time</responseTime>
  <message>Response Message</message>
</registerTokenResponse>
```

Register Token response for ApplePay:

```xml
<registerTokenResponse id="99999" reportGroup="RGI">
  <litleTxnId>Transaction Id</litleTxnId>
  <litleToken>Token</litleToken>
  <type>Method of Payment</type>
  <response>Response Code</response>
  <responseTime>Response Time</responseTime>
</registerTokenResponse>
```
Example: Batch Register Token Response - Credit Card

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <registerTokenResponse id="99999" reportGroup="RG1">
      <cnpTxnId>21122700</cnpTxnId>
      <cnpToken>1111000100360002</cnpToken>
      <bin>400510</bin>
      <type>VI</type>
      <response>801</response>
      <responseTime>2016-10-26T17:21:51</responseTime>
      <message>Account number was successfully registered</message>
    </registerTokenResponse>
  </batchResponse>
</litleResponse>
```

Example: Batch Register Token Request - Direct Debit

```xml
<litleResponse version="11.1" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" litleSessionId="987654321">
  <batchResponse id="01234567" litleBatchId="4455667788" merchantId="100">
    <registerTokenResponse id="99999" reportGroup="RG1">
      <litleTxnId>21122700</litleTxnId>
      <litleToken>1111000100360002</litleToken>
      <type>VI</type>
      <eCheckAccountSuffix>511</eCheckAccountSuffix>
      <response>801</response>
      <responseTime>2016-10-26T17:21:51</responseTime>
      <message>Account number was successfully registered</message>
    </registerTokenResponse>
  </batchResponse>
</litleResponse>

3.3.31 RFR Transactions (Batch Only)

An RFR (Request For Response) transaction enables you to request a response file for a previously submitted Batch. You make the request by submitting either the `litleSessionId` of the Batch, or in the case of a request for an Account Updater file, the `accountUpdateFileRequestData` element.

**NOTE:** The use of RFR transactions for Account Updater files apply only to the legacy Account Updater solution.

### 3.3.31.1 RFR Request

You must specify the RFR request as follows.

```
<RFRRequest>
  <litleSessionId | accNum>
</RFRRequest>
```

**Example: RFR Request for Payment Transaction Batch**

The following example shows an RFR request for the response, with 7766554321 as the value of the `<litleSessionId>` element.

```
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="0">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <RFRRequest>
    <cnpSessionId>7766554321</cnpSessionId>
  </RFRRequest>
</litleRequest>
```

**Example: RFR Request for an Account Updater File**

```
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123" numBatchRequests="0">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <RFRRequest>
    <cnpSessionId>7766554321</cnpSessionId>
  </RFRRequest>
</litleRequest>
```
3.3.31.2 RFR Response

When using an RFR request to obtain the response file for a payment transaction Batch, the RFR response is exactly the same as the original session response associated with the <litleSessionId> you submitted in the RFR request. The session ID returned in the response will be the session ID of the original session.

When using an RFR request in an Account Updater scenario, you will receive either an Account Updater Completion response, if the file is ready, or an Account Updater RFR “not ready” response, as shown in the example below.

Example: Account Updater RFR “not ready” Response

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema">
  <RFRResponse response="1" message="The account update file is not ready yet. Please try again later."/>
</RFRResponse>
</litleResponse>
```

3.3.32 Sale Transactions

The Sale transaction enables you to both authorize fund availability and deposit those funds by means of a single transaction. The Sale transaction is also known as a conditional deposit, because the deposit takes place only if the authorization succeeds. If the authorization is declined, the deposit will not be processed.

NOTE: If the authorization succeeds, the deposit is processed automatically, regardless of the AVS, CVV2, CVC2, or CID response, except for American Express transactions. For American Express, a failure to match the security code (CID) results in a declined transaction with the Response Reason Code of 352 - Decline CVV2/CID Fail.

3.3.32.1 Sale Request

You must specify the Sale request as follows. The structure of the request is identical for either an Online or a Batch submission.
NOTE: When you submit the CVV2/CVC2/CID in a registerTokenRequest, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. To use the store value when submitting an Auth/Sale transaction, set the cardValidationNum value to 000.

```
<sale id="Sale Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Authorization Amount</amount>
  <secondaryAmount>Secondary Amount</secondaryAmount>
  <orderSource>Order Entry Source</orderSource>
  <customerInfo>
    <billToAddress>
      <shipToAddress>
        [<card>|<paypage>|<token>|<paypal>|<mpos>|<applepay>]] (Choice)
        <billMeLaterRequest>
        <cardholderAuthentication>
        <customBilling>
        <taxType>payment or fee</taxType>
        <enhancedData>
        <processingInstructions>
        <pos>
        <payPalOrderComplete>Send true in the final Sale</payPalOrderComplete>
        <amexAggregatorData>
        <allowPartialAuth>
        <healthcareIIAS>
        <filtering>
        <merchantData>
        <recyclingRequest> (for Recycling Engine merchants)
        <fraudFilterOverride>
        <recurringRequest> (for Recurring Engine merchants)
        <debtRepayment>true or false</debtRepayment>
        <advancedFraudChecks>
        <wallet>
        <processingType>processingType Enum</processingType>
        <originalNetworkTransactionId>Network Txn Value</originalNetworkTransactionId>
        <originalTransactionAmount>Amount from Orig Txn</originalTransactionAmount>
```
Example: Batch Sale Request

```xml
<litleRequest version="11.4" xmlns="http://www.litle.com/schema" id="123"
numBatchRequests="1">
  <authentication>
    <user>userName</user>
    <password>password</password>
  </authentication>
  <batchRequest id="01234567" numSales="1" saleAmount="12522" merchantId="100">
    <sale id="AX54321678" reportGroup="RG27">
      <orderId>12z58743y1</orderId>
      <amount>12522</amount>
      <orderSource>ecommerce</orderSource>
      <billToAddress>
        <name>John Doe</name>
        <addressLine1>123 4th street</addressLine1>
        <addressLine2>Apt. 20</addressLine2>
        <addressLine3>second floor</addressLine3>
        <city>San Jose</city>
        <state>CA</state>
        <zip>95032</zip>
        <country>USA</country>
        <email>jdoe@isp.com</email>
        <phone>408-555-1212</phone>
      </billToAddress>
      <card>
        <type>MC</type>
        <number>5186005800001012</number>
        <expDate>1110</expDate>
      </card>
    </sale>
  </batchRequest>
</litleRequest>
```

Example: Online Sale Request

```xml
Example: Online Sale Request

NOTE: The example below includes an <orderSource> value of 3dsAuthenticated and includes the <cardholderAuthentication> information. Use this <orderSource> value only if you are a 3DS merchant and authenticated the cardholder.

Also, the values for the <authenticationValue> and <authenticationTransactionId> elements in the example below have been truncated.
```
3.3.32.2 Sale Response

The Sale response message is identical for Online and Batch transactions except Online includes the postDate element. The Sale response has the following structure:

```xml
<saleResponse id="Sale Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <orderId>Order Id</orderId>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date of Posting</postDate> (Online Only)
  <message>Response Message</message>
  <authCode>Approval Code</authCode>
  <approvedAmount>Approved amount for partial Auth</approvedAmount>
  <accountInformation>
    <fraudResult>
    </fraudResult>
    <billMeLaterResponseData>
    </billMeLaterResponseData>
    <tokenResponse> (for Tokenized merchants submitting card data)
    </tokenResponse>
    <enhancedAuthResponse>
    </enhancedAuthResponse>
    <accountUpdater>
    </accountUpdater>
    <recycling> (included for declined Auths if feature is enabled)
    </recycling>
    <recurringResponse> (for Recurring Engine merchants)
    </recurringResponse>
    <giftCardResponse> (included if Gift Card is Method of Payment)
    </giftCardResponse>
    <applepayResponse> (included if an ApplePay transaction)
    </applepayResponse>
    <cardSuffix>Card Last 4</cardSuffix> (included for ApplePay using VI or MC)
    <androidpayResponse>
    </androidpayResponse>
    <networkTransactionId>Txn ID returned from network</networkTransactionId>
  </accountInformation>
</saleResponse>
```

**Example:** Batch Sale Response

```xml
<litleResponse version="11.4" xmlns="http://www.litle.com/schema" id="123" response="0" message="Valid Format" cnpSessionId="987654321">
  <batchResponse id="01234567" cnpBatchId="4455667788" merchantId="100">
    <saleResponse id="AX54321678" reportGroup="RG27">
      <cnpTxnId>84568456</cnpTxnId>
      <response>000</response>
      <orderId>12z58743y1</orderId>
      <responseTime>2016-09-01T10:24:31</responseTime>
      <message>Approved</message>
      <authCode>123456</authCode>
    </saleResponse>
  </batchResponse>
</litleResponse>
```
Example: Online Sale Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <saleResponse id="1" reportGroup="ABC Division" customerId="030945">
    <cnpTxnId>1100030055</cnpTxnId>
    <response>000</response>
    <orderId>23423434</orderId>
    <postDate>2011-07-11</postDate>
    <message>Approved</message>
    <authCode>123457</authCode>
    <fraudResult>
      <avsResult>01</avsResult>
      <cardValidationResult>U</cardValidationResult>
      <authenticationResult>2</authenticationResult>
    </fraudResult>
  </saleResponse>
</litleOnlineResponse>
```

Example: Online Sale Response for Tokenized Merchant Sending Card Data

A tokenized merchant that includes card information in the request receives a response message that includes the token element. The example below is an Online response.

```xml
<saleResponse id="99999" reportGroup="RG1" customerId="444">
  <token>
    <tokenValue>1234567890</tokenValue>
  </token>
  <cnpTxnId>84568457</cnpTxnId>
  <response>000</response>
  <orderId>12z58743y7</orderId>
  <responseTime>2016-09-01T10:24:31</responseTime>
  <message>Approved</message>
  <authCode>123456</authCode>
  <fraudResult>
    <avsResult>00</avsResult>
    <authenticationResult>2</authenticationResult>
  </fraudResult>
</saleResponse>
```
Example: Online Sale Response with Account Updater Info

<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <saleResponse id="1" reportGroup="ABC Division" customerId="038945">
    <cnpTxnId>1100030055</cnpTxnId>
    <response>000</response>
    <orderId>23423434</orderId>
    <postDate>2011-07-11</postDate>
    <message>Approved</message>
    <authCode>123457</authCode>
    <accountUpdater>
      <originalCardInfo>
        <type>VI</type>
        <number>4234823492346234</number>
        <expDate>1112</expDate>
      </originalCardInfo>
      <newCardInfo>
        <type>VI</type>
        <number>4234823490005777</number>
        <expDate>1114</expDate>
      </newCardInfo>
    </accountUpdater>
  </saleResponse>
</litleOnlineResponse>
Example: Batch Sales Response with Recurring Info

The following is an example of the Recurring Response file produced daily to provide information about the transactions submitted by the Recurring Engine. This file is delivered via sFTP.

```xml
<litleResponse version="11.1" xmlns="http://www.litle.com/schema" response="0" message="Valid Format" litleSessionId="82912082263408653">
  <batchResponse litleBatchId="82912082263408661" merchantId="101">
    <saleResponse reportGroup="Default Report Group">
      <litleTxnId>82912082263409610</litleTxnId>
      <response>000</response>
      <orderId>recurring_appr1</orderId>
      <responseTime>2017-01-30T20:15:51</responseTime>
      <message>Approved</message>
      <authCode>11111</authCode>
      <fraudResult>
        <avsResult>01</avsResult>
      </fraudResult>
      <recurringResponse>
        <subscriptionId>82912081866997773</subscriptionId>
        <responseCode>473</responseCode>
        <responseMessage>Scheduled recurring payment processed</responseMessage>
        <recurringTxnId>211014241510</recurringTxnId>
      </recurringResponse>
    </saleResponse>
    <saleResponse reportGroup="Default Report Group">
      <litleTxnId>82912082263409611</litleTxnId>
      <response>000</response>
      <orderId>recurring_appr2</orderId>
      <message>Approved</message>
      <authCode>123457</authCode>
      <fraudResult>
        <avsResult>00</avsResult>
      </fraudResult>
      <recurringResponse>
        <subscriptionId>82912081866997799</subscriptionId>
      </recurringResponse>
    </saleResponse>
  </batchResponse>
</litleResponse>
```
<responseMessage>Scheduled recurring payment processed</responseMessage>
<recurringTxnId>211014245016</recurringTxnId>
</recurringResponse>
</saleResponse>

<saleResponse reportGroup="Default Report Group">
<litleTxnId>82912082263410337</litleTxnId>
<response>110</response>
<orderId>recurring_decline1</orderId>
<responseTime>2017-01-30T20:15:56</responseTime>
<message>Insufficient Funds</message>
<fraudResult>
<avsResult>34</avsResult>
</fraudResult>
<recycling>
<recycleEngineActive>true</recycleEngineActive>
</recycling>
<recurringResponse>
<subscriptionId>82912081866997807</subscriptionId>
<responseCode>473</responseCode>
<responseMessage>Scheduled recurring payment processed</responseMessage>
<recurringTxnId>211014245115</recurringTxnId>
</recurringResponse>
</saleResponse>

<saleResponse reportGroup="Default Report Group">
<litleTxnId>82912082263410378</litleTxnId>
<response>110</response>
<orderId>recurring_decline3</orderId>
<responseTime>2017-01-30T20:15:56</responseTime>
<message>Insufficient Funds</message>
<fraudResult>
<avsResult>34</avsResult>
</fraudResult>
<recycling>
<recycleEngineActive>true</recycleEngineActive>
</recycling>
<recurringResponse>
<subscriptionId>82912081866997807</subscriptionId>
<responseCode>473</responseCode>
<responseMessage>Scheduled recurring payment processed</responseMessage>
<recurringTxnId>211014245313</recurringTxnId>
</recurringResponse>
</saleResponse>
</batchResponse>
</litleResponse>
### 3.3.33 Unload Transactions

The Unload transaction removes funds from an active Gift Card. The unload amount cannot exceed the available balance on the Gift Card. If you attempt to unload more than the available balance, the transaction will be declined with a response Code of 209 - Invalid Amount.

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.

#### 3.3.33.1 Unload Request

You must specify the Unload request as follows. The structure of the request is identical for either an Online or a Batch submission.

```xml
<unload id="Unload Id" reportGroup="iQ Report Group" customerId="Customer Id">
  <orderId>Order Id</orderId>
  <amount>Amount to Load</amount>
  <orderSource>Order Entry Source</orderSource>
  <card>
    <type>GC</type>
    <number>9000000000000001</number>
    <pin>1234</pin>
  </card>
</unload>
```

**Example:** Online Unload Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema" merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <unload id="834262" reportGroup="ABC Division" customerId="038945">
    <orderId>65347567</orderId>
    <amount>40000</amount>
    <orderSource>ecommerce</orderSource>
    <card>
      <type>GC</type>
      <number>9000000000000001</number>
      <pin>1234</pin>
    </card>
  </unload>
</litleOnlineRequest>
```
### 3.3.3.2 Unload Response

An Unload response has the following structure. The response message is identical for Online and Batch transactions except Online includes the `<postDate>` element.

```xml
<unloadResponse id="Unload Id" reportGroup="iQ Report Group"
customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <postDate>Date transaction posted (Online Only)</postDate>
    <message>Response Message</message>
    <fraudResult>
        <giftCardResponse>
            <txnTime>Date transaction posted</txnTime>
            <refCode>Reference Code</refCode>
            <systemTraceId>System Trace ID</systemTraceId>
            <sequenceNumber>Sequence Number</sequenceNumber>
            <availableBalance>Available Balance</availableBalance>
        </giftCardResponse>
    </fraudResult>
</unloadResponse>
```

**Example: Unload Response**

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
    response="0" message="Valid Format">
    <unloadResponse id="834262" reportGroup="ABC Division">
        <cnpTxnId>9695064321</cnpTxnId>
        <response>000</response>
        <responseTime>2017-03-22T15:13:43</responseTime>
        <postDate>2017-03-22</postDate>
        <message>Approved</message>
        <giftCardResponse>
            <txnTime>2017-03-22T12:00:00</txnTime>
            <refCode>003558</refCode>
            <systemTraceId>834528</systemTraceId>
            <sequenceNumber>123456</sequenceNumber>
            <availableBalance>0</availableBalance>
        </giftCardResponse>
    </unloadResponse>
</litleOnlineResponse>
```

### 3.3.34 Unload Reversal Transactions (Online Only)

The Unload Reversal transaction reverses the operation of a Unload transaction, returning the value removed from the Gift Card by the Unload transaction. The Unload Reversal references the associated Unload transaction by means of the `<litleTxnId>` element returned in the Unload response. You cannot perform a partial Unload Reversal. This transaction always reverses the full amount of the referenced Unload transaction. This transaction type is available only for Online transactions.
### 3.3.34.1 Unload Reversal Request

You must structure a Unload Reversal request as shown in the following examples.

```xml
<unloadReversal id="Unload Reversal Id" reportGroup="iQ Report Group"
customerId="Customer Id">
  <litleTxnId>Transaction Id from CGCredit Response</litleTxnId>
  <card>
    <originalRefCode>Reference Code from Unload Response</originalRefCode>
    <originalAmount>Amount from Unload Transaction</originalAmount>
    <originalTxnTime>Transaction Time from Unload Response</originalTxnTime>
    <originalSystemTraceId>Trace Id from Unload Response</originalSystemTraceId>
    <originalSequenceNumber>Seq Num from Unload Rsp</originalSequenceNumber>
  </card>
</unloadReversal>
```

**Example: Online Unload Reversal Request**

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <unloadReversal id="12345" customerId="Customer Id" reportGroup="Unload Reversals">
    <cnpTxnId>12345678901234567890</cnpTxnId>
    <card>
      <type>GC</type>
      <number>1234102000003558</number>
      <cardValidationNum>888</cardValidationNum>
      <pin>1234</pin>
    </card>
    <originalRefCode>123456</originalRefCode>
    <originalAmount>1900</originalAmount>
    <originalTxnTime>2017-03-21T10:02:46</originalTxnTime>
    <originalSystemTraceId>678901</originalSystemTraceId>
    <originalSequenceNumber>123456</originalSequenceNumber>
  </unloadReversal>
</litleOnlineRequest>
```

**NOTE:** Worldpay must enable you for (Closed Loop) Gift Card transactions to use this transaction type. Please consult your Relationship Manager for additional information about Gift Card transactions.
### 3.3.34.2 Unload Reversal Response

An Unload Reversal response has the following structure.

```xml
<unloadReversalResponse id="Unload Reversal Id" reportGroup="iQ Report Group" customerId="Customer Id">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <responseTime>Date and Time in GMT</responseTime>
    <postDate>Date transaction posted</postDate>
    <message>Response Message</message>
    <giftCardResponse>
        <txnTime>Date transaction posted</txnTime>
        <refCode>Reference Code</refCode>
        <systemTraceId>System Trace ID</systemTraceId>
        <sequenceNumber>Sequence Number</sequenceNumber>
        <availableBalance>Available Balance</availableBalance>
    </giftCardResponse>
</unloadReversalResponse>
```

**Example:** Online Unload Reversal Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
    <unloadReversalResponse id="834262" reportGroup="ABC Division">
        <cnpTxnId>9695064321</cnpTxnId>
        <response>000</response>
        <responseTime>2017-03-22T15:13:43</responseTime>
        <postDate>2017-03-22</postDate>
        <message>Approved</message>
        <giftCardResponse>
            <txnTime>2017-03-22T12:00:00</txnTime>
            <refCode>003558</refCode>
            <systemTraceId>834528</systemTraceId>
            <sequenceNumber>123456</sequenceNumber>
            <availableBalance>0</availableBalance>
        </giftCardResponse>
    </unloadReversalResponse>
</litleOnlineResponse>
```

### 3.3.35 Update Plan Transactions

You use the Update Plan transaction to activate/deactivate Plans associated with recurring payments. When you deactivate a Plan, by setting the `active` flag to `false`, you can no longer reference that Plan for use with subscriptions. Existing subscriptions already using the deactivated Plan will continue to use
the Plan until the subscription is either modified or completed. You can also reactivate a deactivated Plan by updating the Plan and setting the active flag to true.

### 3.3.35.1 Update Plan Request

You must structure an Update Plan request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```xml
<updatePlan>
  <planCode>Plan Reference Code</planCode>
  <active>true or false</active>
</updatePlan>
```

**Example:** Online Update Plan Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>password</password>
  </authentication>
  <updatePlan>
    <planCode>Reference_Code</planCode>
    <active>false</active>
  </updatePlan>
</litleOnlineRequest>
```

### 3.3.35.2 Update Plan Response

An Update Plan response has the following structure. The response message is identical for Online and Batch transactions.

```xml
<updatePlanResponse>
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <message>Response Message</message>
  <responseTime>Date and Time in GMT</responseTime>
  <planCode>Plan Reference Code</subscriptionId>
</updatePlanResponse>
```

**Example:** Online Update Plan Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
  response="0" message="Valid Format">
  <updatePlanResponse>
```

```xml
</litleOnlineResponse>
```
3.3.36 Update Subscription Transactions

The Update Subscription transaction allows you to change certain subscription information associated with a recurring payment. Using this transaction type you can change the plan, card, billing information, and/or billing date. You can also create, update, or delete a Discount and/or an Add On.

**NOTE:** You can include multiple create, update, and delete Discounts and Add On operations in a single `updateSubscription` transaction.

### 3.3.36.1 Update Subscription Request

You must structure an Update Subscription request as shown in the following examples. The structure of the request is identical for either an Online or a Batch submission.

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <planCode>Plan Reference Code</subscriptionId>
  <billToAddress>
    [ <card> | <paypage> | <token> ]
  <billingDate>New Billing Date</billingDate>
  <createDiscount>
  <updateDiscount>
  <deleteDiscount>
  <createAddOn>
  <updateAddOn>
  <deleteAddOn>
</updateSubscription>
```

**Example: Online Update Subscription Request**

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">  <authentication>  <user>User Name</user>
```
3.3.36.2 Update Subscription Response

An Update Subscription response has the following structure. The response message is identical for Online and Batch transactions.

```xml
<updateSubscriptionResponse>
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
</updateSubscriptionResponse>
```
Example: Online Update Subscription Response

```xml
<littleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema" response="0" message="Valid Format">
  <updateSubscriptionResponse>
    <cnpTxnId>1100030055</cnpTxnId>
    <response>000</response>
    <message>Approved</message>
    <subscriptionId>123457</subscriptionId>
  </updateSubscriptionResponse>
</littleOnlineResponse>
```

### 3.3.37 Update Card Validation Number Transactions

When you submit the CVV2/CVC2/CID in a registerTokenRequest, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. This is done to accommodate merchant systems/workflows where the security code is available at the time of token registration, but not at the time of the Auth/Sale. If for some reason you need to change the value of the security code supplied at the time of the token registration, use an updateCardValidationNumOnToken transaction.

**NOTE:** You should only use this transaction type if you had previously submitted the account number and security code in a registerTokenRequest transaction and now need to change the CVV2/CVC2/CID value.

#### 3.3.37.1 Update Card Validation Number Request

The updateCardValidationNumOnToken transaction has the following structure:

```xml
<updateCardValidationNumOnToken id = "Update Id" customerId="Customer Id" reportGroup="iQ Report Group">
  <orderId>Order Id</orderId>
  <litleToken>Token</litleToken>
  <cardValidationNum>CVV2/CVC2/CID Value</cardValidationNum>
</updateCardValidationNumOnToken>
```
**Example:** Online Update Card Validation Number Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
    merchantId="100">
    <updateCardValidationNumOnToken id="99999" customerId="444" reportGroup="RG1">
        <orderId>F12345</orderId>
        <cnpToken>1111000101039449</cnpToken>
        <cardValidationNum>987</cardValidationNum>
    </updateCardValidationNumOnToken>
</litleOnlineRequest>
```

### 3.3.37.2 Update Card Validation Number Response

The `updateCardValidationNumOnTokenResponse` transaction has the following structure:

```xml
<updateCardValidationNumOnTokenResponse id="Update Id" reportGroup="iQ Report Group">
    <litleTxnId>Transaction Id</litleTxnId>
    <response>Response Code</response>
    <message>Response Message</message>
    <responseTime>Date and Time in GMT</responseTime>
</updateCardValidationNumOnTokenResponse>
```

**Example:** Online Update Card Validation Number Response

```xml
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
    response="0" message="Valid Format">
    <updateCardValidationNumOnTokenResponse id="99999" customerId="444"
        reportGroup="RG1">
        <cnpTxnId>21122700</cnpTxnId>
        <response>803</response>
        <message>Card Validation Number Updated</message>
        <responseTime>2017-06-09T17:21:51</responseTime>
    </updateCardValidationNumOnTokenResponse>
</litleOnlineResponse>
```

### 3.3.38 Void Transactions (Online Only)

You use a Void transaction to cancel a settlement transaction prior to us delivering the transaction to the card brands (see **Multiple Daily Delivery** on page 5). You can void Capture, Credit, and Sale transactions. Also, if you use Recycling Engine, you can use the `void` transaction to halt the recycling of a `sale` transaction. In this case the response may include the `recycling` element. (see **Using Void to Halt Recycling Engine** on page 77).
If you attempt to void a transaction after the cutoff time, the system returns a response code of 362 and the message, **Transaction Not Voided - Already Settled**. In this situation, you can cancel the original transaction by using its reverse transaction, as shown in Table 3-2.

### Table 3-2  Cancelling a Transaction That Cannot Be Voided

<table>
<thead>
<tr>
<th>If you had originally sent this transaction...</th>
<th>Cancel it by using this transaction...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capture Transactions</td>
<td>Credit Transactions</td>
</tr>
<tr>
<td>Sale Transactions</td>
<td>Credit Transactions</td>
</tr>
<tr>
<td>Credit Transactions</td>
<td>Sale Transactions</td>
</tr>
<tr>
<td>Force Capture Transactions</td>
<td>Credit Transactions</td>
</tr>
</tbody>
</table>

#### 3.3.38.1 Void Request

The Void request references the `<litleTxnId>` of the previously approved transaction. You must structure a Void request as follows.

```xml
<void id = "Void Id" reportGroup="iQ Report Group">
  <litleTxnId>Transaction Id</litleTxnId>
  <processingInstructions/>
</void>
```

**Example:** Online Void Request

```xml
<litleOnlineRequest version="11.4" xmlns="http://www.litle.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <void id="1" reportGroup="Void Division">
    <cnpTxnId>345454444</cnpTxnId>
  </void>
</litleOnlineRequest>
```

**NOTE:** Before submitting a Void, please allow a minimum of 60 seconds to elapse after submitting the transaction you wish to void. This timing ensures our system fully records the first (to be voided) transaction in our database.

Do not use Void transactions to void an Authorization. To remove an Authorization use an Authorization Reversal transaction (see Authorization Reversal Transactions on page 204.)
3.3.38.2 Void Response

The Void response has the following structure.

```
<voidResponse id="Void Id" reportGroup="iQ Report Group">
  <litleTxnId>Transaction Id</litleTxnId>
  <response>Response Code</response>
  <responseTime>Date and Time in GMT</responseTime>
  <postDate>Date of Posting</postDate>
  <message>Response Message</message>
  <recycling> (May be included if halting recycling.)
</voidResponse>
```

**Example:** Online Void Response

```
<litleOnlineResponse version="11.4" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <voidResponse id="1" reportGroup="Void Division">
    <cnpTxnId>1100026202</cnpTxnId>
    <response>000</response>
    <responseTime>2017-06-16T19:43:38</responseTime>
    <postDate>2017-06-16</postDate>
    <message>Approved</message>
  </voidResponse>
</litleOnlineResponse>
```

**Example:** Online Void Response with Recycling Element

When you use a Void transaction to halt recycling, the response may include the `recycling` element. (see Using Void to Halt Recycling Engine on page 77).

```
<litleOnlineResponse version="11.1" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <voidResponse id="1" reportGroup="Void Division">
    <litleTxnId>1100026202333456789</litleTxnId>
    <response>000</response>
    <responseTime>2017-06-16T19:43:38</responseTime>
    <postDate>2017-06-16</postDate>
    <message>Approved</message>
    <recycling>
      <creditLitleTnxId>1234567890123456789</creditLitleTnxId>
    </recycling>
  </voidResponse>
</litleOnlineResponse>
```
cnpAPI Elements

This chapter provides definitions for the elements and attributes used by the cnpAPI. This information is intended to be used in combination with the various cnpAPI schema files to assist you as you build the code necessary to submit transactions to our transaction processing systems. Each section defines a particular element, its relationship to other elements (parents and children), as well as any attributes associated with the element.

For additional information on the structure of cnpAPI requests and responses using these elements, as well as XML examples, please refer to Chapter 3, "cnpAPI Transaction Examples".

The XML elements defined in this chapter are listed alphabetically.
4.1 accNum

The accNum element is a required child of the echeck, originalAccountInfo, and newAccountInfo elements defining the account number of the Direct Debit account. 

Type = String; minLength = N/A; maxLength = 17

Parent Elements:

echeck, newAccountInfo, originalAccountInfo, accountInfo

Attributes:

None

Child Elements:

None
4.2 accountInfo

The accountInfo element is a required child of the submerchantCredit, and submerchantDebit transactions. It contains child elements used to provide details concerning the Sub-merchant account.

Parent Elements:
submerchantCredit, submerchantDebit, vendorCredit, vendorDebit

Attributes:
None

Child Elements:
Required: accType, accNum, routingNum
Optional: ccdPaymentInformation

NOTE: Although shown as an optional element in the schema, the checkNum element should not be used as a child of accountInfo.

Example: accountInfo Structure

```xml
<accountInfo>
  <accType>Account Type Abbreviation</accType>
  <accNum>Account Number</accNum>
  <routingNum>Routing Number</routingNum>
  <ccdPaymentInformation>Payment Description</ccdPaymentInformation>
</accountInfo>
```
4.3 accountInformation

The accountInformation element is an optional child of the authorizationResponse and saleResponse elements. It contains two children that define the card type and account number.

Parent Elements:
authorizationResponse, saleResponse

Attributes:
None

Child Elements:
Required: type
Optional: number
4.4 accountNumber

The accountNumber element is a required child of the registerTokenRequest element defining the account number for which you are requesting a token. It is also an optional child of the virtualGiftCardResponse element, where it defines the account number of the requested Virtual Gift Card.

Type = String; minLength = 13; maxLength = 25

Parent Elements:
registerTokenRequest, virtualGiftCardResponse

Attributes:
None

Child Elements:
None
4.5  accountNumberLength

The accountNumberLength element is a required child of the virtualGiftCard element defining the requested length of the virtual Gift Card number you are requesting.

**Type** = Integer; **Allowed Values** between 13 and 25 inclusive.

**IMPORTANT:** Although the schema defines the allowed values as any integer between 13 and 25 inclusive, it this time you can only use a value of either 16 or 19.

**Parent Elements:**
virtualGiftCard

**Attributes:**
None

**Child Elements:**
None
4.6 accountUpdate

The accountUpdate element is the parent element for all Account Updater request transactions. You can use this only with Batch transactions.

Parent Elements:
batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)

orderid, cardOrToken (allows the substitution of either the card or token elements)
4.7 accountUpdateFileRequestData

The `accountUpdateFileRequestData` element is a child of the `RFRRequest` element, required when requesting the response file for an (legacy) Account Updater submission.

**Parent Elements:**
- `RFRRequest`

**Attributes:**
- None

**Child Elements:**
- **Required:** merchantId
- **Optional:** postDay

**Example:** accountUpdateFileRequestData Structure

```
<accountUpdateFileRequestData>
  <merchantId>Merchant ID</merchantId>
  <postDay>Post Date</postDay>
</accountUpdateFileRequestData>
```
4.8 accountUpdater

The accountUpdater element is an optional child of the authorizationResponse, captureResponse, echeckSalesResponse, echeckCreditResponse, echeckVerificationResponse, forceCaptureResponse, and saleResponse elements. This element is included in the response messages when the submitted account information has changed.

In the case of Direct Debit accounts, the system automatically updates the information sent to the ACH network and includes the original and updated information in the response. Similarly, if you use the Account Updater service (for credit cards), the system automatically repairs the card information sent to the card networks and depending upon the option you select, can return the info to you.

Parent Elements:
- authorizationResponse
- captureResponse
- forceCaptureResponse
- echeckCreditResponse
- echeckRedepositResponse
- echeckSalesResponse
- saleResponse

Attributes:
None

Child Elements:
Required: extendedCardResponse, newAccountInfo, newCardInfo, newCardTokenInfo, newTokenInfo, originalAccountInfo, originalCardInfo, originalCardTokenInfo, originalTokenInfo

Example: accountUpdater Structure - Credit Cards without extendedCardResponse

```xml
<accountUpdater>
  <originalCardInfo>
    <type>Card Type</type>
    <number>Old Account Number</number>
    <expDate>Old Expiration Date</expDate>
  </originalCardInfo>
  <newCardInfo>
    <type>Card Type</type>
    <number>New Account Number</number>
    <expDate>New Expiration Date</expDate>
  </newCardInfo>
</accountUpdater>
```

IMPORTANT: When using Account Updater (any variation), you must always code to receive the extendedCardResponse element and its children. Worldpay always returns this information whenever applicable regardless of whether you receive other account updater information in the transaction response message.
Example: accountUpdater Structure - Credit Cards with extendedCardResponse

```xml
<accountUpdater>
  <originalCardInfo>
    <type>Card Type</type>
    <number>Old Account Number</number>
    <expDate>Old Expiration Date</expDate>
  </originalCardInfo>
  <newCardInfo>
    <type>Card Type</type>
    <number>New Account Number</number>
    <expDate>New Expiration Date</expDate>
  </newCardInfo>
  <extendedCardResponse>
    <message>Code Description</message>
    <code>Either 501 or 504</code>
  </extendedCardResponse>
</accountUpdater>
```

Example: accountUpdater Structure - Credit Cards only extendedCardResponse

```xml
<accountUpdater>
  <extendedCardResponse>
    <message>Code Description</message>
    <code>Either 501 or 504</code>
  </extendedCardResponse>
</accountUpdater>
```

Example: accountUpdater Structure - Credit Cards (tokenized Merchant)

```
NOTE: This structure can also include the <extendedCardResponse> element.

<accountUpdater>
  <originalCardTokenInfo>
    <litleToken>Old Token</litleToken>
    <type>Card Type</type>
    <expDate>Old Expiration Date</expDate>
    <bin>Old Card BIN</bin>
  </originalCardTokenInfo>
</accountUpdater>
```
<newCardTokenInfo>
  <litleToken>New Token</litleToken>
  <type>Card Type</type>
  <expDate>New Expiration Date</expDate>
  <bin>New Card BIN</bin>
</newCardTokenInfo>

Example: accountUpdater Structure - Direct Debit (for non-Tokenized Merchant)

<accountUpdater>
  <originalAccountInfo>
    <accType>Original Account Type</accType>
    <accNum>Original Account Number</accNum>
    <routingNum>Original Routing Number</routingNum>
  </originalAccountInfo>
  <newAccountInfo>
    <accType>New Account Type</accType>
    <accNum>New Account Number</accNum>
    <routingNum>New Routing Number</routingNum>
  </newAccountInfo>
</accountUpdaterFileRequestData>

Example: accountUpdater Structure - Direct Debit (for Tokenized Merchant)

<accountUpdater>
  <originalTokenInfo>
    <accType>Original Account Type</accType>
    <litleToken>Original Token</litleToken>
    <routingNum>Original Routing Number</routingNum>
  </originalTokenInfo>
  <newTokenInfo>
    <accType>New Account Type</accType>
    <litleToken>New Token</litleToken>
    <routingNum>New Routing Number</routingNum>
  </newTokenInfo>
</accountUpdaterFileRequestData>
4.9  accountUpdateResponse

The accountUpdaterResponse element is the parent element for all Account Update responses transactions. You can use this only with Batch transactions.

Parent Elements:

batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the accountUpdate transaction.</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the accountUpdate transaction.</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the accountUpdate transaction.</td>
</tr>
</tbody>
</table>

Child Elements: (Required)

litleTxnId, orderId, response, responseTime, message

Child Elements: (Optional)

updatedCard, originalCard, originalToken, updatedToken
4.10 accType

The accType element is a required child of the echeck, originalAccountInfo, and newAccountInfo elements defining the type of eCheck account used in the transaction.

**Type** = Choice (Enum); **Enumerations** = Checking, Savings, Corporate, or Corp Savings

**NOTE:** Use Corporate for Corporate Checking accounts.

**Parent Elements:**
echeck, newAccountInfo, originalAccountInfo, originalTokenInfo, newTokenInfo, accountInfo

**Attributes:**
None

**Child Elements:**
None
4.11 actionReason

The actionReason element is an optional child of the authReversal element defining if the reversal is due to suspected fraud.

Type = String (Enum); Enumerations = SUSPECT_FRAUD

NOTE: When you include this optional element in an authReversal transaction, the information will be forwarded to MasterCard as part of the MasterCard eCommerce Fraud Alert program.

When you include this optional element in an credit transaction, the system uses the information to track potentially fraudulent transactions for future analysis.

Parent Elements:
authReversal, credit

Attributes:
None

Child Elements:
None
4.12 activate

The activate element is the parent element for the transaction type that activates a Gift Card.

Parent Elements:
littleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (Required)
orderId, amount, orderSource, card, virtualGiftCard
4.13 activateResponse

The activateResponse element is the parent element for information returned to you in response to an activate transaction. You can use this element in either Online or Batch transactions.

Parent Elements:
littleOnlineResponse, batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Activate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Activate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Activate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: littleTxnId, response, responseTime, message, giftCardResponse

Optional: postDate, fraudResult, giftCardResponse, virtualGiftCardResponse
4.14 activateReversal

The `activateReversal` element is the parent element for the transaction type that reverses the activation of a Gift Card.

Parent Elements:
- litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (Required)
- card, originalRefCode, originalAmount, originalTxnTime, originalSystemTraceId, originalSequenceNumber

Child Elements: (Optional)
- litleTxnId, virtualGiftCardBin
4.15 activateReversalResponse

The `activateReversalResponse` element is the parent element for information returned to you in response to an `activateReversal` transaction. You can use this element in either Online or Batch transactions.

Parent Elements:
- `litleOnlineResponse`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Activate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>    <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Activate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>   <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Activate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>    <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

Child Elements:

Required: `litleTxnId`, `response`, `responseTime`, `message`, `giftCardResponse`

Optional: `postDate`
4.16 **active**

The `active` element is an optional child of both the `createPlan` and `updatePlan` elements. You use this flag to activate/deactivate a Plan. When you deactivate a Plan, you can no longer reference that Plan for use with subscriptions. Existing subscriptions making use of the deactivated Plan will continue to use the Plan until either modified or completed. You can also reactivate a deactivated Plan by updating the Plan and setting the `active` flag to true.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
`createPlan, updatePlan`

**Attributes:**
None

**Child Elements:**
None
4.17 addOnCode

The `addOnCode` element is the identifier of a defined add on charge to a subscription. You use this element when creating, updating, and deleting an Add On applied to a subscription.

**Type** = String; **minLength** = N/A; **maxLength** = 25

**Parent Elements:**
createAddOn, updateAddOn, deleteAddOn

**Attributes:**
None

**Child Elements:**
None
4.18 addressIndicator

The `addressIndicator` element is an optional child of the `billMeLaterResponseData` element and indicates whether the shipping address is a commercial (`c`) or residential (`r`) shipping address.

Type = String; minLength = N/A; maxLength = 1

Parent Elements:
`billMeLaterResponseData`

Attributes:
None

Child Elements:
None
4.19  addressLine1, addressLine2, addressLine3

The elements addressLine1, addressLine2, and addressLine3 define the address information in both the billToAddress and shipToAddress elements.

Type = String; minLength = N/A; maxLength = 35

Parent Elements:
- billToAddress
- shipToAddress

Attributes:
- None

Child Elements:
- None
4.20 advancedAVSResult

The advancedAVSResult element is an optional child element of the fraudResult element. It defines the American Express Advanced AVS response codes that can be returned as verification of information supplied in the <phone> and/or <email> child elements of the <billToAddress> element. For a list of possible values, please refer to AAVS Response Codes on page 809.

NOTE: You must be specifically enabled to use the Advanced AVS feature. Please consult your Relationship Manager for additional information.

Type = String; minLength = N/A; maxLength = 3

Parent Elements:
fraudResult

Attributes:
None

Child Elements:
None
4.21  advancedFraudChecks

The advancedFraudChecks element is an optional child of both the authorization and sale elements and a required child of the fraudCheck element.

Parent Elements:
authorization, sale, fraudCheck

Attributes:
None

Child Elements:
threatMetrixSessionId, customAttribute1 through customAttribute5

Example: advancedFraudChecks Structure

```xml
<advancedFraudChecks>
  <threatMetrixSessionId>Session Id from Threat Metrix</threatMetrixSessionId>
  <customAttribute1>Some attribute passed to ThreatMetrix</customAttribute1>
  <customAttribute2>Some attribute passed to ThreatMetrix</customAttribute2>
  <customAttribute3>Some attribute passed to ThreatMetrix</customAttribute3>
  <customAttribute4>Some attribute passed to ThreatMetrix</customAttribute4>
  <customAttribute5>Some attribute passed to ThreatMetrix</customAttribute5>
</advancedFraudChecks>
```
4.22 advancedFraudResults

The advancedFraudResults element is an optional child of both the fraudResult and the fraudCheckResponse elements. Child elements return the results of advanced fraud checks performed by Threat Metrix, as well as a list of the rules triggered from the ThreatMetrix (merchant) Policy.

Parent Elements:
fraudResult, fraudCheckResponse

Attributes:
None

Child Elements: (Optional)

NOTE: The elements, deviceReputationScore and triggeredRule, always appear, unless deviceReviewStatus is unavailable.

deviceReviewStatus, deviceReputationScore, triggeredRule

Example: advancedFraudChecks Structure

<advancedFraudResults>
  <deviceReviewStatus>pass, fail, review, or unavailable</deviceReviewStatus>
  <deviceReputationScore>Score Returned from ThreatMetix</deviceReputationScore>
  <triggeredRule>Triggered Rule #1</triggeredRule>
  ...
  ...
  ...
  <triggeredRule>Triggered Rule #N</triggeredRule>
</advancedFraudResults>
4.23 affiliate

The affiliate element is an optional child element of the merchantData element. You can use it to track transactions associated with various affiliate organizations.

**Type** = String; **minLength** = N/A; **maxLength** = 25

**Parent Elements:**

*merchantData*

**Attributes:**

None

**Child Elements:**

None
4.24 affluence

The affluence element is an optional child of the enhancedAuthResponse element and defines whether the card used falls into one of the two defined affluent categories. If the card does not meet the definition of either category, the system does not return the affluence element.

NOTE: Please consult your Relationship Manager for additional information concerning this feature.

Type = String (enum); minLength = N/A; maxLength = N/A

Parent Elements:
enhancedAuthResponse

Attributes:
None

Child Elements:
None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MASS AFFLUENT</td>
<td>Returned for certain Visa and MasterCard cards indicating high income customers (&gt;100K annual income).</td>
</tr>
<tr>
<td>AFFLUENT</td>
<td>Returned for certain Visa and MasterCard cards indicating high income customers with high spending patterns (&gt;100K annual income and &gt;40K in card usage).</td>
</tr>
</tbody>
</table>

NOTE: The Affluence indicator applies only to certain Visa and MasterCard cards. This indicator does not apply to American Express or Discover cards.
4.25 allowPartialAuth

The `allowPartialAuth` element is an optional child of both Authorization and Sale transactions, which allows you to specify whether to authorize a partial amount if the entire requested authorization amount exceeds available credit/balance.

**NOTE:** If you settle in a currency other than US or Canada, you cannot use partial authorizations. When submitting transactions for private label gift cards (card type of GC), the use of partial authorizations is determined by a setting in the Merchant Profile (on or off globally). This flag has no effect.

**Type** = Boolean;  **Valid Values** = true or false

**Parent Elements:**
- authorization
- sale

**Attributes:**
- None

**Child Elements:**
- None

**NOTE:** For a Sale transaction, the deposit will be for the partial amount.
4.26 amexAggregatorData

The `amexAggregatorData` element defines Amex Aggregator specific information in the cnpAPI. The system does not use the information unless you are designated as an Aggregator by American Express.

**Parent Elements:**
authorization, captureGivenAuth, credit, forceCapture, sale

**Attributes:**
None

**Child Elements: (Required)**
sellerId, sellerMerchantCategoryCode

**Example:** amexAggregatorData Structure

```xml
<amexAggregatorData>
  <sellerId>Seller Id</sellerId>
  <sellerMerchantCategoryCode>MerchantCategoryCode</sellerMerchantCategoryCode>
</amexAggregatorData>
```
4.27 amount

The `amount` element is a child of several elements. When used in a payment transaction, this element defines the amount of the transaction. When `amount` is a child of the `subscription` element, it defines the amount of the recurring payment. As a child of the `activate`, `load`, or `unload` transactions, `amount` defines the initial value of a newly activated gift card, the amount loaded onto a reloadable gift card, or the amount unloaded from a gift card, respectively. When used in an Instruction-Based Dynamic Payout transaction type, it specifies the amount of funds to transfer. Supply the value in cents without a decimal point. For example, a value of 1995 signifies $19.95.

**NOTE:** For the following currencies, the value you submit is the amount without implied decimal: Burundian Franc, Chilean Peso, Comoro Franc, Djibouti Franc, Guinea Franc, Iceland Krona, Japanese Yen, South Korean Won, Vanuatu Vatu, Paraguayan Guarani, Rwandan Franc, Vietnamese Dong, Uganda Shilling, CFA Franc BEAC, CFA Franc BCEAO, and CFP Franc. For example, if the currency is Japanese Yen, entering 2000 means 2000 Yen, NOT 20.00 Yen.

Type = Integer; `totalDigits` = 12

**Parent Elements:**

The `amount` element is a required child of each of the following Parent Elements: `activate`, `authorization`, `credit` (required if the original transaction was not processed by Worldpay), `captureGivenAuth`, `echeckCredit` (required if the original transaction was not processed by Worldpay), `echeckSale`, `echeckVerification`, `forceCapture`, `fraudCheck`, `load`, `sale`, `unload`

This element is also a required child of the following Instruction-Based Dynamic Payout transaction types: `payFacCredit`, `payFacDebit`, `physicalCheckCredit`, `physicalCheckDebit`, `reserveCredit`, `reserveDebit`, `submerchantCredit`, `submerchantDebit`, `vendorCredit`, `vendorDebit`, `fastAccessFunding`

The `amount` element is an optional child of each of the following Parent Elements: `authReversal`, `capture`, `credit`, `echeckCredit`, `subscription`

**NOTE:** For all cases where the `amount` element is optional, except when a child of the `subscription` element, if you do not specify a value, the system uses the entire amount from the referenced (by `littleTxnId`) transaction. When `amount` is a child of the `subscription` element, if you do not specify a value, the amount defaults to the value defined in the referenced recurring plan (`planCode` element).

**Attributes:**

None

**Child Elements:**

None
4.28 androidpayResponse

The `androidpayResponse` element is an optional child of several transaction types and is returned in response messages, when the `orderSource` in the request is `androidpay`.

**Parent Elements:**
`authorizationResponse`, `registerTokenResponse`, `saleResponse`

**Attributes:**
None

**Child Elements (all optional):**
`cryptogram`, `expMonth`, `expYear`, `eciIndicator`

**Example: androidpayResponse Structure**

```xml
<androidpayResponse>
  <cryptogram>Payment Cryptogram</cryptogram>
  <expMonth>Expiration Month</expMonth>
  <expYear>Expiration Year</expYear>
  <eciIndicator>eCommerce Indicator</eciIndicator>
</androidpayResponse>
```
4.29 applepay

The applepay element is an optional child of several transaction types and takes the place of the card element in Apple Pay transaction where the merchant does not decrypt the (Apple Pay) PKPaymentToken prior to submitting the transaction. It contains child elements for the component parts of the PKPaymentToken.

Parent Elements:
authorization, registerTokenRequest, sale

Attributes:
None

Child Elements (all required):
data, header, signature, version

Example: applepay Structure

    <applepay>
        <data>User Name</data>
        <header>
            <applicationData>Base64 Hash of App Data Property</applicationData>
            <ephemeralPublicKey>Base64 Encoded Ephemeral Public Key</ephemeralPublicKey>
            <publicKeyHash>Base64 Hash of Public Merchant Key Cert</publicKeyHash>
            <transactionId>Hex Transaction Id</transactionId>
        </header>
        <signature>Signature of Payment and Header Data</signature>
        <version>Payment Token Version Info</version>
    </applepay>
## 4.30 applepayResponse

The `applepayResponse` element is an optional child of several transaction types and is returned in response messages, when the request includes the `applepay` element.

**Parent Elements:**
- `authorizationResponse`
- `registerTokenResponse`
- `saleResponse`

**Attributes:**
None

**Child Elements:**
- `applicationPrimaryAccountNumber`
- `applicationExpirationDate`
- `currencyCode`
- `transactionAmount`
- `cardholderName`
- `deviceManufacturerIdentifier`
- `paymentDataType`
- `onlinePaymentCryptogram`
- `eciIndicator`

**Example:** applepayResponse Structure

```xml
<applepayResponse>
    <applicationPrimaryAccountNumber>App PAN</applicationPrimaryAccountNumber>
    <applicationExpirationDate>App PAN Exp Date</applicationExpirationDate>
    <currencyCode>Currency Code</currencyCode>
    <transactionAmount>Amount of Transaction</transactionAmount>
    <cardholderName>Name of cardholder</cardholderName>
    <deviceManufacturerIdentifier>Id of Device Mfr</deviceManufacturerIdentifier>
    <paymentDataType>Type of Payment Data</paymentDataType>
    <onlinePaymentCryptogram>Payment Cryptogram</onlinePaymentCryptogram>
    <eciIndicator>eCommerce Indicator</eciIndicator>
</applepayResponse>
```

**IMPORTANT:** The system never returns the `applicationPrimaryAccountNumber` element in a `registerTokenResponse` message, since this transaction type includes a token replacing the application PAN.
4.31 applicationData

The applicationData element is an optional child of the header element and provides the SHA-256 hash, hex encoded string of the original PKPaymentRequest of the Apple Pay transaction.

**Type** = Hex Encoded String; **minLength** = N/A; **maxLength** = 10000

**Parent Elements:**
- header

**Attributes:**
- None

**Child Elements:**
- None
4.32 applicationExpirationDate

The `applicationExpirationDate` element is an optional child of the `applepayResponse` element and defines expiration date of the application primary account number.

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
applepayResponse

**Attributes:**
None

**Child Elements:**
None
4.33 applicationPrimaryAccountNumber

The `applicationPrimaryAccountNumber` element is an optional child of the `applepayResponse` element and defines the primary account number associated with the application.

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
- applepayResponse

**Attributes:**
None

**Child Elements:**
None

**IMPORTANT:** The system never returns the `applicationPrimaryAccountNumber` element (child of the `applepayResponse` element) in a `registerTokenresponse` message, since this transaction type includes a token replacing the application PAN.
4.34 approvedAmount

The approvedAmount element defines the dollar amount of the approval. It appears in an authorization or sale response only if the allowPartialAuth element is set to true in the request transaction. It can also appear as the child of a deactivateResponse message where it specifies the value of the Gift Card prior to deactivation.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**
authorizationResponse, saleResponse, deactivateResponse

**Attributes:**
None

**Child Elements:**
None
4.35 authAmount

The authAmount element is an optional child of the authInformation element and is used to define the dollar amount of the authorization for Capture Given Auth transactions.

Type = Integer; totalDigits = 8

Parent Elements:
authInformation

Attributes:
None

Child Elements:
None
4.36 authCode

The authCode element is an optional child of both the authorizationResponse and saleResponse elements. It is also a required child of the authInformation element (used in captureGivenAuth transactions), where it specifies the authorization code from the associated Authorization or Sale transaction.

Type = String; minLength = N/A; maxLength = 6

Parent Elements:
authInformation, authorizationResponse, saleResponse

Attributes:
None

Child Elements:
None
4.37 authDate

The `authDate` element is a required child of the `authInformation` element and defines the date of the associated Authorization transaction.

**Type** = Date; **Format** = YYYY-MM-DD

**Parent Elements:**

authInformation

**Attributes:**

None

**Child Elements:**

None
4.38 authenticatedByMerchant

The authenticatedByMerchant element is an optional child element of the cardholderAuthentication element. This element indicates whether or not the customer has logged in to a secure web site or has been authenticated by the call center ANI.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**

- cardholderAuthentication

**Attributes:**

- None

**Child Elements:**

- None
4.39 authentication

The authentication element is a required element of both the litleOnlineRequest and the batchRequest elements. It contains child elements used to authenticate that the XML message is from a valid user.

**NOTE:** The authentication element and its child elements, user and password, are used to identify the merchant submitting transactions. They do not provide other access to platform or to the iQ reporting system.

**Parent Elements:**
litleOnlineRequest, litleRequest

**Attributes:**
None

**Child Elements:**
Required: user, password

**Example: authentication Structure**

```xml
<authentication>
  <user>User Name</user>
  <password>Password</password>
</authentication>
```
4.40 authenticationResult

The `authenticationResult` element is an optional child element of the `fraudResult` element. It defines the 3DS results from either Verified by Visa, MasterCard SecureCode, or Discover ProtectBuy. For a list of possible values, please refer to 3DS Authentication Result Codes on page 806.

**Type** = String; **minLength** = N/A; **maxLength** = 1

**Parent Elements:**

fraudResult

**Attributes:**

None

**Child Elements:**

None
4.41 authenticationTransactionId

The `authenticationTransactionId` element is an optional child of the `cardholderAuthentication` element. This element defines the Verified by Visa Transaction Id or the Visa TAVV (Token Authentication Verification Value). This element is not used for Discover ProtectBuy transactions.

You must include this element for Visa transactions, when the `orderSource` element is set to `3dsAuthenticated`.

**Type** = Base 64 Encoded String; **minLength** = N/A; **maxLength** = 28

**Parent Elements:**
- `cardholderAuthentication`

**Attributes:**
- None

**Child Elements:**
- None
4.42 authenticationValue

The `authenticationValue` element is an optional child of the `cardholderAuthentication` element. This element defines either the Visa CAVV value (fixed length 28 characters) or the MasterCard UCAF value (variable length up to 32 characters).

You must include this element for Visa and MasterCard transactions, when the `orderSource` element is set to `3dsAuthenticated`, as well as MasterCard transactions if the `orderSource` element is set to `3dsAttempted`.

You also use this element to send Worldpay the Apple Pay cryptogram if you decrypt the Apple Pay PKPaymentToken (see Merchant Decryption of Apple Pay PKPaymentToken on page 55).

NOTE: The Apple Pay cryptogram is already BASE64 encoded. Do not double encode the cryptogram.

Type = Base 64 Encoded String; `minLength` = N/A; `maxLength` = 56

Parent Elements:
`cardholderAuthentication`

Attributes:
None

Child Elements:
None
### 4.43 authInformation

The `authInformation` element is a required child of the `captureGivenAuth` element. It contains child elements used to provide details concerning the external (to the systems) Authorization.

**Parent Elements:**
- `captureGivenAuth`

**Attributes:**
- None

**Child Elements:**
- **Required:** `authDate`, `authCode`
- **Optional:** `fraudResult`, `authAmount`

**Example:** authInformation Structure

```xml
<authInformation>
  <authDate>Auth Date</authDate>
  <authCode>Approval Code</authCode>
  <fraudResult>
    <avsResult>AVS Verification Result Code</avsResult>
    <cardValidationResult>Validation Result Code</cardValidationResult>
    <authenticationResult>Verified by Visa Result Code</authenticationResult>
  </fraudResult>
  <authAmount>Amount of Authorization</authAmount>
</authInformation>
```
4.44 authorization

The authorization element is the parent element for all Authorization transactions. You can use this element in either Online or Batch transactions.

Parent Elements:
- litleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: orderId, amount, orderSource, (choice of) card, paypal, paypage, mpos, token, or applepay, cardholderAuthentication

NOTE: The cardholderAuthentication child element is required only for 3-D Secure transactions.

Optional: customerId, billToAddress, shipToAddress, billMeLaterRequest, processingInstructions, pos, customBilling, taxType, enhancedData, amexAggregatorData, allowPartialAuth, healthcareIIAS, merchantData, recyclingRequest, fraudFilterOverride, surchargeAmount, debtRepayment, recurringRequest, advancedFraudChecks, secondaryAmount, wallet, processingType, origOrderId, password, originalNetworkTransactionId, originalTransactionAmount
4.45 authorizationResponse

The authorizationResponse element is the parent element for information returned to you in response to an Authorization transaction. It can be a child of either a litleOnlineResponse element or a batchResponse element.

Parent Elements:

litleOnlineResponse, batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Authorization transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Authorization transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Authorization transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: litleTxnId, orderId, response, responseTime, message

Optional: postDate, cardProductId (see Note below), authCode, authorizationResponseSubCode (see Note below), approvedAmount, accountInformation, fraudResult, billMeLaterResponseData, tokenResponse, enhancedAuthResponse, accountUpdater, recycling, recurringResponse, giftCardResponse, applepayResponse, cardSuffix, androidpayResponse, networkTransactionId

NOTE: The postDate child element is returned only in responses to Online transactions.

The cardProductId element returns a raw code referencing the card type. Please consult your Relationship Manager for additional information.

The authorizationResponseSubCode element is not used at this time.
4.46 authorizationSourcePlatform

The `authorizationSourcePlatform` element is an optional child element of the `billMeLaterRequest` element. This element defines the physical platform that was used for submitting the authorization request (not the order). Specify as needed for auditing and re-authorization management purposes.

**Type** = String; **minLength** = N/A; **maxLength** = 1 (valid values below)

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>application processing</td>
</tr>
<tr>
<td>B</td>
<td>batch capture, recurring or mail order</td>
</tr>
<tr>
<td>C</td>
<td>call center</td>
</tr>
<tr>
<td>F</td>
<td>fulfillment/order management</td>
</tr>
<tr>
<td>K</td>
<td>kiosk</td>
</tr>
<tr>
<td>M</td>
<td>mobile device gateway</td>
</tr>
<tr>
<td>P</td>
<td>processor or gateway reauthorization</td>
</tr>
<tr>
<td>R</td>
<td>retail POS</td>
</tr>
</tbody>
</table>

**Parent Elements:**

`billMeLaterRequest`

**Attributes:**

None

**Child Elements:**

None
authReversal

The authReversal element is the parent element for all Authorization Reversal transactions. You can use this element in either Online or Batch transactions. Also see Notes on the Use of Authorization Reversal Transactions on page 70. Also, if you use the Recycling Engine, you can use the authReversal transaction to halt the recycling of an authorization transaction.

Parent Elements:
litleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: littleTxnId

Optional: amount, actionReason, paypalNotes, surchargeAmount

**NOTE:** If you do not specify an amount child element, the system reverses the full amount from the associated Authorization transaction.
4.48 authReversalResponse

The authReversalResponse element is the parent element for information returned to you in response to an Auth Reversal transaction. It can be a child of either a litleOnlineResponse element or a batchResponse element.

Parent Elements:
- litleOnlineResponse
- batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Authorization Reversal transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Authorization Reversal transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Authorization Reversal transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
- Required: litleTxnId, response, responseTime, message
- Optional: postDate

NOTE: The system returns the postDate child element only in responses to Online transactions.
4.49 availableBalance

The availableBalance element is a required child element of the fundingSource element and an optional child of the giftCardResponse element. It defines the outstanding available balance on the submitted prepaid or Gift Card. If the balance cannot be determined, the element returns, “Not Available.”

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
- fundingSource, giftCardResponse

**Attributes:**
None

**Child Elements:**
None

**NOTE:** The fundingSource element and its child elements, type and availableBalance are associated with the Insights features (see Issuer Insights on page 24.)

Please consult your Relationship Manager for additional information.
4.50 avsResult

The avsResult element is an optional child element of the fraudResult element. It defines the Address Verification response code returned by the networks. For a list of possible values, please refer to AVS Response Codes on page 808.

_Type_ = String; _minLength_ = N/A; _maxLength_ = 2

**Parent Elements:**
fraudResult

**Attributes:**
None

**Child Elements:**
None
4.51 balanceInquiry

The balanceInquiry element is the parent element for the transaction type that queries the available balance of a Gift Card.

Parent Elements:
litleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)
orderId, orderSource, card
4.52 balanceInquiryResponse

The `balanceInquiryResponse` element is the parent element for information returned to you in response to a `balanceInquiry` transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**
- `litleOnlineResponse`
- `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String   | Yes       | The response returns the same value submitted in the Balance Inquiry transaction.  
`minLength = 1  maxLength = 36` |
| customerId     | String   | No        | The response returns the same value submitted in the Balance Inquiry transaction.  
`minLength = N/A  maxLength = 50` |
| reportGroup    | String   | Yes       | The response returns the same value submitted in the Balance Inquiry transaction.  
`minLength = 1  maxLength = 25` |

**Child Elements:**

Required: `litleTxnId`, `orderId`, `response`, `responseTime`, `message`

Optional: `postDate`, `fraudResult`, `giftCardResponse`
4.53 batchRequest

This is the root element for all cnpAPI Batch requests.

Parent Elements:
- litleRequest

Attributes:

NOTE: Include the count and amount attributes for all transaction types included in the Batch. For example if you submit one Sale transaction of $10, include numSales="1" and saleAmount="1000". You must always include an amount, if you include a count. Do not include corresponding attributes, if the Batch does not include the transaction type.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>A unique string to identify this batchRequest within the system.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>numAuths</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Authorization transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>authAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Authorization transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>numAuthReversals</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of AuthReversal transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>authReversalAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of AuthReversal transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>numGiftCardAuthReversals</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of giftCardAuthReversal transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>--------------</td>
<td>-----------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>giftCardAuthReversalOriginalAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of giftCardAuthReversal transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numCaptures</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Capture transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>captureAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Capture transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numGiftCardCaptures</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of giftCardCapture transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>giftCardCaptureAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of giftCardCapture transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numCredits</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>creditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numGiftCardCredits</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of giftCardCredit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>giftCardCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of giftCardCredit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numForceCaptures</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Force Capture transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>forceCaptureAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Force Capture transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numSales</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Sale transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A  <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>saleAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Sale transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numCaptureGivenAuths</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Capture Given Auth transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A  <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>captureGivenAuthAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Capture Given Auth transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numEcheckSales</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Sale transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A  <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>echeckSalesAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of eCheck Sale transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numEcheckCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A  <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>echeckCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of eCheck Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits</strong> = 10</td>
</tr>
<tr>
<td>numEcheckVerification</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Verification transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength</strong> = N/A  <strong>maxLength</strong> = N/A</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>echeckVerificationAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of eCheck Verification transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>numEcheckRedeposit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Redeposit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numEcheckPreNoteSale</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Prenotification Sale transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numEcheckPreNoteCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of eCheck Prenotification Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numAccountUpdates</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Account Update transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numTokenRegistrations</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Token Registration transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numUpdateCardValidationNumOnTokens</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Update Card Validation Number request transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numCancelSubscriptions</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Cancel Subscription transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numUpdateSubscriptions</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Update Subscription transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numCreatePlans</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Create Plan transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numUpdatePlans</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Update Plan transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>numActivates</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of (Gift Card) Activate transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = N/A</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>numDeactives</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of (Gift Card) Deactivate transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>activateAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of (Gift Card) Activate transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500. $totalDigits = 10$</td>
</tr>
<tr>
<td>numLoads</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of (Gift Card) Load transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>loadAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of (Gift Card) Load transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500. $totalDigits = 10$</td>
</tr>
<tr>
<td>numUnloads</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of (Gift Card) Unload transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>unloadAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of (Gift Card) Unload transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500. $totalDigits = 10$</td>
</tr>
<tr>
<td>numBalanceInquiries</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of (Gift Card) Balance Inquiry transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>numPayFacCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of PayFac Credit transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>numPayFacDebit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of PayFac Debit transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>numSubmerchantCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Submerchant Credit transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>numSubmerchantDebit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Submerchant Debit transactions in the batchRequest. $minLength = N/A$ $maxLength = N/A$</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>numReserveCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Reserve Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numReserveDebit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Reserve Debit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numVendorCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Vendor Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numVendorDebit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Vendor Debit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numPhysicalCheckCredit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Physical Check Credit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numPhysicalCheckDebit</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Physical Check Debit transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>numFundingInstructionVoid</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total count of Funding Instruction Void transactions in the batchRequest.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = N/A</td>
</tr>
<tr>
<td>payFacCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of PayFac Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>payFacDebitAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of PayFac Debit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>submerchantCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Sub-merchant Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>totalDigits = 10</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Type</td>
<td>Required?</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>submerchantDebitAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Sub-merchant Debit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>reserveCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Reserve Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>reserveDebitAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Reserve Debit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>vendorCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Vendor Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>vendorDebitAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Vendor Debit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>physicalCheckCreditAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Physical Check Credit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>physicalCheckDebitAmount</td>
<td>Integer</td>
<td>No</td>
<td>Defines the total dollar amount of Physical Check Debit transactions in the batchRequest. The decimal point is implied. For example, you enter $25.00 as 2500.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>totalDigits = 10</strong></td>
</tr>
<tr>
<td>merchantId</td>
<td>String</td>
<td>Yes</td>
<td>A unique string to identify the merchant within the system.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength = N/A  maxLength = 50</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: International currencies are supported on a per merchantId basis.</td>
</tr>
</tbody>
</table>
### Child Elements:

At least one of the following required: `activate`, `authorization`, `authReversal`, `balanceInquiry`, `cancelSubscription`, `capture`, `captureGivenAuth`, `createPlan`, `credit`, `deactivate`, `echeckCredit`, `echeckPreNoteSale`, `echeckRedeposit`, `echeckSale`, `echeckVerification`, `forceCapture`, `giftCardAuthReversal`, `giftCardCapture`, `giftCardCredit`, `load`, `registerTokenRequest`, `sale`, `unload`, `updateCardValidationNumOnToken`, `updatePlan`, `updateSubscription`, `payFacCredit`, `payFacDebit`, `reserveCredit`, `reserveDebit`, `submerchantCredit`, `submerchantDebit`, `vendorCredit`, `vendorDebit`

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sameDayFunding</td>
<td>Boolean</td>
<td>No</td>
<td>Used for Dynamic Payout Funding Instructions only. Set to <strong>true</strong> to mark this Batch of Funding Instructions for same day funding. Also, see <a href="#">Same Day Funding</a> on page 852.</td>
</tr>
</tbody>
</table>
4.54 batchResponse

The `batchResponse` element is the parent element for information returned to you in response to a batch you submitted for processing. It is a child of a `litleResponse` element.

**Parent Elements:**

`litleResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Batch Request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 25</code></td>
</tr>
<tr>
<td>littleBatchId</td>
<td>Long</td>
<td>Yes</td>
<td>A unique value assigned by Worldpay to identify the batch.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 19</code></td>
</tr>
<tr>
<td>merchantId</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Batch Request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 50</code></td>
</tr>
</tbody>
</table>

**Child Elements:**


**NOTE:** The `batchResponse` contains child elements corresponding to the requests submitted in the `batchRequest`. For example, if the `batchRequest` contained 10 `authorization` and 8 `capture` transactions, the `batchResponse` would contain 10 `authorizationResponse` and 8 `captureResponse` transactions.
4.55 beginningBalance

The beginningBalance element is an optional child of the giftCardResponse element. It defines the available balance on the submitted Gift Card prior to the requested operation.

NOTE: Although included in the schema, the beginningBalance, endingBalance, and cashBackAmount elements are not currently supported.

Type = String; minLength = N/A; maxLength = 20

Parent Elements:
giftCardResponse

Attributes:
None

Child Elements:
None
4.56 billingDate

The `billingDate` element is an optional child of the `updateSubscription` element that defines the new date for the recurring billing when the scheduled date need to be changed.

**Type** = Date; **Format** = YYYY-MM-DD

**Parent Elements:**
 updateSubscription

**Attributes:**
 None

**Child Elements:**
 None
4.57 billMeLaterRequest

The `billMeLaterRequest` element is the parent of several child elements used to define merchant, product type, terms and conditions, and other items for PayPal Credit authorization transactions, or where you must reference an external (to Worldpay) BML transaction.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Parent Elements:**
- authorization
- captureGivenAuth
- credit
- sale

**Attributes:**
None

**Child Elements:** (all optional)
- bmlMerchantId
- bmlProductType
- itemCategoryCode
- termsAndConditions
- preapprovalNumber
- virtualAuthenticationKeyPresenceIndicator
- virtualAuthenticationKeyData
- authorizationSourcePlatform

**NOTE:** The following elements appear in the schema as children of the `billMeLaterRequest` element, but are not used at this time:
- authorizationSourcePlatform
- customerBillingAddressChanged
- customerEmailChanged
- customerPasswordChanged
- customerPhoneChanged
- merchantPromotionalCode
- secretQuestionAnswer
- secretQuestionCode

**Example:** billMeLaterRequest

```xml
<billMeLaterRequest>
    <bmlMerchantId>bmlMerchantId</bmlMerchantId>
    <bmlProductType>bmlProductType</bmlProductType>
    <termsAndConditions>termsAndConditions</termsAndConditions>
    <preapprovalNumber>preapprovalNumber</preapprovalNumber>
    <virtualAuthenticationKeyPresenceIndicator> Indicator
    </virtualAuthenticationKeyPresenceIndicator>
    <virtualAuthenticationKeyData> virtualAuthenticationKeyData
```
</virtualAuthenticationKeyData>
   <itemCategoryCode>itemCategoryCode</itemCategoryCode>
   <authorizationSourcePlatform>platformType</authorizationSourcePlatform>
</billMeLaterRequest>
4.58 billMeLaterResponseData

The billMeLaterResponseData element is the parent of several child elements used in the response XML for PayPal Credit authorization transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Parent Elements:**
authorizationResponse, saleResponse

**Attributes:**
None

**Child Elements:**
bmlMerchantId, creditLine, addressIndicator

**NOTE:** The following elements appear in the schema as children of the billMeLaterResponseData element, but are not used at this time:
- approvedTermsCode
- loanToValueEstimator
- promotionalCodeOffer
- riskEstimator
- riskQueueAssignment

**Example: billMeLaterResponseData Structure**

```xml
<billMeLaterResponseData>
  <bmlMerchantId>bmlMerchantId</bmlMerchantId>
  <creditLine>creditLine</creditLine>
  <addressIndicator>addressIndicator</addressIndicator>
</billMeLaterResponseData>
```
4.59  billToAddress

The billToAddress element contains several child elements that define the postal mailing address (and telephone number) used for billing purposes. It also contains several elements used for the eCheck verification process.

Parent Elements:
authorization, captureGivenAuth, credit, echeckCredit (required if original transaction was not processed by Worldpay), echeckPreNoteCredit, echeckPreNoteSale, echeckSale, echeckVerification, forceCapture, fraudCheck, sale, updateSubscription

Attributes:
None

Child Elements: (all optional)
name, firstName, middleInitial, lastName, companyName, addressLine1, addressLine2, addressLine3, city, state, zip, country, email, phone

NOTE: You must submit the name element for echeckSale and echeckCredit transactions. If you do not submit the customer name in these eCheck transactions, we return the response 709 - Invalid Data.

You must submit the name, country, and email elements in sale transactions using the sepaDirectDebit method of payment. Failure to include the these elements results in a 901 - Missing Name Response Code, 903 - Missing Billing Country Code, or 905 - Missing Email Address respectively.

For an eCheckVerification transaction, you must submit the firstName and lastName elements instead of the name element (middleInitial is optional). For a corporate account you must include the companyName element in addition to the firstName and lastName elements. In both cases, you also must include the address, city, state, zip and phone information.

For a corporate account, if you do not have the name of the check issuer, you can use a value of “ unavailable” for the firstName and lastName elements.

Example:  billToAddress Structure

    <billToAddress>
        <name>Customer’s Full Name</name>
        <firstName>Customer’s First Name</firstName>
        <middleInitial>Customer’s Middle Initial</middleInitial>
        <lastName>Customer’s Last Name</lastName>
        <companyName>Company’s Name</companyName> (include for echeckVerification of corporate account)
        <addressLine1>Address Line 1</addressLine1>
        <addressLine2>Address Line 2</addressLine2>
        <addressLine3>Address Line 3</addressLine3>
<city>City</city>
<state>State Abbreviation</state>
<zip>Postal Code</zip>
<country>Country Code</country>
<email>Email Address</email>
<phone>Telephone Number</phone>
</billToAddress>
4.60 bin

The bin element provides the 6-digit Bank (or Issuer) Identification Number of the Issuing Bank. The system returns this value in XML responses when issuing new tokens to replace Visa or MasterCard account numbers.

For Discover and American Express cards, this element is empty in a registerTokenResponse.

For Discover, when included with Account Updater information in a payment transaction response, the bin element will have a value of discov.

Type = String; minLength = 0; maxLength = 6

Parent Elements:
The bin element is an optional child of each listed parent element.

registerTokenResponse, tokenResponse, newCardTokenInfo, originalCardTokenInfo, originalToken, updatedToken

Attributes:
None

Child Elements:
None
4.61 bmlMerchantId

The `bmlMerchantId` element is a value assigned by PayPal Credit to identify the merchant within the PayPal Credit system.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = Long; **minLength** = N/A; **maxLength** = 19

**Parent Elements:**
The `bmlMerchantId` element is an optional child of each listed parent element.

`billMeLaterRequest`, `billMeLaterResponseData`

**Attributes:**
None

**Child Elements:**
None
4.62  bmlProductType

The bmlProductType element is a value assigned by PayPal Credit to identify the merchant account type within the PayPal Credit system.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = String; **minLength** = 2; **maxLength** = 2

**Parent Elements:**

The bmlProductType element is an optional child of each listed parent element.

*billMeLaterRequest, billMeLaterResponseData*

**NOTE:** At this time, the only valid value for this element is BL.

**Attributes:**

None

**Child Elements:**

None
4.63 bypassVelocityCheck

The `bypassVelocityCheck` element is an optional child of the `processingInstructions` element, which allows you to specify whether or not the system performs velocity checking on the transaction.

![NOTE: Velocity Checking is not currently supported.](image)

Type = Boolean; Valid Values = true or false

Parent Elements:

- `processingInstructions`

Attributes:

None

Child Elements:

None
4.64 campaign

The campaign element is an optional child element of the merchantData element. You can use it to track transactions associated with various marketing campaigns.

Type = String; minLength = N/A; maxLength = 25

Parent Elements:
merchantData

Attributes:
None

Child Elements:
None
4.65 cancelSubscription

The `cancelSubscription` element is the parent element for the transaction that cancels a subscription. You can use this element in either Online or Batch transactions.

**Parent Elements:**
littleOnlineRequest, batchRequest

**Attributes:**
None

**Child Elements:**
Required: `subscriptionId`
cancelSubscriptionResponse

The cancelSubscriptionResponse element is the parent element for the response to a cancelSubscription transaction.

Parent Elements:
litleOnlineResponse, batchResponse

Attributes:
None

Child Elements:
Required: subscriptionId, litleTxnId, response, message, responseTime
4.67 capability

The capability element is a required child of the pos element, which describes the capability of the point of sale terminal.

Type = String (Enum); minLength = N/A; maxLength = N/A

Parent Elements:

pos

Attributes:

None

Child Elements:

None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notused</td>
<td>terminal not used</td>
</tr>
<tr>
<td>magstripe</td>
<td>magnetic stripe reader capability</td>
</tr>
<tr>
<td>keyedonly</td>
<td>keyed entry only capability</td>
</tr>
</tbody>
</table>

NOTE: For CAT (Cardholder Activated Terminal) transactions, the capability element must be set to magstripe, the cardholderId element must be set to nopin, and the catLevel element must be set to self service.
4.68 capture

The capture element is the parent element for all Capture (deposit) transactions. You can use this element in either Online or Batch transactions.

Parent Elements:
littleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
<tr>
<td>partial</td>
<td>Boolean</td>
<td>No</td>
<td>If there is more than one capture that references the same littleTxnId, set this attribute to “true” for each of those partial captures. The default value is false.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: If you settle in a currency other than US or Canada, you cannot use partial captures.</td>
</tr>
</tbody>
</table>

Child Elements:

Required: littleTxnId, PayPalOrderComplete (required only if closing a PayPal order)

Optional: amount, enhancedData, PayPalNotes, pin, processingInstructions, secondaryAmount, surchargeAmount

**NOTE:** If you do not specify an amount, the system uses the full amount from the associated Authorization transaction
4.69  captureAmount

The `captureAmount` element is a required child of the `giftCardCapture` element and defines the amount of the capture (deposit). This value must be less than or equal to the value of the `originalAmount` element. Setting this element to a value less than the `originalAmount` implies that this is a partial capture.

**Type** = Integer; **totalDigits** = 12

**Parent Elements:**
- `giftCardCapture`

**Attributes:**
- None

**Child Elements:**
- None
4.70 captureGivenAuth

The `captureGivenAuth` element is the parent element for all Capture Given Auth transactions. These are specialized Capture transactions used when the `litleTxnId` for the associated Authorization is unknown or when the Authorization occurred outside our system. You can use this element in either Online or Batch transactions.

Parent Elements:
- `litleOnlineRequest`, `batchRequest`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1   maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A   maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1   maxLength = 25</code></td>
</tr>
</tbody>
</table>

Child Elements:

Required: `orderId`, `authInformation`, `amount`, `orderSource`, choice of `card`, `token`, `mpos`, or `paypage`

Optional: `billToAddress`, `shipToAddress`, `customBilling`, `taxType`, `enhancedData`, `processingInstructions`, `pos`, `amexAggregatorData`, `merchantData`, `secondaryAmount`, `surchargeAmount`, `debtRepayment`, `processingType`, `originalNetworkTransactionId`, `originalTransactionAmount`

**NOTE:** If you do not specify an `amount` child element, the system uses the full amount from the associated Authorization transaction.
4.71 captureGivenAuthResponse

The `captureGivenAuthResponse` element is the parent element for information returned to you in response to a Capture Given Auth transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**

- `litleOnlineResponse`
- `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String  | Yes       | The response returns the same value submitted in the Capture Given Auth transaction.  
|                |         |           | minLength = 1  maxLength = 36 |
| customerId     | String  | No        | The response returns the same value submitted in the Capture Given Auth transaction.  
|                |         |           | minLength = N/A  maxLength = 50 |
| reportGroup    | String  | Yes       | The response returns the same value submitted in the Capture Given Auth transaction.  
|                |         |           | minLength = 1  maxLength = 25 |

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`
Optional: `postDate`, `tokenResponse`, `giftCardResponse`

**NOTE:** The system returns the `postDate` child element only in responses to Online transactions.
4.72 captureResponse

The `captureResponse` element is the parent element for information returned to you in response to a Capture transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**
- `litleOnlineResponse`, `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`

Optional: `postDate`, `accountUpdater`, `giftCardResponse`

**NOTE:** The system returns the `postDate` child element only in responses to Online transactions.
The `card` element defines payment card information. It is a required element for most transaction types unless the transaction uses an alternate payment method such as PayPal. It contains one or more child elements depending upon whether the transaction is a card-not-present or a card-present (face-to-face) transaction.

**Parent Elements:**
activate, activateReversal, accountUpdate, authorization, balanceInquiry, captureGivenAuth, credit, deactivate, deactivateReversal, depositReversal, forceCapture, giftCardAuthReversal, giftCardCapture, giftCardCredit, load, loadReversal, refundReversal, sale, unload, unloadReversal, updateSubscription, fastAccessFunding

**Attributes:**
None

**Child Elements:**
For card-not-present transactions (Required): `type`, `number`, `expDate`
For card-present transactions (Required): `track`
For both transactions types (Optional): `cardValidationNum`
For Gift Card transactions (Optional): `pin`

**Example:**  card Structure - Card-Not-Present

```xml
<card>
  <type>Card Type Abbreviation</type>
  <number>Account Number</number>
  <expDate>Expiration Date</expDate>
  <cardValidationNum>Card Validation Number</cardValidationNum>
</card>
```

**Example:**  card Structure - Card-Present

```xml
<card>
  <track>Magnetic Stripe Read</track>
</card>
```

**Example:**  card Structure - Gift Card (Card-Not-Present)

```xml
<card>
  <type>GC</type>
  <number>Account Number</number>
  <expDate>Expiration Date</expDate>
</card>
```
<cardValidationNum>Card Validation Number</cardValidationNum>
<pin>Pin Number</pin>
</card>
4.74 cardAcceptorTaxId

The `cardAcceptorTaxId` element is an optional child of the `detailTax` element and defines the merchant's Tax Id. This ID is nine digits long if the merchant is domiciled in the U.S. If the card acceptor tax ID is unknown, do not include this element.

Type = String; minLength = 1; maxLength = 20

Parent Elements:

detailTax

Attributes:
None

Child Elements:
None
4.75 cardholderAuthentication

The cardholderAuthentication element is an optional child element of the Authorization and Sale transactions. The children of this element have two purposes. The first is to define Verified by Visa or MasterCard SecureCode data in the Authorization or Sale transactions (authenticationValue, authenticationTransactionId, and authenticatedByMerchant elements). The remaining child element, customerIpAddress, as well as the authenticationTransactionId, are used in PayPal Credit transactions. The customerIpAddress element can also be used to supply the customer IP Address by merchants enabled for American Express Advanced AVS services.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Parent Elements:**
authorization, sale

**Attributes:**
None

**Child Elements:**
authenticationValue, authenticationTransactionId, customerIpAddress, authenticatedByMerchant

**Example:** cardholderAuthentication Structure

```xml
<cardholderAuthentication>
  <authenticationValue>BwABBQI1AgAAAAAgJDUCAAAAAAA=</authenticationValue>
  <authenticationTransactionId>EaOMucALHqLaEGAgk=</authenticationTransactionId>
  <customerIpAddress>Customer Ip</customerIpAddress>
  <authenticatedByMerchant>Boolean</authenticatedByMerchant>
</cardholderAuthentication>
```

**NOTE:** The values for the <authenticationValue> and <authenticationTransactionId> elements in the example above have been truncated.
4.76 cardholderId

The cardholderId element is a required child of the pos element, which describes the method used for cardholder identification at the point of sale.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
pos

**Attributes:**
None

**Child Elements:**
None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>signature</td>
<td>customer signature obtained</td>
</tr>
<tr>
<td>pin</td>
<td>PIN number</td>
</tr>
<tr>
<td>nopin</td>
<td>unattended terminal - no PIN pad</td>
</tr>
<tr>
<td>directmarket</td>
<td>mail, telephone, or online</td>
</tr>
</tbody>
</table>

**NOTE:** For CAT (Cardholder Activated Terminal) transactions, the capability element must be set to magstripe, the cardholderId element must be set to nopin, and the catLevel element must be set to self service.
4.77  cardholderName

The cardholderName element is an optional child of the applepayResponse element and provides the name of the cardholder whose card initiated the Apple Pay transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 512

**Parent Elements:**
applepayResponse

**Child Elements:**
None
4.78 cardOrToken

The `cardOrToken` element is an abstract that allows the substitution of either the card or token element. You must specify one of the two substitution elements as a child of the `accountUpdate` element.

**Parent Elements:**

`accountUpdate`

**Substitution Options:**

`card`, `token`
4.79 cardProductType

The `cardProductType` element is an optional child of the `enhancedAuthResponse` element and whether the card used is commercial or consumer.

Type = String (enum); minLength = N/A; maxLength = N/A

Parent Elements:
- enhancedAuthResponse

Attributes:
None

Child Elements:
None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMERCIAL</td>
<td>The card is a commercial card.</td>
</tr>
<tr>
<td>CONSUMER</td>
<td>The card is a consumer card.</td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>The type of card is not known.</td>
</tr>
</tbody>
</table>
4.80 cardSuffix

The `cardSuffix` element is an optional child of both the `authorizationResponse` and `saleResponse` elements. It provides the last four digits of the actual PAN for Apple Pay transactions, when the underlying card is Visa, MasterCard, or Discover.

Type = String; `minLength = 3; maxLength = 6`

Parent Elements:
authorizationResponse, saleResponse

Attributes:
None

Child Elements:
None
4.81 cardValidationNum

The cardValidationNum element is an optional child of the card element, which you use to submit either the CVV2 (Visa), CVC2 (MasterCard), or CID (American Express and Discover) value.

**NOTE:** Some American Express cards may have a 4-digit CID on the front of the card and/or a 3-digit CID on the back of the card. You can use either of the numbers for card validation, but not both.

When you submit the CVV2/CVC2/CID in a registerTokenRequest, the platform encrypts and stores the value on a temporary basis for later use in a tokenized Auth/Sale transaction submitted without the value. This is done to accommodate merchant systems/workflows where the security code is available at the time of token registration, but not at the time of the Auth/Sale. If for some reason you need to change the value of the security code supplied at the time of the token registration, use an updateCardValidationNumOnToken transaction. To use the stored value when submitting an Auth/Sale transaction, set the cardValidationNum value to 000.

The cardValidationNum element is an optional child of the virtualGiftCardResponse element, where it defines the value of the validation Number associated with the Virtual Gift Card requested.

**NOTE:** The use of the cardValidationNum element in the registerTokenRequest only applies when you submit an accountNumber element.

Type = String; minLength = N/A; maxLength = 4

Parent Elements:
card, paypage, token, registerTokenRequest, updateCardValidationNumOnToken, virtualGiftCardResponse

Attributes:
None

Child Elements:
None
4.82  cardValidationResult

The `cardValidationResult` element is an optional child element of the `fraudResult` element. It defines the Card Validation response code returned by the networks. For a list of possible values, please refer to Card Validation Response Codes on page 812.

**Type** = String;  **minLength** = N/A;  **maxLength** = 2

**Parent Elements:**
- fraudResult

**Attributes:**
- None

**Child Elements:**
- None
4.83 cashBackAmount

The cashBackAmount element is an optional child of the giftCardResponse element. It defines the Cash Back amount provided to the Gift Card holder.

**NOTE:** Although included in the schema, the `beginningBalance`, `endingBalance`, and `cashBackAmount` elements are not currently supported.

Type = String; minLength = N/A; maxLength = 20

Parent Elements:
- giftCardResponse

Attributes:
- None

Child Elements:
- None
4.84 catLevel

The catLevel element is an optional child of the pos element, which describes the capability of the Cardholder Activated Terminal. Although optional in the schema, it is required for all CAT transactions.

Type = String (Enum); minLength = N/A; maxLength = N/A

Parent Elements:

pos

Attributes:
None

Child Elements:
None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>self service</td>
<td>Cardholder Activated Terminal level</td>
</tr>
</tbody>
</table>
4.85 ccdPaymentInformation

The ccdPaymentInformation element is an optional child of the echeck element. This element is intended for use by Payment Facilitators using Instruction Based Dynamic Payout to submit a description of the transaction. The description will appear in the extended detail section of the receiver’s bank statement, if that section is supported by the receiver’s bank.

*Type* = String; *minLength* = N/A; *maxLength* = 80

**Parent Elements:**
- accountInfo, echeck

**Attributes:**
None

**Child Elements:**
None
4.86 chargeback

The chargeback element is an optional child of the filtering element. To disable the chargeback filtering operation for a selected transaction include the chargeback element with a setting of false.

**Type** = Boolean; **Valid Value** = false

**NOTE:** Although included in the schema, the `<chargeback>` element is not supported. To override the chargeback filter for a selected transaction, use the fraudFilterOverride flag (see [fraudFilterOverride](#) on page 513). Please consult your Relationship Manager for additional information.

**Parent Elements:**

- filtering

**Attributes:**

None

**Child Elements:**

None
4.87 checkNum

The `checkNum` element is an optional child of the `echeck` element defining the check number of used in the transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 15

**Parent Elements:**
echeck

**Attributes:**
None

**Child Elements:**
None
4.88 checkoutId

The checkoutId element is an optional child of the token element specifying the low value token replacing the CVV value. You use this feature when you already have the consumer's card (i.e., token) on file, but wish the consumer to supply the CVV value for a new transaction. This LVT remains valid for 24 hours from the time of issue.

Type = String; minLength = 18; maxLength = 18

Parent Elements:

- token

Attributes:

None

Child Elements:

None

NOTE: For additional information about this element, please refer to the Worldpay eComm Enterprise eProtect Integration Guide.
4.89  city

The city element defines the customer's city name in the billToAddress and shipToAddress elements. In the customBilling element, city defines the location of the merchant for card-present transactions.

Type = String; minLength = N/A; maxLength = 35

Parent Elements:
billToAddress, shipFromPostalCode, customBilling

Attributes:
None

Child Elements:
None
4.90 clinicOtherAmount

The `clinicAmount` element is an optional child of the `healthcareAmounts` element and defines the healthcare amount used for the clinic/office visits. The decimal is implied. Example: 500 = $5.00.

Type = Integer; totalDigits = 8

Parent Elements:
Optional: healthcareAmounts

Attributes:
None

Child Elements:
None
4.91 code

The code element is a required child of the extendedCardResponse element. The code/message combination can be either 501 - The account was closed, or 504 - Contact the cardholder for updated information.

**Type** = String; **minLength** = N/A; **maxLength** = 3

**Parent Elements:**

extendedCardResponse

**Attributes:**

None

**Child Elements:**

None
4.92 commodityCode

The commodityCode element is an optional child of the lineItemData element, which specifies the Identifier assigned by the card acceptor that categorizes the purchased item. Although the schema defines it as an optional child of the enhancedData element, it is required by Visa for Level III interchange rates.

Type = String; minLength = 1; maxLength = 12

Parent Elements:
lineItemData

Attributes:
None

Child Elements:
None

NOTE: A commodity code is a numeric code representing a particular product or service. The code can be 3, 5, 7, or 11 digits in length. The longer the code the more granular the description of the product/service. For example, code 045 is used for Appliances and Equipment, Household Type, while code 04506 represents the sub-set of Appliances, Small Electric.

The codes are issued by the NIGP (National Institute of Governmental Purchasing. Their site, www.nigp.com, offers a subscription based code search engine, as well as downloadable lists for purchase.

You can also find many lists published online by performing a simple search on “Commodity Codes”.

4.93 **companyName**

The `companyName` element is an optional child of the `billToAddress` element, which specifies the name of the company associated with the corporate checking account. This element is required when performing an eCheck Verification of a check from a corporate account, as defined by the `<accType>` child of the `<echeck>` element.

**Type** = String; **minLength** = N/A; **maxLength** = 40

**Parent Elements:**
- `billToAddress`

**Attributes:**
- None

**Child Elements:**
- None
4.94 country

The country element defines the country portion of the postal mailing address in both the billToAddress and shipToAddress elements.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = 3

**NOTE:** The enumerations for this element are listed under <countryTypeEnum> in the cnpAPI Common XSD. The country names corresponding to the abbreviations can be found in the ISO 3166-1 standard.

All Country Codes are 2-character except for the United States, which accepts both US and USA.

The country element is required for sale transactions using the sepaDirectDebit method of payment. Failure to include the name element results in a 903 - Missing Billing Country Code Response Code.

Parent Elements:

billToAddress, shipFromPostalCode

Attributes:

None

Child Elements:

None
4.95 createAddOn

The `createAddOn` element is the parent of several child elements used to define an additional charge added to a new or existing subscription.

**Parent Elements:**

`subscription`, `updateSubscription`

**Attributes:**

None

**Child Elements (all Required):**

`addOnCode`, `name`, `amount`, `startDate`, `endDate`

**Example: customerInfo Structure**

```xml
<createAddOn>
  <name>Name of Add On</name>
  <amount>Amount of Add On</amount>
  <startDate>Start Date of Add On Charge</startDate>
  <endDate>End Date of Add On Charge</endDate>
</createAddOn>
```
4.96 createDiscount

The `createDiscount` element is the parent of several child elements used to define a discount to be applied to a new or existing subscription.

**Parent Elements:**
*subscription*, *updateSubscription*

**Attributes:**
None

**Child Elements (all Required):**
discountCode, name, amount, startDate, endDate

**Example: customerInfo Structure**

```xml
<createDiscount>
   <discountCode>Discount Reference Code</discountCode>
   <name>Name of Discount</name>
   <amount>Amount of Discount</amount>
   <startDate>Start Date of Discount</startDate>
   <endDate>End Date of Discount</endDate>
</createDiscount>
```
4.97 createPlan

The createPlan element is the parent of several child element that define the attributes of a recurring payment plan. You associate Plans with subscriptions to define the billing behavior for the recurring payment.

**Parent Elements:**
litleOnlineRequest, batchRequest

**Attributes:**
None

**Child Elements:**
planCode, name, description, intervalType, amount, numberOfPayments, trialNumberOfIntervals, trialIntervalType, active

**Example:** createPlan Structure

```xml
<createPlan>
  <planCode>Plan Reference Code</planCode>
  <name>Name of Plan</name>
  <description>Description of Plan</description>
  <intervalType>The Type of Interval</intervalType>
  <amount>Amount of Recurring Payment</amount>
  <numberOfPayments>1 to 99</numberOfPayments>
  <trialNumberOfIntervals>Number of Trial Period Intervals</trialNumberOfIntervals>
  <trialIntervalType>Type of Trial Period Interval</trialIntervalType>
  <active>true or false</active>
</createPlan>
```
4.98 createPlanResponse

The `createPlanResponse` element is the parent element for the response to a `createPlan` transaction.

**Parent Elements:**
- litleOnlineResponse, batchResponse

**Attributes:**
- None

**Child Elements:**
- planCode, litleTxnId, response, message, responseTime
The `credit` element is the parent element for all Credit transactions. You can use this element in either Online or Batch transactions.

**Parent Elements:**
- `litleOnlineRequest`
- `batchRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

For credits to transactions processed by Worldpay (Required): `litleTxnId`

For credits to transactions processed by Worldpay (Optional): `amount`, `payPalNotes`, `pin`, `secondaryAmount`, `surchargeAmount`

**NOTE:** If you do not specify an `amount`, the system uses the full amount from the associated settlement transaction.

You must include the `amount` element for credits to transactions not processed by Worldpay.

For credits to transactions not processed by Worldpay (Required): `orderId`, `amount`, `orderSource`, choice of `card`, `token`, `paypage`, `mpos`, `paypal`, or `applepay`

For credits to transactions not processed by Worldpay (Optional): `billToAddress`, `billMeLaterRequest`, `amexAggregatorData`

For both transaction types (Optional): `customBilling`, `taxType`, `enhancedData`, `processingInstructions`, `merchantData`, `actionReason`, `pos`
4.100 creditAmount

The creditAmount element is a required child of the giftCardCredit transaction, where it specifies the amount of the refund. Supply the value in cents without a decimal point. For example, a value of 1995 signifies $19.95.

Type = Integer; totalDigits = 12

Parent Elements:

giftCardCredit

Attributes:

None

Child Elements:

None
4.101 creditLine

The creditLine element is an optional child of the billMeLaterResponseData element and indicates the credit line of the customer. The amount is specified using a two-digit implied decimal.

Type = Integer; totalDigits = 8

Parent Elements:
billMeLaterResponseData

Attributes:
None

Child Elements:
None
4.102 creditLitleTxnId

The creditLitleTxnId element is the Transaction Id (litleTxnId) of a Credit transaction automatically submitted under the following conditions:

- You submitted a Void transaction to halt the recycling of a declined Sale transaction by the Recovery/Recycling Engine.
- The Sale transaction has already been approved and captured.
- Your Recovery/Recycling Engine configuration enables automatic refunds.
- The system has successfully submitted a Credit transaction on your behalf.

Type = Long; minLength = N/A; maxLength = 19

Parent Elements:
recycling

Attributes:
None

Child Elements:
None
4.103 creditResponse

The `creditResponse` element is the parent element for information returned to you in response to a Credit transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**
- `litleOnlineResponse`, `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the credit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the credit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the credit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`

Optional: `postDate`, `tokenResponse`, `giftCardResponse`, `applepayResponse`

**NOTE:** The system returns the `postDate` child element only in responses to Online transactions.
4.104 cryptogram

The cryptogram element is an optional child of the androidResponse element and provides the BASE64 Encoded signature cryptogram associated with the Google Pay transaction.

Type = Base 64 Encoded String; minLength = N/A; maxLength = 56

Parent Elements:
androidpayResponse

Attributes:
None

Child Elements:
None
4.105 currencyCode

The `currencyCode` element is an optional child of the `applepayResponse` element and provides the 3-character code for the currency used in the transaction.

Type = String; minLength = N/A; maxLength = 3

Parent Elements:
applepayResponse

Attributes:
None

Child Elements:
None
4.106 customAttribute1

The customAttribute1 through customAttribute5 elements are an optional children of the advancedFraudChecks element. These elements allow users of Advanced Fraud Tools with self-serve rules to submit additional custom data to ThreatMetrix for inclusion in the fraud evaluation process. For example, if you assigned a certain attribute to your customers for segmentation purposes, you might wish to submit the assigned value and also establish a ThreatMetrix rule so the value was included in the evaluation.

**NOTE:** Once you decide to use a particular custom attribute for a particular value set and establish a rule for its evaluation, you can not change its use.

**Type** = String; **minLength** = 1; **maxLength** = 200

**Parent Elements:**
advancedFraudChecks

**Attributes:**
None

**Child Elements:**
None
4.107 customBilling

The customBilling element allows you to specify custom billing descriptor information for the transaction. This billing descriptor is used instead of the descriptor defined as the default billing descriptor. If you do not define this element, the system uses the default.

**NOTE:** If you submit a captureGivenAuth transaction with a customBilling element and Worldpay finds a matching Authorization (see Capture Given Auth Transaction on page 72), the system uses the customBilling information from the Authorization and discards the information from the captureGivenAuth.

**Parent Elements:**
authorization, captureGivenAuth, credit, echeckCredit, echeckSale, forceCapture, sale

**IMPORTANT:** Although the schema includes the customBilling element as a child of echeckCredit and echeckSale, Worldpay does not support its use in these instances.

**Attributes:**
None

**Child Elements:**
Required for card-not-present transactions: phone, or url

**NOTE:** Please consult your Relationship Manager prior to using the <url> element. The contents of this element are discarded unless Worldpay specifically enables the use it in your cnpAPI submissions.

Required for card present transactions: city

Optional for either: descriptor

**Example:** customBilling Structure - Card-Not-Present (using phone child)

```xml
<customBilling>
  <phone>Telephone Number</phone>
  <descriptor>Billing Descriptor</descriptor>
</customBilling>
```

**Example:** customBilling Structure - Card-Not-Present (using url child)

```xml
<customBilling>
  <url>retail.url</url>
  <descriptor>www.retail.com</descriptor>
</customBilling>
```
Example: customBilling Structure - Card-Present

    <customBilling>
      <city>City</city>
      <descriptor>Billing Descriptor</descriptor>
    </customBilling>
4.108 customIdentifier

The `customIdentifier` element is an optional child of Sub-merchant funding instruction request transaction types, as well as eCheck Sale/Credit/Redeposit requests. Payment Facilitators/Merchants can use this element to specify a Billing Descriptor to appear on the bank statements of the parties involved in the funds transfer or Direct Debit transaction.

**NOTE:** The information you provide in this element populates the Individual ID field of the ACH Record. The use of this field and its appearance on bank statements is at the discretion of the bank producing the statement.

Type = String; `minLength` = N/A; `maxLength` = 15

Parent Elements:
- `echeckCredit`
- `echeckRedeposit`
- `echeckSale`
- `submerchantCredit`
- `submerchantDebit`

Attributes:
None

Child Elements:
None
4.109 customerInfo

The `customerInfo` element is the parent of several child elements used to define customer information for PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Parent Elements:**
- authorization
- sale

**Attributes:**
- None

**Child Elements:**
- ssn, dob, customerRegistrationDate, customerType, incomeAmount, employerName, customerWorkTelephone, residenceStatus, yearsAtResidence, yearsAtEmployer

**NOTE:** Under certain conditions PayPal Credit requires you to include ssn, dob, customerType and customerRegistrationDate.

**Example: customerInfo Structure**

```
<customerInfo>
    <ssn>ssn</ssn>
    <dob>dob</dob>
    <customerRegistrationDate>customerRegistrationDate</customerRegistrationDate>
    <customerType>customerType</customerType>
    <incomeAmount>incomeAmount</incomeAmount>
    <incomeCurrency>incomeCurrency</incomeCurrency>
    <employerName>employerName</employerName>
    <customerWorkTelephone>customerWorkTelephone</customerWorkTelephone>
    <residenceStatus>residenceStatus</residenceStatus>
    <yearsAtResidence>yearsAtResidence</yearsAtResidence>
    <yearsAtEmployer>yearsAtEmployer</yearsAtEmployer>
</customerInfo>
```
4.110 customerIpAddress

The `customerIpAddress` element is an optional child element of the `cardholderAuthentication` element. This element defines the IP Address of the customer’s system. This element is used either for PayPal Credit transactions or to supply the customer IP Address by merchants enabled for American Express Advanced AVS services.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = Ip Address; **Format** = nnn.nnn.nnn.nnn

**NOTE:** This element is required for BML ecommerce transactions to succeed.

**Parent Elements:**

`cardholderAuthentication`

**Attributes:**

None

**Child Elements:**

None
4.111 customerReference

The `customerReference` element defines a reference string used by the customer for the purchase (for example, a Purchase Order Number). Although the schema defines it as an optional child of the `enhancedData` element, Worldpay recommends you include the element if a value is available/supplied and omit this element if it is blank.

**Type** = String; **minLength** = 1; **maxLength** = 17

**Parent Elements:**
- `enhancedData`

**Attributes:**
- None

**Child Elements:**
- None
4.112 customerRegistrationDate

The `customerRegistrationDate` element is an optional child of the `customerInfo` element defining the earliest date on file with this customer. The latest allowable date is the current date. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = Date; **Format** = YYYY-MM-DD

**NOTE:** In order for a PayPal Credit transaction to succeed, you must include this element if the customer does not have a PayPal Credit account.

**Parent Elements:**
- `customerInfo`

**Attributes:**
- None

**Child Elements:**
- None
4.113 customerType

The `customerType` element is an optional child of the `customerInfo` element defining whether the customer is a new or existing customer. An existing customer is a customer in good standing that has been registered with the merchant for a minimum of 30 days and has made at least one purchase in the last 30 days. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = Choice (Enum); **Enumerations** = New or Existing

**NOTE:** In order for a PayPal Credit transaction to succeed, you must include this element if the customer does not have a PayPal Credit account.

**Parent Elements:**

- `customerInfo`

**Attributes:**

- None

**Child Elements:**

- None
4.114 customerWorkTelephone

The `customerWorkTelephone` element is an optional child of the `customerInfo` element and defines the customer's work telephone number. It is used in combination with several other elements to provide information for some PayPal Credit transactions.

NOTE: As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
- customerInfo

**Attributes:**
- None

**Child Elements:**
- None
4.115 data

The `data` element is a required child of the `applepay` element. It is the payment data dictionary, BASE64 encoded string from the PKPaymentToken.

**Type** = String; **minLength** = N/A; **maxLength** = 2000

**Parent Elements:**
- applepay

**Attributes:**
- None

**Child Elements:**
- None
4.116 deactivate

The deactivate element is the parent element for the transaction type that deactivate a Gift Card.

Parent Elements:

litleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)

orderid, orderSource, card
4.117 deactivateResponse

The `deactivateResponse` element is the parent element for information returned to you in response to a `deactivate` transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**
- `litleOnlineResponse`
- `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deactivate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Deactivate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deactivate transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`

Optional: `postDate`, `fraudResult`, `giftCardResponse`
4.118 deactivateReversal

The `deactivateReversal` element is the parent element for the transaction type that reverses the deactivation of a Gift Card.

**Parent Elements:**
- litleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:** (Required)
- card
- originalRefCode
- originalAmount
- originalTxnTime
- originalSystemTraceId
- originalSequenceNumber

**Child Elements:** (Optional)
- litleTxnId
4.119 deactivateReversalResponse

The `deactivateReversalResponse` element is the parent element for information returned to you in response to an `deactivateReversal` transaction.

**Parent Elements:**
- littleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deactivate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Deactivate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deactivate Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**
- Required: `littleTxnId`, `response`, `responseTime`, `message`
- Optional: `postDate`, `giftCardResponse`
4.120 debtRepayment

The `debtRepayment` element is an optional child of `authorization`, `captureGivenAuth`, `forceCapture`, and `sale` transactions. You use this flag only if the method of payment is Visa and the transaction is for a debt repayment. Also, your MCC must be either 6012 or 6051. If you set this flag to true and you are not one of the allowed MCCs, the system declines the transaction with a Response Reason Code of 852 - Debt Repayment only allowed for VI transactions on MCCs 6012 and 6051.

**Type** = Boolean; **Valid Values** = true or false (default)

**Parent Elements:**
- `authorization`
- `captureGivenAuth`
- `forceCapture`
- `sale`

**Attributes:**
None

**Child Elements:**
None
4.121 deleteAddOn

The `deleteAddOn` element is the parent element used to define an Add On to be removed from an existing subscription.

**Parent Elements:**
`updateSubscription`

**Attributes:**
None

**Child Elements (all Required):**
`addOnCode`

**Example: deleteAddOn Structure**

```
<deleteAddOn>
</deleteAddOn>
```
4.122 deleteDiscount

The `deleteDiscount` element is the parent element used to define a discount to be removed from an existing subscription.

**Parent Elements:**
updateSubscription

**Attributes:**
None

**Child Elements (all Required):**
discountCode

**Example: deleteDiscount Structure**

```xml
<deleteDiscount>
  <discountCode>Discount Reference Code</discountCode>
</deleteDiscount>
```
4.123 deliveryType

The `deliveryType` element is an optional child of the `enhancedData` element and defines the shipping method used for delivery of the product.

Type = String (enum); `minLength` = N/A; `maxLength` = N/A

**Parent Elements:**
- `enhancedData`

**Attributes:**
- None

**Child Elements:**
- None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNC</td>
<td>Cash and Carry</td>
</tr>
<tr>
<td>DIG</td>
<td>Digital Delivery</td>
</tr>
<tr>
<td>PHY</td>
<td>Physical Delivery</td>
</tr>
<tr>
<td>SVC</td>
<td>Service Delivery</td>
</tr>
<tr>
<td>TBD (default)</td>
<td>To be determined.</td>
</tr>
</tbody>
</table>

4.124 dentalAmount

The `dentalAmount` element is an optional child of the `healthcareAmounts` element and defines the healthcare amount used for dental related purchases. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**
Optional: `healthcareAmounts`

**Attributes:**
None

**Child Elements:**
None
4.125 depositReversal

The depositReversal element is the parent element for a Gift Card specific transaction type that reverses a giftCardCapture or sale transaction.

Parent Elements:

litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (Required)

- card, originalRefCode, originalAmount, originalTxnTime, originalSystemTraceId, originalSequenceNumber

Child Elements: (Optional)

- littleTxnId
4.126 depositReversalResponse

The `depositReversalResponse` element is the parent element for information returned to you in response to an `depositReversal` transaction.

Parent Elements:
- `litleOnlineResponse`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deposit Reversal transaction.</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Deposit Reversal transaction.</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Deposit Reversal transaction.</td>
</tr>
</tbody>
</table>

Child Elements:
- Required: `litleTxnId`, `response`, `responseTime`, `message`, `giftCardResponse`
- Optional: `postDate`
4.127 description

The `description` element is an optional child of the `createPlan` element and provides a text description of the Plan.

**Type** = String; **minLength** = N/A; **maxLength** = 100

**Parent Elements:**
- `createPlan`

**Attributes:**
- None

**Child Elements:**
- None
4.128 descriptor

The descriptor element is an optional child of the customBilling element. This element defines the text you wish to display on the customer bill, enabling the customer to better recognize the charge.

**Type** = String; **minLength** = 4; **maxLength** = 25

**NOTE:** If you include a prefix:

- the prefix must be either 3, 7, or 12 characters in length.
- you must use an asterisk (*) after the prefix as a separator, in one of the following positions: 4th, 8th, or 13th. Do not use an asterisk in more than one position.
- Use only the following valid characters:
  - Numbers
  - Letters
  - Special characters as follows: ampersand, asterisk (Required; see note above), comma, dash, period, or pound sign.

**Parent Elements:**

customBilling

**Attributes:**

None

**Child Elements:**

None

**NOTE:** When using a descriptor with Direct Debits, Worldpay maps the first 16 characters to the Company Name field of the ACH message and the last nine characters as the company phone number.
4.129 destinationCountryCode

The `destinationCountryCode` element defines the country portion of the postal mailing address in the `enhancedData` element.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = 3

**NOTE:** The enumerations for this element are listed under `<countryTypeEnum>` in the cnpAPI Common XSD. The country names corresponding to the abbreviations can be found in the ISO 3166-1 standard.

**Parent Elements:**
`enhancedData`

**Attributes:**
None

**Child Elements:**
None
4.130 `destinationPostalCode`

The `destinationPostalCode` element defines the postal code of the destination in the `enhancedData` element.

**Type** = String; **minLength** = N/A; **maxLength** = 20

| NOTE: Although the schema specifies the maxLength of the `<destinationPostalCode>` element as 20 characters, in practice you should never exceed 10 characters in your submissions. |

**Parent Elements:**
- `enhancedData`

**Attributes:**
- None

**Child Elements:**
- None
4.131 detailTax

The `detailTax` element is an optional child of both the `enhancedData` and `lineItemData` elements, which you use to specify detailed tax information (for example, city or local tax). The total sum of the `detailTax` values should match either the `salesTax` value, if `detailTax` is a child of `enhancedData`, or the `taxAmount` element if `detailTax` is a child of `lineItemData`.

**Parent Elements:**
`enhancedData`, `lineItemData`

**Attributes:**
None

**Child Elements:**
Required: `taxAmount`
Optional: `taxIncludedInTotal`, `taxRate`, `taxTypeIdentifier`, `cardAcceptorTaxId`

**Example: detailTax Structure**

```xml
<detailTax>
  <taxIncludedInTotal>true or false</taxIncludedInTotal>
  <taxAmount>Additional Tax Amount</taxAmount>
  <taxRate>Tax Rate of This Tax Amount</taxRate>
  <taxTypeIdentifier>Tax Type Enum</taxTypeIdentifier>
  <cardAcceptorTaxId>Tax ID of Card Acceptor</cardAcceptorTaxId>
</detailTax>
```
4.132 deviceManufacturerIdentifier

The `deviceManufacturerIdentifier` element is an optional child of the `applepayResponse` element and defines the manufacturer of the device originating the transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
- `applepayResponse`

**Attributes:**
- None

**Child Elements:**
- None
**4.133 deviceReputationScore**

The `deviceReputationScore` element is an optional child of the `advancedFraudResults` element and specifies score resulting from the ThreatMetrix analysis of the customer device.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**

`advancedFraudResults`

**Attributes:**

None

**Child Elements:**

None
4.134 deviceReviewStatus

The `deviceReviewStatus` element is a required child of the `advancedFraudResults` element and specifies the results of the comparison of the deviceReputationScore against the threshold levels configured for the merchant. Typical values are: pass, fail, review, unavailable, and invalid_session.

**NOTE:** The system returns `invalid_session` if you submitted a session Id using an invalid prefix.

Type = String; minLength = N/A; maxLength =

Parent Elements:

`advancedFraudResults`

Attributes:

None

Child Elements:

None
4.135 discountAmount

The `discountAmount` element defines the amount of the discount for the order. Although the schema defines it as an optional child of the `enhancedData` element, it is required by Visa for Level III interchange rates. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**
- `enhancedData`

**Attributes:**
- None

**Child Elements:**
- None
4.136 discountCode

The discountCode element is the identifier of a defined discount. You use this element when creating, updating, and deleting a discount applied to a subscription.

Type = String; minLength = N/A; maxLength = 25

Parent Elements:
createDiscount, updateDiscount, deleteDiscount

Attributes:
None

Child Elements:
None
4.137 dob

The `dob` element is an optional child of the `customerInfo` element. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = Date; Format = YYYY-MM-DD

**NOTE:** In order for a PayPal Credit transaction to succeed, you must include this element if:

- the customer does not have a PayPal Credit account

or

- the customer has a PayPal Credit account, but the account has not been authenticated.

You do not need to include this element if you authenticated the PayPal Credit account.

**Parent Elements:**
- `customerInfo`

**Attributes:**
- None

**Child Elements:**
- None
**4.138 dutyAmount**

The `dutyAmount` element defines duty on the total purchased amount for the order. Although the schema defines it as an optional child of the `enhancedData` element, it is required by Visa for Level III interchange rates. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**

`enhancedData`

**Attributes:**

None

**Child Elements:**

None
4.139 echeck

The echeck element is a required child of the echeckSale, echeckVerification, and echeckCredit (when the credit is against a transaction not originally processed through our system) elements. It contains child elements used to provide details concerning the Direct Debit account.

Parent Elements:
echeckCredit, echeckPreNoteCredit, echeckPreNoteSale, echeckSale, echeckVerification

Attributes:
None

Child Elements:
Required: accType, accNum, routingNum
Optional: checkNum, ccdPaymentInformation

Example: echeck Structure

```xml
<echeck>
    <accType>Account Type Abbreviation</accType>
    <accNum>Account Number</accNum>
    <routingNum>Routing Number</routingNum>
    <checkNum>Check Number</checkNum>
    <ccdPaymentInformation>Payment Description</ccdPaymentInformation>
</echeck>
```
4.140 eCheckAccountSuffix

The eCheckAccountSuffix element is an optional child of the tokenResponse element that provides the last three characters of the Direct Debit account number.

Type = String; minLength = 3; maxLength = 3

Parent Elements:
registerTokenResponse, tokenResponse

Attributes:
None

Child Elements:
None
4.141 echeckCredit

The echeckCredit element is the parent element for all eCheck Credit transactions. You can use this element in either Batch or Online transactions.

**IMPORTANT:** Although the schema includes the customBilling element as a child of echeckCredit and echeckSale, Worldpay does not support its use in these instances.

**Parent Elements:**
- batchRequest
- litleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1    maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A    maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1    maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

For credits to transactions processed by Worldpay (Required): litleTxnId

For credits to transactions processed by Worldpay (Optional): amount

**NOTE:** Omit the amount child element, to use the full amount from the echeckSale.

For credits to transactions not processed by Worldpay (Required): orderId, amount, orderSource, billToAddress, echeckOrEcheckToken (allows the substitution of either the echeck or echeckToken elements)

For credits to transactions not processed by Worldpay (Optional): merchantData

For both (Optional): customBilling, secondaryAmount, customIdentifier
4.142 echeckCreditResponse

The echeckCreditResponse element is the parent element for information returned to you in response to an echeckCredit transaction.

Parent Elements:
batchResponse, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the echeckCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)
litleTxnId, response, responseTime, message

Optional: postDate, accountUpdater

**NOTE:** The postDate child element is returned only in responses to Online transactions.
4.143 echeckForToken

The `echeckForToken` element is a child of the `registerTokenRequest` element. It contains the routing and account number of the eCheck account which the system uses to generate a token.

**Parent Elements:**
`registerTokenRequest`

**Attributes:**
None

**Child Elements:**
Required: `accNum`, `routingNum`

**Example: echeck Structure**

```xml
<echeckForToken>
    <accNum>Account Number</accNum>
    <routingNum>Routing Number</routingNum>
</echeckForToken>
```
4.144 echeckOrEcheckToken

The `echeckOrEcheckToken` element is an abstract that allows the substitution of either the `echeck` or `echeckToken` element. In eCheck transactions, except echeckVoid, you must specify one of the two substitution elements as a child.

**Parent Elements:**

echeckCredit, echeckRedeposit, echeckSale, echeckVerification

**Substitution Options:**
echeck, echeckToken
4.145 echeckPreNoteCredit

The echeckPreNoteCredit element is the parent element for all eCheck Prenotation Credit transactions. You use this transaction type to perform an eCheck Prenotation, when the subsequent eCheck transaction will be an eCheck Credit transaction. You can use this element only in Batch transactions. This transaction type is only supported for US transactions.

Parent Elements:
batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: orderld, orderSource, billToAddress, echeck
Optional: merchantData
4.146 echeckPreNoteCreditResponse

The echeckPreNoteCreditResponse element is the parent element for information returned to you in response to an echeckPreNoteCredit transaction.

Parent Elements:

batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckPreNoteCredit transaction. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the echeckPreNoteCredit transaction. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckPreNoteCredit transaction. minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)

litleTxnId, response, responseTime, message
4.147 echeckPreNoteSale

The echeckPreNoteSale element is the parent element for all eCheck Prenotification Sale transactions. You use this transaction type to perform an eCheck Prenotification, when the subsequent eCheck transaction will be an eCheck Sale transaction. You can use this element only in Batch transactions. This transaction type is only supported for US transactions.

Parent Elements:
batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: orderId, orderSource, billToAddress, echeck
Optional: merchantData
4.148 echeckPreNoteSaleResponse

The `echeckPreNoteSaleResponse` element is the parent element for information returned to you in response to an `echeckPreNoteSale` transaction.

**Parent Elements:**
- `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>echeckPreNoteSale</code> transaction. <code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the <code>echeckPreNoteSale</code> transaction. <code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>echeckPreNoteSale</code> transaction. <code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:** (all Required)
- `litleTxnId`, `response`, `responseTime`, `message`
4.149 echeckRedeposit

The echeckRedeposit element is the parent element for all eCheck Redeposit transactions. You use this transaction type to manually attempt redeposits of eChecks returned for either Insufficient Funds or Uncollected Funds. You can use this element in either Batch or Online transactions.

NOTE: Do not use this transaction type if you are enabled for the Auto Redeposit feature. If you are enabled for the Auto Redeposit feature, the system will reject any echeckRedeposit transaction you submit.

Parent Elements:
batchRequest, littleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: littleTxnId
Optional: echeckOrEcheckToken (allows the substitution of either the echeck or echeckToken elements), merchantData, customIdentifier
4.150 echeckRedepositResponse

The echeckRedepositResponse element is the parent element for information returned to you in response to an echeckRedeposit transaction.

Parent Elements:
batchResponse, littleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckSale transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the echeckSale transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckSale transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: littleTxnId, response, responseTime, message
Optional: postDate, accountUpdater

NOTE: The system returns the postDate child element only in responses to Online transactions.
4.151 echeckSale

The echeckSale element is the parent element for all eCheck Sale transactions. You can use this element in either Batch or Online transactions.

**Parent Elements:**
batchRequest, littleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**
Required: orderId, amount, orderSource, billToAddress, echeckOrEcheckToken (allows the substitution of either the echeck or echeckToken elements)

**NOTE:** You must use one of the following values for the orderSource element: telephone, ecommerce, recurringtel, or echeckppd.

Optional: shipToAddress, verify, customBilling, merchantData, secondaryAmount, customIdentifier

**IMPORTANT:** Although the schema includes the customBilling element as a child of echeckCredit and echeckSale, Worldpay does not support its use in these instances.
4.152 echeckSalesResponse

The `echeckSalesResponse` element is the parent element for information returned to you in response to an echeckSale transaction.

**Parent Elements:**
- batchResponse
- littleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckSale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the echeckSale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the echeckSale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: littleTxnId, response, responseTime, message

Optional: postDate, accountUpdater, tokenResponse

**NOTE:** The system returns the postDate child element only in responses to Online transactions.
4.153 echeckToken

The echeckToken element replaces the echeck element in tokenized eCheck transactions and defines the tokenized account information.

Parent Elements:
echeckCredit, echeckRedeposit, echeckSale, echeckVerification

Attributes:
None

Child Elements:
Required: litleToken, routingNum, accType
Optional: checkNum

Example: echeck Structure

```
<echeckToken>
  <litleToken>Token</litleToken>
  <routingNum>Routing Number</routingNum>
  <accType>Account Type Abbreviation</accType>
  <checkNum>Check Number</checkNum>
</echeckToken>
```
4.154 echeckVerification

The `echeckVerification` element is the parent element for all eCheck Verification transactions. You use this transaction type to initiate a comparison of the consumer’s account information against positive/negative databases. You can use this element in either Batch or Online transactions. This transaction type is only supported for US transactions.

**Parent Elements:**

`batchRequest`, `litleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1     maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A    maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1     maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `orderId`, `amount`, `orderSource`, `billToAddress`, `echeckOrEcheckToken` (allows the substitution of either the `echeck` or `echeckToken` elements)

Optional: `merchantData`
4.155 echeckVerificationResponse

The `echeckVerificationResponse` element is the parent element for information returned to you in response to a eCheck Verification transaction.

Parent Elements:
- batchResponse, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the capture transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
- Required: `litleTxnId`, `response`, `responseTime`, `message`
- Optional: `postDate`

NOTE: The system returns the postDate child element only in responses to Online transactions.
4.156 echeckVoid

The echeckVoid element is the parent element for all eCheck Void transactions. You use this transaction type to either cancel an eCheck Sale transaction, as long as the transaction has not yet settled, or halt automatic redeposit attempts of eChecks returned for either Insufficient Funds or Uncollected Funds. You can use this element only in Online transactions.

Parent Elements:
litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: litleTxnId
4.157 echeckVoidResponse

The echeckVoidResponse element is the parent element for information returned to you in response to an eCheck Void transaction.

**Parent Elements:**
- litleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the void transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the void transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the void transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements: (all Required)**
- litleTxnId, response, responseTime, message, postDate
4.158 eciIndicator

The eciIndicator element is an optional child of the applepayResponse and androidpayResponse elements and specifies electronic commerce indicator associated with an Apple Pay/Google Pay transaction.

Type = String; minLength = N/A; maxLength = 2

Parent Elements:
applepayResponse, androidpayResponse

Attributes:
None

Child Elements:
None
4.159 email

The email element defines the email address of the customer in both the billToAddress and shipToAddress elements.

**NOTE:** The email element is required for sale transactions using the sepaDirectDebit method of payment. Failure to include the name element results in a 905 - Missing Email Address Response Code.

Type = String; minLength = N/A; maxLength = 100

Parent Elements:

billToAddress, shipToAddress

Attributes:

None

Child Elements:

None
4.160 employerName

The employerName element is an optional child of the customerInfo element and defines the name of the customer's place of employment. It is used in combination with several other elements to provide information for some PayPal Credit transactions.

NOTE: As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = String; minLength = N/A; maxLength = 20

Parent Elements:

customerInfo

Attributes:

None

Child Elements:
4.161 encryptedTrack

The encryptedTrack element is a required child of the mpos element. This element provides the encrypted track data resulting from a card swipe using a ROAM device.

Type = String; minLength = 1; maxLength = 1028

Parent Elements:

mpos

Attributes:

None

Child Elements:

None
4.162 endDate

The `endDate` element is an optional child of both the `createAddOn` and `createDiscount` element, where it specifies either the ending date of the Add On charge or the ending date of the discount.

**Type** = Date; **Format** = YYYY-MM-DD

**Parent Elements:**
createAddOn, createDiscount

**Attributes:**
None

**Child Elements:**
None
4.163 endingBalance

The endingBalance element is an optional child of the giftCardResponse element. It defines the available balance on the submitted Gift Card after the requested operation.

**NOTE:** Although included in the schema, the beginningBalance, endingBalance, and cashBackAmount elements are not currently supported.

*Type = String; minLength = N/A; maxLength = 20*

**Parent Elements:**
giftCardResponse

**Attributes:**
None

**Child Elements:**
None
4.164 endpoint

The endpoint element is a required child of the networkResponse element. It defines the card network (i.e., Visa, MasterCard, Discover, or American Express) acting as an endpoint for the submitted transaction.

Type = String; minLength = N/A; maxLength = 256

Parent Elements:

networkResponse

Attributes:

None

Child Elements:

None
4.165 enhancedAuthResponse

The enhancedAuthResponse element is an optional child of both the authorizationResponse and saleResponse elements. Through its child elements, it can provide information concerning whether the card used for the transaction is Prepaid and if so, the available balance. Depending upon the card used, other elements can indicate affluence, card product type, prepaid card type, reloadability and whether or not it is a virtual account number.

Parent Elements:
authorizationResponse, saleResponse

Attributes:
None

Child Elements: (all Optional)
fundingSource, affluence, issuerCountry, cardProductType, virtualAccountNumber, networkResponse

Example: enhancedAuthResponse - with fundingSource

```xml
<enhancedAuthResponse>
  <fundingSource>
    <type>PREPAID</type>
    <availableBalance>0</availableBalance>
    <reloadable>YES|NO|UNKNOWN</reloadable>
    <prepaidCardType>GIFT</prepaidCardType>
  </fundingSource>
</enhancedAuthResponse>
```

Example: enhancedAuthResponse - with affluence

```xml
<enhancedAuthResponse>
  <affluence>AFFLUENT</affluence>
</enhancedAuthResponse>
```

Example: enhancedAuthResponse - with issuerCountry

```xml
<enhancedAuthResponse>
  <issuerCountry>MEX</issuerCountry>
</enhancedAuthResponse>
```

Example: enhancedAuthResponse - with cardProductType

```xml
<enhancedAuthResponse>
  <cardProductType>CONSUMER</cardProductType>
</enhancedAuthResponse>
```
Example: enhancedAuthResponse - with virtualAccountNumber

```
<enhancedAuthResponse>
  <virtualAccountNumber>true or false</virtualAccountNumber>
</enhancedAuthResponse>
```

Example: enhancedAuthResponse - with networkResponse

```
<enhancedAuthResponse>
  <networkResponse>
    <endpoint>VISA</endpoint>
    <networkField fieldNumber="4" fieldName="Transaction Amount">
      <fieldValue>00000000962</fieldValue>
    </networkField>
    <networkField fieldNumber="44" fieldName="Additional Response Data">
      <networkSubField fieldNumber="0">
        <fieldValue></fieldValue>
      </networkSubField>
      <networkSubField fieldNumber="1">
        <fieldValue>5</fieldValue>
      </networkSubField>
    </networkField>
  </networkResponse>
</enhancedAuthResponse>
```
4.166 enhancedData

The enhancedData element allows you to specify extra information concerning a transaction in order to qualify for certain purchasing interchange rates. The following tables provide information about required elements you must submit to achieve Level 2 or Level 3 Interchange rates for Visa and MasterCard.

- For MasterCard:
  - The transaction must be taxable.
  - The tax charged must be between 0.1% and 30% of the transaction amount, except as noted below.
  - For Level 3, the transaction must use a corporate, business, or purchasing card.

  **NOTE:** You can qualify for MasterCard Level 2 rates without submitting the total tax amount (submit 0) if your MCC is one of the following: 4111, 4131, 4215, 4784, 8211, 8220, 8398, 8661, 9211, 9222, 9311, 9399, 9402.

  This also applies to Level 3, regardless of your MCC (i.e., submission of 0 allowed).

- You must include at least one line item with amount, description, and quantity defined.

**TABLE 4-1** MasterCard Level 2/Level 3 Data Requirements

<table>
<thead>
<tr>
<th>MasterCard Level 2 Data</th>
<th>MasterCard Level 3 Data</th>
<th>cnpAPI Element (child of enhancedData unless noted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer Code (if supplied by customer)</td>
<td>Customer Code (if supplied by customer)</td>
<td>customerReference</td>
</tr>
<tr>
<td>Card Acceptor Tax ID</td>
<td>Card Acceptor Tax ID</td>
<td>cardAcceptorTaxId (child of detailTax)</td>
</tr>
<tr>
<td>Total Tax Amount</td>
<td>Total Tax Amount</td>
<td>salesTax</td>
</tr>
<tr>
<td>Product Code</td>
<td></td>
<td>productCode (child of lineItemData)</td>
</tr>
<tr>
<td>Item Description</td>
<td></td>
<td>itemDescription (child of lineItemData)</td>
</tr>
<tr>
<td>Item Quantity</td>
<td></td>
<td>quantity (child of lineItemData)</td>
</tr>
<tr>
<td>Item Unit of Measure</td>
<td></td>
<td>unitOfMeasure (child of lineItemData)</td>
</tr>
<tr>
<td>Extended Item Amount</td>
<td></td>
<td>lineItemTotal (child of lineItemData) or lineItemTotalWithTax (child of lineItemData)</td>
</tr>
</tbody>
</table>

In addition to the requirements below, please be aware of the following:
For Visa:

- The transaction must be taxable.
- The tax charged must be between 0.1% and 22% of the transaction amount, except as noted.
- For Level 3, the transaction must use a corporate or purchasing card.

**NOTE:** Sales Tax is no longer required for Level 3. In this case omit the `<salesTax>` element entirely.

### TABLE 4-2  Visa Level 2/Level 3 Data Requirements

<table>
<thead>
<tr>
<th>Visa Level 2 Data</th>
<th>Visa Level 3 Data</th>
<th>cnpAPI Element (child of enhancedData unless noted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales Tax</td>
<td>Sales Tax*</td>
<td>salesTax</td>
</tr>
<tr>
<td>Discount Amount</td>
<td>discountAmount</td>
<td></td>
</tr>
<tr>
<td>Freight/Shipping Amount</td>
<td>shippingAmount</td>
<td></td>
</tr>
<tr>
<td>Duty Amount</td>
<td>dutyAmount</td>
<td></td>
</tr>
<tr>
<td>Item Sequence Number</td>
<td>itemSequenceNumber (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Item Commodity Code</td>
<td>commodityCode (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Item Description</td>
<td>itemDescription (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Product Code</td>
<td>productCode (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Quantity</td>
<td>quantity (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Unit of Measure</td>
<td>unitOfMeasure (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>unit Cost</td>
<td>unitCost (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Discount per Line Item</td>
<td>itemDiscountAmount (child of lineItemData)</td>
<td></td>
</tr>
<tr>
<td>Line Item Total</td>
<td>lineItemTotal (child of lineItemData)</td>
<td></td>
</tr>
</tbody>
</table>

* Sales Tax is no longer required for Level 3. In this case omit the `<salesTax>` element entirely.

**NOTE:** For Discover transactions, the interchange rate is not impacted by the presence of Level 2 or Level 3 data.

**Parent Elements:**

- authorization, capture, captureGivenAuth, credit, forceCapture, sale
Child Elements: (all Optional)

- customerReference
- salesTax
- deliveryType
- taxExempt
- discountAmount
- shippingAmount
- dutyAmount
- shipFromPostalCode
- destinationPostalCode
- destinationCountryCode
- invoiceReferenceNumber
- orderDate
- detailTax
- lineItemData

Example: enhancedData Structure

```xml
<enhancedData>
  <customerReference>Customer Reference</customerReference>
  <salesTax>Amount of Sales Tax Included in Transaction</salesTax>
  <deliveryType>TBD</deliveryType>
  <taxExempt>true or false</taxExempt>
  <discountAmount>Discount Amount Applied to Order</discountAmount>
  <shippingAmount>Amount to Transport Order</shippingAmount>
  <dutyAmount>Duty on Total Purchase Amount</dutyAmount>
  <shipFromPostalCode>Ship From Postal Code</shipFromPostalCode>
  <destinationPostalCode>Ship To Postal Code</destinationPostalCode>
  <invoiceReferenceNumber>Merchant Invoice Number</invoiceReferenceNumber>
  <orderDate>Date Order Placed</orderDate>
  <detailTax>
    <taxIncludedInTotal>true or false</taxIncludedInTotal>
    <taxAmount>Additional Tax Amount</taxAmount>
    <taxRate>Tax Rate of This Tax Amount</taxRate>
    <taxTypeIdentifier>Tax Type Enum</taxTypeIdentifier>
    <cardAcceptorTaxId>Tax ID of Card Acceptor</cardAcceptorTaxId>
  </detailTax>
  <lineItemData>
    <itemSequenceNumber>Line Item Number within Order</itemSequenceNumber>
    <itemDescription>Description of Item</itemDescription>
    <quantity>Quantity of Item</quantity>
    <unitOfMeasure>Unit of Measurement Code</unitOfMeasure>
    <taxAmount>Sales Tax or VAT of Item</taxAmount>
    <lineItemTotal>Total Amount of Line Item</lineItemTotal>
    <lineItemTotalWithTax>taxAmount + lineItemTotal</lineItemTotalWithTax>
    <itemDiscountAmount>Discount Amount</itemDiscountAmount>
    <commodityCode>Card Acceptor Commodity Code for Item</commodityCode>
  </lineItemData>
</enhancedData>
```
<unitCost>Price for One Unit of Item</unitCost>
</lineItemData>
</enhancedData>
4.167 entryMode

The `entryMode` element is a required child of the `pos` element, which describes the method used for card data entry at the point of sale.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
- `pos`

**Attributes:**
- None

**Child Elements:**
- None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>notused</td>
<td>terminal not used</td>
</tr>
<tr>
<td>keyed</td>
<td>card number manually entered</td>
</tr>
<tr>
<td>track1</td>
<td>track 1 read</td>
</tr>
<tr>
<td>track2</td>
<td>magnetic stripe read (track 2 when known or when the terminal makes no distinction between tracks 1 and 2.)</td>
</tr>
<tr>
<td>completread</td>
<td>complete magnetic stripe read and transmitted</td>
</tr>
</tbody>
</table>
4.168 ephemeralPublicKey

The `ephemeralPublicKey` element is a required child of the `header` element and provides the BASE64 Encoded string of the ephemeral public key bytes from the Apple Pay transaction.

Type = Base 64 Encoded String; minLength = N/A; maxLength = 400

Parent Elements:

- header

Attributes:
None

Child Elements:
None
4.169 expDate

The `expDate` element is a child of the `card`, `token`, `paypage` elements, which specifies the expiration date of the card (format: mmyy) and is required for card-not-present transactions.

**NOTE:** Although the schema defines the `expDate` element as an optional child of the `card`, `token` and `paypage` elements, you must submit a value for card-not-present transactions.

Type = String; minLength = 4; maxLength = 4

Parent Elements:

- card, newCardInfo, newCardTokenInfo, originalCard, originalCardInfo, originalCardTokenInfo, originalToken, paypage, token, updatedCard, updatedToken

Attributes:

None

Child Elements:

None

**NOTE:** You should submit whatever expiration date you have on file, regardless of whether or not it is expired/stale.

We recommend all merchant with recurring and/or installment payments participate in the Account Updater program.
4.170 expMonth

The `expMonth` element is an optional child of the `androidpayResponse` element, which specifies the month of expiration of the network token (format: mm).

**Type** = String; **minLength** = 2; **maxLength** = 2

**Parent Elements:**
- `androidpayResponse`

**Attributes:**
- None

**Child Elements:**
- None
4.171 expYear

The `expYear` element is an optional child of the `androidpayResponse` element, which specifies the year of expiration of the network token (format: yyyy).

**Type** = String; **minLength** = 4; **maxLength** = 4

**Parent Elements:**
androidpayResponse

**Attributes:**
None

**Child Elements:**
None
4.172 extendedCardResponse

The `extendedCardResponse` element is an optional child of the `accountUpdater` element, which contains two child elements, `code` and `message`. The codes/messages can be any of the following:

- 500 - The account number was changed
- 501 - The Account Was Closed
- 502 - The expiration date was changed
- 504 - Contact the cardholder for updated information
- 507 - Cardholder has opted out of the update program

**NOTE:** When using Account Updater (any variation), you must always code to receive the `extendedCardResponse` element and its children. We always return this information whenever applicable regardless of whether you receive other account updater information in the transaction response message.

**Parent Elements:**

- `accountUpdater`

**Attributes:**

None

**Child Elements:**

- `code`
- `message`

**Example: newCardInfo Structure**

```xml
<extendedCardResponse>
  <message>Message for Code</message>
  <code>Either 500, 501, 502, 504, or 507</code>
</extendedCardResponse>
```
4.173 fastAccessFunding

The **fastAccessFunding** element is the parent element for the transaction type that a Payment Facilitator uses to move funds to a sub-merchant held debit card. The transfer of funds occurs within 30 minutes.

**NOTE:** For additional information about PayFac Instruction-Based Dynamic Payout and the use of this transaction type, please refer to the *PayFac Instruction-Based Dynamic Payout Info* document.

**Parent Elements:**
LittleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: amount, fundingSubmerchantId, fundsTransferId, submerchantName, (choice of) card, paypage, or token
4.174 fastAccessFundingResponse

The `fastAccessFundingResponse` element is the parent element for information returned to you in response to a `fastAccessFunding` transaction.

**Parent Elements:**
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the submerchantCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to <code>true</code> and the entire original response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId, fundsTransferId, response, responseTime, message, postDate`
4.175 fieldValue

The fieldValue element is a child of both the networkField and networkSubField elements. The fieldValue element is required when a child of the networkField element and no subfield exists. This element is always required when used as a child of the networkSubField element. It provides the raw data for the designated field, extracted from the network ISO 8583 response message.

Type = String; minLength = N/A; maxLength = 9999

Parent Elements:
networkField, networkSubField

Attributes:
None

Child Elements:
None
4.176 filtering

The filtering element is an optional child of either the Authorization or Sale request transaction. You use its child elements to selectively enable/disable the various Basic Fraud toolkit features. Setting either the <international> or <chargeback> child element to false disables that filtering feature for the transaction. The <prepaid> child can be set to true to enable the feature selectively, or set to false to disable the feature for the transaction, if you elected to use the filter all prepaid configuration option.

Parent Elements: 
authorization, sale

Attributes: None

Child Elements: 
Optional: prepaid, international, chargeback

NOTE: Although included in the schema and shown in the example below, the <chargeback> element is not supported. To override the chargeback filter for a selected transaction, use the fraudFilterOverride flag (see fraudFilterOverride on page 513). Please consult your Relationship Manager for additional information.

Example: filtering Structure

<filtering>
  <prepaid>true or false</prepaid>
  <international>false</international>
  <chargeback>false</chargeback>
</filtering>
4.177 finalPayment

The finalPayment element is a required child of the littleInternalRecurringRequestType. A value of true indicates that this is the final payment of the subscription. Since this element is used only in internally generated transaction, you do not need to code for its use.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
littleInternalRecurringRequest

**Attributes:**
None

**Child Elements:**
None
4.178 firstName

The firstName element is a child of the billToAddress element, which specifies the first name of the account holder and is required for echeckVerification transactions.

**NOTE:** When performing an eCheck Verification for a corporate account, you must include values for the **firstName** and **lastName** elements. If you do not have the name of the check issuer, you can use a value of “unavailable” for both elements.

Type = String; minLength = N/A; maxLength = 25

Parent Elements:
- billToAddress

Attributes:
- None

Child Elements:
- None
4.179 forceCapture

The forceCapture element is the parent element for all Force Capture transactions. These are specialized Capture transactions used when you do not have a valid Authorization for the order, but have fulfilled the order and wish to transfer funds. You can use this element in either Online or Batch transactions.

**NOTE:** You must be authorized by Worldpay before processing this transaction type. In some instances, using a Force Capture transaction can lead to chargebacks and fines.

Parent Elements:
liteOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: orderId, amount, orderSource, choice of card, token, mpos, paypage, or applepay

Optional: billToAddress, customBilling, taxType, enhancedData, processingInstructions, pos, amexAggregatorData, merchantData, productCode, secondaryAmount, surchargeAmount, debtRepayment, processingType
4.180 forceCaptureResponse

The forceCaptureResponse element is the parent element for information returned to you in response to a Force Capture transaction. It can be a child of either a litleOnlineResponse element or a batchResponse element.

**Parent Elements:**
litleOnlineResponse, batchResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String | Yes       | The response returns the same value submitted in the Force Capture transaction. 
|                |        |           | minLength = 1  maxLength = 36 |
| customerId     | String | No        | The response returns the same value submitted in the Force Capture transaction. 
|                |        |           | minLength = N/A  maxLength = 50 |
| reportGroup    | String | Yes       | The response returns the same value submitted in the Force Capture transaction. 
|                |        |           | minLength = 1  maxLength = 25 |

**Child Elements:**

Required: litleTxnId, response, responseTime, message

Optional: postDate, tokenResponse, accountUpdater, giftCardResponse, applepayResponse

**NOTE:** The system returns the postDate child element only in responses to Online transactions.
4.181 formatId

The `formatId` element is a required child of the `mpos` element. This element identifies the format of the encrypted track submitted from the device.

Type = String; minLength = 1; maxLength = 1028

Parent Elements:

- mpos

Attributes:

None

Child Elements:

None
4.182 fraudCheck

The `fraudCheck` element is the parent element for the standalone Advanced Fraud Check transaction. You use this transaction type to retrieve the results of the Advanced Fraud Check tool without submitting an Authorization or Sale transaction. You can use this element only in Online transactions.

**Parent Elements:**
littleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

- Required: advancedFraudChecks
- Optional: billToAddress, shipToAddress, amount
4.183 fraudCheckResponse

The `fraudCheckResponse` element is the parent element for information returned to you in response to a standalone Fraud Check transaction.

**Parent Elements:**
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String  | Yes       | The response returns the same value submitted in the Force Capture transaction. 
|                |         |           | `minLength = 1`  
|                |         |           | `maxLength = 36` |
| customerId     | String  | No        | The response returns the same value submitted in the Force Capture transaction. 
|                |         |           | `minLength = N/A`  
|                |         |           | `maxLength = 50` |
| reportGroup    | String  | Yes       | The response returns the same value submitted in the Force Capture transaction. 
|                |         |           | `minLength = 1`  
|                |         |           | `maxLength = 25` |

**Child Elements:**
- Required: `litleTxnId`, `response`, `responseTime`, `message`
- Optional: `advancedFraudResults`
4.184 fraudFilterOverride

The `fraudFilterOverride` element is an optional child of both the `authorization` and the `sale` elements. A setting of `true` will override (disable) all fraud filters for the submitted transaction.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
authorization, sale

**Attributes:**
None

**Child Elements:**
None
4.185 fraudResult

The `fraudResult` element is an optional child of the `authorizationResponse`, `saleResponse` and the `authInformation` elements. It contains child elements defining any fraud checking results.

**Parent Elements:**
activateResponse, authorizationResponse, deactivateResponse, loadResponse, saleResponse, unloadResponse, authInformation

**Attributes:**
None

**Child Elements: (All Optional)**
authenticationResult, avsResult, cardValidationResult, advancedAVSResult, advancedFraudResults

**Example:  fraudResult Structure**

```xml
<fraudResult>
  <avsResult>00</avsResult>
  <cardValidationResult>N</cardValidationResult>
  <authenticationResult>2</authenticationResult>
  <advancedAVSResult>011</advancedAVSResult>
  <advancedFraudResults>
    <deviceReviewStatus>pass, fail, review, or unavailable</deviceReviewStatus>
    <deviceReputationScore>Score from ThreatMetix</deviceReputationScore>
    <triggeredRule>Triggered Rule_may occur multiple times</triggeredRule>
  </advancedFraudResults>
</fraudResult>
```

**NOTE:** The `advancedAVSResults` element applies only to American Express transactions. Also, you must be enabled specifically to use the Advanced AVS feature. Please consult your Relationship Manager for additional information.
4.186 fundingInstructionVoid

The `fundingInstructionVoid` element is the parent element for the transaction type that a Payment Facilitator uses to void a submitted, but not yet processed funding instruction. You must submit the `fundingInstructionVoid` transaction prior to your cutoff time on the same day you submitted the instruction to be voided. This rule also applies to funding instructions submitted on weekends.

Parent Elements:
- `batchRequest`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A    maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 25</code></td>
</tr>
</tbody>
</table>

Child Elements:

Required: `litleTxnId`
4.187 fundingInstructionVoidResponse

The `fundingInstructionVoidResponse` element is the parent element for information returned to you in response to a `fundingInstructionVoid` transaction.

**Parent Elements:**

* batchResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String   | Yes       | The response returns the same value submitted in the credit transaction. 
|                |          |           | minLength = 1  maxLength = 36 |
| customerId     | String   | No        | The response returns the same value submitted in the credit transaction. 
|                |          |           | minLength = N/A  maxLength = 50 |
| reportGroup    | String   | Yes       | The response returns the same value submitted in the credit transaction. 
|                |          |           | minLength = 1  maxLength = 25 |

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`
4.188 fundingSource

The fundingSource element is an optional child of the enhancedAuthResponse element. Through its child elements, it provides information concerning whether the card used for the transaction is Prepaid, Credit, Debit, or FSA and if Prepaid, the available balance, the type of prepaid card, and whether it is reloadable.

Parent Elements:
- enhancedAuthResponse

Attributes:
- None

Child Elements:
- Required: type, availableBalance, reloadable, prepaidCardType

Example: fundingSource Structure

```xml
<fundingSource>
  <type>PREPAID</type>
  <availableBalance>0</availableBalance>
  <reloadable>YES|NO|UNKNOWN</reloadable>
  <prepaidCardType>GIFT</prepaidCardType>
</fundingSource>
```

NOTE: The fundingSource element and its child elements, type and availableBalance are associated with the Insights features (see Issuer Insights on page 24.)

Please consult your Relationship Manager for additional information.
4.189 fundingSubmerchantId

The `fundingSubmerchantId` element is a required child of each funding instruction request transaction type, where it specifies the identifier of the sub-merchant whose funds the instruction moves.

**NOTE:** If you process transactions solely on the Worldpay platform or on both the Worldpay and Worldpay eCommerce platforms, use the Worldpay-supplied Sub-merchant Merchant Id as the value of the `fundingSubmerchantId` element.

If you process solely on the Worldpay eCommerce platform, use the eCommerce-supplied value obtained, after creating the Sub-merchant, by submitting a Sub-Merchant Retrieval Request via the Merchant Provisioner API as the value of the `fundingSubmerchantId` element.

Please refer to the *Worldpay eComm PayFac API Reference Guide* for additional information.

Type = String; minLength = N/A; maxLength = 50

Parent Elements:
- payFacCredit
- payFacDebit
- physicalCheckCredit
- physicalCheckDebit
- reserveCredit
- reserveDebit
- submerchantCredit
- submerchantDebit
- vendorCredit
- vendorDebit
- fastAccessFunding
- fastAccessFundingResponse

Attributes:
None

Child Elements:
None
4.190 fundsTransferId

The `fundsTransferId` element is a required child of each funding instruction request/response transaction type, where it specifies the Payment Facilitator assigned identifier for the transaction. You must use unique values for each transaction across your entire organization. The system mirrors this identifier back in the response messages.

**Type** = String; **minLength** = N/A; **maxLength** = 36 (for all except `fastAccessFunding` and `fastAccessFundingResponse`)

**Type** = String; **minLength** = N/A; **maxLength** = 16 (for `fastAccessFunding` and `fastAccessFundingResponse`)

**Parent Elements:**
- `payFacCredit`
- `payFacCreditResponse`
- `payFacDebit`
- `payFacDebitResponse`
- `physicalCheckCredit`
- `physicalCheckCreditResponse`
- `physicalCheckDebit`
- `physicalCheckDebitResponse`
- `reserveCredit`
- `reserveCreditResponse`
- `reserveDebit`
- `reserveDebitResponse`
- `submerchantCredit`
- `submerchantCreditResponse`
- `submerchantDebit`
- `submerchantDebitResponse`
- `vendorCredit`
- `vendorCreditResponse`
- `vendorDebit`
- `vendorDebitResponse`
- `fastAccessFunding`
- `fastAccessFundingResponse`

**Attributes:**
None

**Child Elements:**
None
4.191 giftCardAuthReversal

The `giftCardAuthReversal` element is the parent element for the transaction type that reverses an authorization on a Gift Card.

**Parent Elements:**
- `litleOnlineRequest`
- `litleRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements: (Required)**
- `card`
- `originalRefCode`
- `originalAmount`
- `originalTxnTime`
- `originalSystemTraceId`
- `originalSequenceNumber`

**Child Elements: (Optional)**
- `litleTxnId`
4.192 giftCardAuthReversalResponse

The `giftCardAuthReversalResponse` element is the parent element for information returned to you in response to a Gift Card Auth Reversal transaction.

Parent Elements:
- `litleOnlineResponse`
- `litleResponse`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id            | String  | Yes       | The response returns the same value submitted in the Force Capture transaction.  
               |         |           | `minLength = 1`  
               |         |           | `maxLength = 36` |
| customerId    | String  | No        | The response returns the same value submitted in the Force Capture transaction.  
               |         |           | `minLength = N/A`  
               |         |           | `maxLength = 50` |
| reportGroup   | String  | Yes       | The response returns the same value submitted in the Force Capture transaction.  
               |         |           | `minLength = 1`  
               |         |           | `maxLength = 25` |

Child Elements:
- Required: `litleTxnId`, `response`, `responseTime`, `message`, `giftCardResponse`
- Optional: `postDate` (returned for online transactions only)
4.193 giftCardBin

The giftCardBin element is a required child of the virtualGiftCard element defining the requested BIN of the virtual Gift Card number requested.

Type = String; minLength = N/A; maxLength = 10

Parent Elements:

virtualGiftCard

Attributes:

None

Child Elements:

None
4.194 giftCardCapture

The `giftCardCapture` element is the parent element for the transaction type that captures funds previously authorized on a Gift Card.

**Parent Elements:**
`litleOnlineRequest, litleRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:** (Required)
- `card`, `captureAmount`, `originalRefCode`, `originalAmount`, `originalTxnTime`

**Child Elements:** (Optional)
- `litleTxnId`
4.195 giftCardCaptureResponse

The giftCardCaptureResponse element is the parent element for information returned to you in response to a Gift Card Capture transaction.

Parent Elements:
litleOnlineResponse, littleResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Force Capture transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Force Capture transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Force Capture transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: littleTxnId, response, responseTime, message, giftCardResponse
Optional: fraudResult, postDate (returned for online transactions only)
## 4.196 giftCardCredit

The `giftCardCredit` element is the parent element for all Gift Card Credit transactions. There are two possible structures for this message type. You use the first structure for credit transactions against captures/sales processed by Worldpay. The alternate structure applies to credits against captures/sales not processed by Worldpay. You can use this transaction type in either Online or Batch submissions.

### Parent Elements:
- `litleOnlineRequest`, `batchRequest`

### Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 25</code></td>
</tr>
</tbody>
</table>

### Child Elements:

For credits to transactions processed by Worldpay (Required): `creditAmount`, `card`

For credits to transactions processed by Worldpay (Optional): `litleTxnId`

For credits to transactions not processed by Worldpay (Required): `orderId`, `creditAmount`, `orderSource`, `card`
4.197 giftCardCreditResponse

The `giftCardCreditResponse` element is the parent element for information returned to you in response to a Gift Card Credit transaction.

**Parent Elements:**
- `litleOnlineResponse`
- `litleResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Force Capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Force Capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Force Capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`, `giftCardResponse`

Optional: `fraudResult`, `postDate` (returned for online transactions only)
4.198 giftCardResponse

The giftCardResponse element is a required child of several transaction types and optional for several others. Through its child elements, it provides details about the available balance on a Gift Card, as well as additional reference items used in subsequent, related transactions. For example, in a giftCardAuthReversal transaction, you must include values of the refCode, systemTraceId, and sequenceNumber from the authorizationResponse in the originalRefCode, originalSystemTraceId, and originalSequenceNumber elements respectively.

Parent Elements:
Optional (per schema, but always returned for Gift Card Transactions): authorizationResponse, captureGivenAuthResponse, forceCaptureResponse, saleResponse

Attributes:
None

Child Elements (All Optional, but must contain at least one child):
Optional: txnTime, refCode, systemTraceId, sequenceNumber, availableBalance, beginningBalance, endingBalance, cashBackAmount

NOTE: Although included in the schema, we do not support the beginningBalance, endingBalance, and cashBackAmount elements currently.

Example: giftCardResponse Structure
<giftCardResponse>
  <txnTime>2016-11-15T12:00:00</txnTime>
  <refCode>abc123</refCode>
  <systemTraceId>123456</systemTraceId>
  <sequenceNumber>123456</sequenceNumber>
  <availableBalance>Balance Available on Gift Card</availableBalance>
</giftCardResponse>
### 4.199 giropay

The `giropay` element is a child of the `sale` transaction that, through its child elements, defines information needed to process an Giropay (Real-time Bank Transfer) transaction. At this time, you can use the iDeal method of payment in Online transactions only.

**Parent Elements:**
- `sale`

**Attributes:**
- None

**Child Elements:**
- Optional: `preferredLanguage`

**Example:** giropay Structure

```xml
<giropay>
    <preferredLanguage>Country Code of Language</preferredLanguage>
</giropay>
```
4.200 giropayResponse

The giropayResponse element is a child of the saleResponse transaction when the method of payment in the request is giropay. It contains child elements that you should store for future use/reference.

Parent Elements:
saleResponse

Attributes:
None

Child Elements (all Optional):
paymentPurpose, redirectUrl, redirectToken

Example: giropayResponse Structure

```xml
<giropayResponse>
  <redirectUrl>URL of Worldpay Supplied Mandate</redirectUrl>
  <redirectToken>Map Mandate to litleTxnId</redirectToken>
  <paymentPurpose>Reference Number + Billing Descriptor</paymentPurpose>
</giropayResponse>
```
4.201 header

The header element is a required child of the applepay element. Its child elements provides information required to process the Apple Pay transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 10000

**Parent Elements:**
applepay

**Attributes:**
None

**Child Elements (All Optional, but must contain at least one child):**
Optional: applicationData, ephemeralPublicKey, publicKeyHash, transactionId

**Example:** header Structure

```xml
<header>Password</header>
   <applicationData>SHA-256 Hash Hex Encoded of App Data Property</applicationData>
   <ephemeralPublicKey>Base64 Encoded Ephemeral Public Key</ephemeralPublicKey>
   <publicKeyHash>Base64 Hash of Public Key Bytes from Merchant Certificate</publicKeyHash>
   <transactionId>Hex Transaction Id</transactionId>
</header>
```
4.202 healthcareAmounts

The healthcareAmount element is a required child of the healthcareIIAS element. Through its child elements, it provides details about the dollar amount and type of IIAS qualified items purchased using Healthcare Prepaid cards.

The value used for the totalHealthcareAmount child must be the sum of the values applied to the following elements: RxAmount, visionAmount, clinicOtherAmount, and dentalAmount.

Parent Elements:
healthcareIIAS

Attributes:
None

Child Elements:
Required: totalHealthcareAmount
Optional: RxAmount, visionAmount, clinicOtherAmount, dentalAmount

Example: fundingSource Structure
<healthcareAmounts>
  <totalHealthcareAmount>Total of Healthcare Items</totalHealthcareAmount>
  <RxAmount>Amount for Medications</RxAmount>
  <visionAmount>Amount for Vision Items</visionAmount>
  <clinicOtherAmount>Amount for Clinic Charges</clinicOtherAmount>
  <dentalAmount>Amount for Dental Charges</dentalAmount>
</healthcareAmounts>
4.203 healthcareIIAS

The healthcareIIAS element is an optional child of Authorization and Sale transactions. Through its child elements, it provides information about IIAS qualified items purchased using Healthcare Prepaid cards.

Parent Elements:
authorization, sale

Attributes:
None

Child Elements:
Required: healthcareAmounts, IIASFlag

Example: fundingSource Structure

<healthcareIIAS>
  <healthcareAmounts>
    <totalHealthcareAmount>Total of Healthcare Items</totalHealthcareAmount>
    <RxAmount>Amount for Medications</RxAmount>
    <visionAmount>Amount for Vision Items</visionAmount>
    <clinicOtherAmount>Amount for Clinic Charges</clinicOtherAmount>
    <dentalAmount>Amount for Dental Charges</dentalAmount>
  </healthcareAmounts>
  <IIASFlag>Y</IIASFlag>
</healthcareIIAS>
4.204 iban

The `iban` element is a required child of the `sepaDirectDebit` element. It defines the International Bank Account Number of the customer.

Type = String; `minLength = 15`; `maxLength = 34`

**Parent Elements:**
- `sepaDirectDebit`

**Attributes:**
- None

**Child Elements:**
- None
4.205 ideal

The `ideal` element is a child of the `sale` transaction that, through its child elements, defines information needed to process an iDEAL (Real-time Bank Transfers) transaction. At this time, you can use the iDEAL method of payment in Online transactions only.

**Parent Elements:**

- sale

**Attributes:**

- None

**Child Elements:**

- Optional: `preferredLanguage`

**Example: ideal Structure**

```xml
<ideal>
    <preferredLanguage>Country Code of Language</preferredLanguage>
</ideal>
```
4.206 idealResponse

The idealResponse element is a child of the saleResponse transaction when the method of payment in the request is ideal. It contains child elements that you should store for future use/reference.

Parent Elements:
saleResponse

Attributes:
None

Child Elements (all Optional):
paymentPurpose, redirectUrl, redirectToken

Example: idealResponse Structure

    <idealResponse>
      <redirectUrl>URL of Worldpay Supplied Mandate</redirectUrl>
      <redirectToken>Map Mandate to litleTxnId</redirectToken>
      <paymentPurpose>Reference Number + Billing Descriptor</paymentPurpose>
    </idealResponse>
4.207 IIASFlag

The IIASFlag element is a required child of the healthcareIIAS element. This element only supports a value of Y.

**Type** = String (enum); **minLength** = N/A; **maxLength** = 1; **Valid Value** = Y

**Parent Elements:**

- healthcareIIAS

**Child Elements:**

- None
**4.208 incomeAmount**

The `incomeAmount` element is an optional child of the `customerInfo` element and defines the yearly income of the customer. It is used in combination with several other elements to provide information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

*Type* = Long; *minLength* = N/A; *maxLength* = N/A

**Parent Elements:**
- customerInfo

**Attributes:**
- None

**Child Elements:**
- None
4.209 incomeCurrency

The `incomeCurrency` element is an optional child of the `customerInfo` element and defines the currency of the `incomeAmount` element. The default value is USD (United States Dollars). It is used in combination with several other elements to provide information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
customerInfo

**Attributes:**
None

**Child Elements:**
None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUD</td>
<td>Australian Dollar</td>
</tr>
<tr>
<td>CAD</td>
<td>Canadian Dollar</td>
</tr>
<tr>
<td>CHF</td>
<td>Swiss Francs</td>
</tr>
<tr>
<td>DKK</td>
<td>Denmark Kroner</td>
</tr>
<tr>
<td>EUR</td>
<td>Euro</td>
</tr>
<tr>
<td>GBP</td>
<td>United Kingdom Pound</td>
</tr>
<tr>
<td>HKD</td>
<td>Hong Kong Dollar</td>
</tr>
<tr>
<td>JPY</td>
<td>Japanese Yen</td>
</tr>
<tr>
<td>NOK</td>
<td>Norwegian Krone</td>
</tr>
<tr>
<td>NZD</td>
<td>New Zealand Dollar</td>
</tr>
<tr>
<td>SEK</td>
<td>Swedish Kronor</td>
</tr>
<tr>
<td>SGD</td>
<td>Singapore Dollar</td>
</tr>
<tr>
<td>USD (default)</td>
<td>United States Dollar</td>
</tr>
</tbody>
</table>
The international element is an optional child of the filtering element. To disable the filtering operation for a selected transaction include the international element with a setting of false.

Type = Boolean; Valid Value = false

Parent Elements:
filtering

Attributes:
None

Child Elements:
None
4.211 intervalType

The intervalType element is a required child of the createPlan element and defines the billing period associated with the Plan.

**Type** = String (enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
createPlan

**Attributes:**
None

**Child Elements:**
None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANNUAL</td>
<td>The billing period is once per year.</td>
</tr>
<tr>
<td>SEMIANNUAL</td>
<td>The billing period is twice per year.</td>
</tr>
<tr>
<td>QUARTERLY</td>
<td>The billing period is every three months.</td>
</tr>
<tr>
<td>MONTHLY</td>
<td>The billing period is every month.</td>
</tr>
<tr>
<td>WEEKLY</td>
<td>The billing period is every week.</td>
</tr>
</tbody>
</table>
4.212 invoiceReferenceNumber

The invoiceReferenceNumber element is an optional child of the enhancedData element, which specifies the merchant’s invoice number. If you do not know the invoice number, do not include this element.

Type = String; minLength = 1; maxLength = 15

Parent Elements:

enhancedData

Attributes:

None

Child Elements:

None
4.213 issuerCountry

The `issuerCountry` element is an optional child of the `enhancedAuthResponse` element, which defines the country of the bank that issued the card submitted in the Authorization or Sale transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 3

**Parent Elements:**
- `enhancedAuthResponse`

**Attributes:**
- None

**Child Elements:**
- None
### 4.214 itemCategoryCode

The `itemCategoryCode` element is an optional child of the `billMeLaterRequest` element and defines the PayPal Credit item category for the type of product sold.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

- **Type** = Integer; **totalDigits** = 4

- **Parent Elements:**
  - `billMeLaterRequest`

- **Attributes:**
  - None

- **Child Elements:**
  - None
4.215 itemDescription

The itemDescription element is a required child of the lineItemData element, which provides a brief text description of the item purchased.

Type = String; minLength = N/A; maxLength = 26

Parent Elements:
lineItemData

Attributes:
None

Child Elements:
None
4.216 itemDiscountAmount

The `itemDiscountAmount` element is an optional child of the `lineItemData` element, which specifies the item discount amount. Although an optional element, it is required by Visa for Level III Interchange rates. The value must be greater than or equal to 0. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**

`lineItemData`

**Attributes:**

None

**Child Elements:**

None
4.217 itemSequenceNumber

The `itemSequenceNumber` element is an optional child of the `lineItemData` element (required for Visa transactions). When providing line item data, you must number each item sequentially starting with 1.

**Type** = Integer; **minInclusive value** = 1, **maxInclusive value** = 99

**Parent Elements:**
- `lineItemData`

**Attributes:**
- None

**Child Elements:**
- None
4.218 ksn

The ksn element is a required child of the mpos element. This element defines the Key serial Number returned from the encrypting device. It is created automatically from the unique identifier of the device and an internal transaction counter.

Type = String; minLength = 1; maxLength = 1028

Parent Elements:

mpos

Attributes:

None

Child Elements:

None
4.219 lastName

The lastName element is a child of the billToAddress element, which specifies the last name of the account holder and is required for echeckVerification transactions.

**NOTE:** When performing an eCheck Verification for a corporate account, you must include values for the firstName and lastName element. If you do not have the name of the check issuer, you can use a value of “unavailable” for both elements.

Type = String; minLength = N/A; maxLength = 25

**Parent Elements:**
billToAddress

**Attributes:**
None

**Child Elements:**
None
**4.220 lineItemData**

The `lineItemData` element contains several child elements used to define information concerning individual items in the order. Although the schema defines it as an optional child of the `enhancedData` element, it is required for Level III interchange rates.

**NOTE:** MasterCard and Visa allow up to 99 instances of this element in a transaction. American Express allows a maximum of 4 instances of this element in a transaction.

**Parent Elements:**

- `enhancedData`

**Attributes:**

None

**Child Elements:**

- **Required:** `itemDescription`
- **Optional:** `itemSequenceNumber`, `productCode`, `quantity`, `unitOfMeasure`, `taxAmount`, `lineItemTotal`, `lineItemTotalWithTax`, `itemDiscountAmount`, `commodityCode`, `unitCost`, `detailTax`

**NOTE:** When including the `lineItemData` element, please be aware of the following rules for its child elements:

- `itemSequenceNumber` is required by Visa
- `productCode`, `quantity`, `unitOfMeasure`, and `lineItemTotal` are required by Visa and MasterCard
- `itemDiscountAmount`, `commodityCode`, and `unitCost` are required by Visa for Level III Interchange rates

**Example: lineItemData Structure**

```xml
<lineItemData>
    <itemSequenceNumber>Line Item Number within Order</itemSequenceNumber>
    <itemDescription>Description of Item</itemDescription>
    <quantity>Quantity of Item</quantity>
    <unitOfMeasure>Unit of Measurement Code</unitOfMeasure>
    <taxAmount>Sales Tax or VAT of Item</taxAmount>
    <lineItemTotal>Total Amount of Line Item</lineItemTotal>
    <lineItemTotalWithTax>taxAmount + lineItemTotal</lineItemTotalWithTax>
    <itemDiscountAmount>Discount Amount</itemDiscountAmount>
    <commodityCode>Card Acceptor Commodity Code for Item</commodityCode>
</lineItemData>
```
<unitCost>Price for One Unit of Item</unitCost>
<detailTax>
  <taxIncludedInTotal>true or false</taxIncludedInTotal>
  <taxAmount>Additional Tax Amount</taxAmount>
  <taxRate>Tax Rate of This Tax Amount</taxRate>
  <taxTypeIdentifier>Tax Type Enum</taxTypeIdentifier>
  <cardAcceptoTaxId>Tax ID of Card Acceptor</cardAcceptoTaxId>
</detailTax>
</lineItemData>
4.221 lineItemTotal

The lineItemTotal element is an optional child of the lineItemData element, which specifies the total cost of the line items purchased, not including tax. For example, if the order was for 500 pencils at $1.00 each, the lineItemTotal would be $500. Although an optional element, it is required by Visa and MasterCard when specifying line item data. The decimal is implied. Example: 500 = $5.00.

Type = Integer; totalDigits = 8

Parent Elements:

lineItemData

Attributes:

None

Child Elements:

None
**4.222 lineItemTotalWithTax**

The `lineItemTotalWithTax` element is an optional child of the `lineItemData` element, which specifies the total cost of the line items purchased including tax. If the tax is not known, do not include this element. The decimal is implied. Example: 500 = $5.00.

**Type =** Integer; **totalDigits =** 8

**Parent Elements:**

`lineItemData`

**Attributes:**

None

**Child Elements:**

None
4.223 litleInternalRecurringRequest

The litleInternalRecurringRequest element and its children is an element structure that exists solely for internally (to system) generated transactions associated with recurring payments managed by the Recurring engine. You do not need to code for this structure.

Parent Elements:
sale

Attributes:
None

Child Elements:
subscriptionId, recurringTxnId, finalPayment
4.224 litleOnlineRequest

This is the root element for all cnpAPI Online requests.

Parent Elements:
None

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Yes</td>
<td>Defines the cnpAPI schema version against which the XML is validated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 10</td>
</tr>
<tr>
<td>xmlns</td>
<td>String</td>
<td>Yes</td>
<td>Defines the URI of the schema definition. This is a fixed location and must be specified as: <a href="http://www.litle.com/schema">http://www.litle.com/schema</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 38</td>
</tr>
<tr>
<td>merchantId</td>
<td>String</td>
<td>Yes</td>
<td>A unique string used to identify the merchant within the system.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: International currencies are supported on a per merchantId basis.</td>
</tr>
<tr>
<td>loggedInUser</td>
<td>String</td>
<td>No</td>
<td>Internal Use Only</td>
</tr>
<tr>
<td>sameDayFunding</td>
<td>Boolean</td>
<td>No</td>
<td>Used for Dynamic Payout Funding Instructions only. Set to true to mark this Funding Instructions for same day funding. Also, see Same Day Funding on page 852.</td>
</tr>
</tbody>
</table>

Child Elements:

Required: authentication

One of the following required: activate, activateReversal, authorization, authReversal, balanceInquiry, cancelSubscription, capture, captureGivenAuth, createPlan, credit, deactivate, deactivateReversal, depositReversal, echeckCredit, echeckRedeposit, echeckSale, echeckVerification, echeckVoid, forceCapture, fraudCheck, giftCardAuthReversal, giftCardCapture, giftCardCredit, load, loadReversal, registerTokenRequest, refundReversal, sale, unload, updateCardValidationNumOnToken, updatePlan, updateSubscription, unloadReversal, void, fastAccessFunding, payFacCredit, payFacDebit, physicalCheckCredit, physicalCheckDebit, reserveCredit, reserveDebit, submerchantCredit, submerchantDebit, vendorCredit, vendorDebit
4.225 litleOnlineResponse

This is the root element for all cnpAPI Online responses.

**Parent Elements:**
None

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Yes</td>
<td>Defines the cnpAPI schema version against which the XML is validated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 10</td>
</tr>
<tr>
<td>xmlns</td>
<td>String</td>
<td>Yes</td>
<td>Defines the URI of the schema definition. This is a fixed location and must be specified as: <a href="http://www.litle.com/schema">http://www.litle.com/schema</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 38</td>
</tr>
<tr>
<td>response</td>
<td>String</td>
<td>Yes</td>
<td>Indicates whether your XML syntax passed validation. Expected values are as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 - XML validation succeeded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - XML validation failed. See the message attribute for more details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Indicates that the submitted content was either improperly formatted XML or non-XML content.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 - Indicates that the submission contains empty or invalid credentials (user and password).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 - Indicates that the merchant has reached the maximum number of concurrent connections.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 - Indicates that systems may have detected message content that violates certain restrictions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 3</td>
</tr>
</tbody>
</table>
Child Elements:


<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>message</td>
<td>String</td>
<td>Yes</td>
<td>XML validation error message. Expected values are as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• If the response attribute returns 0, the message</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>attribute returns the text “Valid Format.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• If the response attribute returns 1, the message</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>attribute returns an error message that helps you</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>to identify and troubleshoot the syntax problem. See XML Validation Error</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Messages on page 829 for example messages.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• If the response attribute returns 2, the message attribute is “System</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Error - Call Litle &amp; Co.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• If the response attribute returns a value of 3, 4, or 5, the message</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>attribute is “There is a problem with the system. Contact</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a>.”</td>
</tr>
</tbody>
</table>

minLength = N/A maxLength = 512
4.226 litleRequest

This is the root element for all cnpAPI Batch requests.

Parent Elements:
None

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| version        | String | Yes       | Defines the cnpAPI schema version against which the XML is validated.  
|                |        |           | minLength = N/A maxLength = 10                                                                                                             |
| xmlns          | String | Yes       | Defines the URI of the schema definition. This is a fixed location and must be specified as:  
|                |        |           | minLength = N/A maxLength = 38                                                                                                             |
| id             | String | No        | A unique string to identify the session within the system.  
|                |        |           | minLength = N/A maxLength = 25                                                                                                             |
| numBatchRequests | Integer | Yes | Defines the total number of batchRequest children included in the litleRequest. If the litleRequest contains only an RFRRequest, then set this attribute to "0". |

Child Elements:

Required: authentication

One of the following required: batchRequest, RFRRequest
4.227 litleResponse

This is the root element for all cnpAPI Batch responses.

Parent Elements:
None

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>String</td>
<td>Yes</td>
<td>Defines the cnpAPI schema version against which the XML is validated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 10</td>
</tr>
<tr>
<td>xmlns</td>
<td>String</td>
<td>Yes</td>
<td>Defines the URI of the schema definition. This is a fixed location and must be specified as: <a href="http://www.litle.com/schema">http://www.litle.com/schema</a>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 38</td>
</tr>
<tr>
<td>id</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Litle Request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 25</td>
</tr>
<tr>
<td>response</td>
<td>String</td>
<td>Yes</td>
<td>Indicates whether your XML syntax passed validation. Expected values are as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 - XML validation succeeded.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - XML validation failed. See the message attribute for more details.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Indicates that the submitted content was either improperly formatted XML or non-XML content.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 - Indicates that the submission contains empty or invalid credentials (user and password).</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 - Indicates that the merchant has reached the maximum number of concurrent connections.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 - Indicates that systems may have detected message content that violates certain restrictions.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 3</td>
</tr>
</tbody>
</table>
**Attribute Name** | **Type** | **Required?** | **Description**
--- | --- | --- | ---
message | String | Yes | XML validation error message. Expected values are as follows:
- If the response attribute returns 0, the message attribute returns the text “Valid Format.”
- If the response attribute returns 1, the message attribute returns an error message that helps you to identify and troubleshoot the syntax problem. See XML Validation Error Messages on page 829 for example messages.
- If the response attribute returns 2, the message attribute is “System Error - Call Litle & Co.”
- If the response attribute returns a value of 3, 4, or 5, the message attribute is “There is a problem with the system. Contact eCommerceSupport@vantiv.com.”

**minLength** = N/A  **maxLength** = 512

litleSessionId | Long | Yes | A unique value assigned by Worldpay to identify the session.

**minLength** = N/A  **maxLength** = 19

**Child Elements:**
One of the following required: batchResponse, RFRResponse
4.228 litleSessionId

The litleSessionId element is a child of the RFRRequest element used to request the response from a previously submitted Batch. The value of the litleSessionId must be the same as the value returned in the corresponding attribute of the litleResponse.

**Type** = Long; **minLength** = N/A; **maxLength** = 19

**Parent Elements:**

RFRRequest

**Attributes:**

None

**Child Elements:**

None
4.229 litleToken

The litleToken element defines the value of the token. The system returns this value in XML responses when issuing new tokens to replace account numbers. The length of the token is the same as the length of the submitted account number for credit card tokens or a fixed length of seventeen (17) characters for eCheck account tokens.

Type = String; minLength = 13; maxLength = 25

Parent Elements:
The litleToken element is an optional child of each listed parent element.

registerTokenResponse, tokenResponse, newCardTokenInfo, originalCardTokenInfo, originalToken, originalTokenInfo, newTokenInfo, updatedToken, token, echeckToken

Attributes:
None

Child Elements:
None
4.230 litleTxnId

The litleTxnId element is used to identify transactions in the system. The system returns this element in XML responses. You use it in various requests to reference the original transaction. For example, when you submit a Capture transaction, you include the litleTxnId for the associated Authorization.

Type = Long; minLength = N/A; maxLength = 19

Parent Elements:


NOTE: While the litleTxnId is optional in the transaction types listed below, Worldpay recommends you include it whenever possible. Failure to include the litleTxnId results in missing information in the Associated Transaction Stream section of the Transaction Detail screen in iQ.

This element is an optional child of the following: activateReversal, deactivateReversal, depositReversal, giftCardAuthReversal, giftCardCapture, giftCardCredit, loadReversal, refundReversal, unloadReversal

NOTE: Although the schema shows the litleTxnId element as an optional child of the authorization, echeckSale, and sale transactions, under normal circumstances, merchants would never use this option.

Attributes:

None

Child Elements:

None
4.231 load

The load element is the parent element for the transaction type that adds funds to a reloadable Gift Card.

**Parent Elements:**

litleOnlineRequest, batchRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements: (all Required)**

orderid, amount, orderSource, card
4.232 loadResponse

The `loadResponse` element is the parent element for information returned to you in response to a `load` transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**
- `litleOnlineResponse`
- `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Load transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Load transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Load transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

- Required: `litleTxnId`, `response`, `responseTime`, `message`
- Optional: `postDate`, `fraudResult`, `giftCardResponse`
4.233 loadReversal

The `loadReversal` element is the parent element for the transaction type that reverses the loading of a Gift Card.

Parent Elements:
- `litleOnlineRequest`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

Child Elements: (Required)
- `card`, `originalRefCode`, `originalAmount`, `originalTxnTime`, `originalSystemTraceld`, `originalSequenceNumber`

Child Elements: (Optional)
- `litleTnlId`
4.234 loadReversalResponse

The `loadReversalResponse` element is the parent element for information returned to you in response to an `loadReversal` transaction.

**Parent Elements:**
- littleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Load Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Load Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Load Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**
- Required: `littleTxnId`, `response`, `responseTime`, `message`
- Optional: `postDate`, `giftCardResponse`
4.235 mandateProvider

The mandateProvider element is a required child of the sepaDirectDebit element and defines whether the Merchant or Worldpay provides the mandate for customer approval.

Type = String (Enum); **Allowed Values** = Merchant or Vantiv

Parent Elements:
sepaDirectDebit

Attributes:
None

Child Elements:
None
4.236 mandateReference

The mandateReference element is an optional child of both the sepaDirectDebit element and the sepaDirectDebitResponse element. You use this element for recurring payments (after the initial transaction) to provide the reference number that links subsequent payments in a recurring stream to the mandate agreed to at the time of the initial payment. Worldpay returns this value in the sepaDirectDebitResponse associated with the initial payment. It is not returned for subsequent payments in a recurring stream.

Type = String; minLength = N/A; maxLength = 256

Parent Elements:
sepaDirectDebit, sepaDirectDebitResponse

Attributes:
None

Child Elements:
None
**4.237 mandateSignatureDate**

The `mandateSignatureDate` element is an optional child of the `sepaDirectDebit` element and defines the date the consumer agreed to the mandate allowing the merchant to debit their account. Although optional, you should always provide this information when the value for the `mandateProvider` element is Merchant.

Type = Date; Format = YYYY-MM-DD

**Parent Elements:**

- `sepaDirectDebit`

**Attributes:**

- None

**Child Elements:**

- None
4.238 mandateURL

The `mandateURL` element is an optional child of the `sepaDirectDebit` element and defines the URL of the mandate to which the consumer agreed, allowing the merchant to debit their account. Although optional, you should always provide this information when the value for the `mandateProvider` element is Merchant.

**Type** = String; **minLength** = 1; **maxLength** = 2000

**Parent Elements:**
- `sepaDirectDebit`

**Attributes:**
- None

**Child Elements:**
- None
4.239 matchCount

The matchCount element is a required child of the queryTransactionResponse element and defines the number of found transactions that matched the criteria submitted in the queryTransaction.

Type = Integer; totalDigits = 2

Parent Elements:
queryTransactionResponse

Attributes:
None

Child Elements:
None
4.240 merchantData

The merchantData element is an optional child element of several transaction types. You can use its children to track transactions based upon marketing campaigns, affiliates, or other user defined parameter.

Parent Elements:
authorization, captureGivenAuth, credit, echeckCredit, echeckPreNoteCredit, echeckPreNoteSale, echeckRedeposit, echeckSale, echeckVerification, forceCapture, sale

Attributes:
None

Child Elements (all optional):
affiliate, campaign, merchantGroupingId
4.241 merchantGroupingId

The `merchantGroupingId` element is an optional child element of the `merchantData` element. You can use it to track transactions based upon this user defined parameter.

**Type** = String; **minLength** = N/A; **maxLength** = 25

**Parent Elements:**
merchantData

**Attributes:**
None

**Child Elements:**
None
4.242 merchantId

The `merchantId` element is a child of the `accountUpdateFileRequestData` element used when you request an Account Update file. This value is a unique string used to identify the merchant within the system.

**Type** = String; **minLength** = N/A; **maxLength** = 50

**Parent Elements:**
accountUpdateFileRequestData

**Attributes:**
None

**Child Elements:**
None

**NOTE:** Several elements use `merchantId` as an attribute, including `batchRequest`, `batchResponse`, and `litleOnlineRequest`.
4.243 message

The message element contains a brief definition of the response code returned for the transaction. When it is a child of the extendedCardResponse element, the only values allowed are either "The account was closed," or "Contact the cardholder for updated information."

For a complete list of response codes and associated messages, please refer to Appendix A.

Type = String; minLength = N/A; maxLength = 512

Parent Elements:

Attributes:
None

Child Elements:
None
4.244 middleInitial

The middleInitial element is a child of the billToAddress element, which specifies the middle initial of the account holder. It is an optional element used for echeckVerification transactions.

**Type** = String; **minLength** = N/A; **maxLength** = 1

**Parent Elements:**

billToAddress

**Attributes:**

None

**Child Elements:**

None
4.245 mpos

The `mpos` element defines payment card information when the transaction originates with a ROAM device.

**Parent Elements:**
authorization, captureGivenAuth, credit, forceCapture, sale, registerTokenRequest

**Attributes:**
None

**Child Elements: (all Required)**
ksn, formatId, encryptedTrack, track1Status, track2Status

**Example: mpos Structure**

```xml
<mpos>
  <ksn>Key Serial Number</ksn>
  <formatId>Format of Encrypted Track</formatId>
  <encryptedTrack>Encrypted Track Data</encryptedTrack>
  <track1Status>Card Validation Number</track1Status>
  <track2Status>Card Validation Number</track2Status>
</mpos>
```
4.246 name

The name element defines the customer name in both the billToAddress and shipToAddress elements. When used as a child of one of the Recurring Engine associated parents (i.e., createAddOn, updateAddOn, createDiscount, updateDiscount, or createPlan), the name element specifies the name of the parent item being created/updated.

Type = String; minLength = N/A; maxLength = 100

Parent Elements:
- billToAddress
- shipToAddress
- createAddOn
- updateAddOn
- createDiscount
- updateDiscount
- createPlan

**NOTE:** The name element is required for Echeck transactions. If you do not submit the customer name in an Echeck transaction, we return Response Code 330 - Invalid Payment Type.

The name element is required for sale transactions using the sepaDirectDebit method of payment and must be a minimum of two characters. Failure to include the name element results in a 901 - Missing Name Response Code. If the value is not at least two characters, the transaction declines with a response code of 902 - Invalid Name

Attributes:
- None

Child Elements:
- None
4.247 networkField

The networkField element is an optional child of the networkResponse element. Its attributes and child elements define the Field Number, Field Name, (Raw) Field Value, as well as any Sub-fields, if applicable.

**Parent Elements:**

networkResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| fieldName      | String   | Yes       | This attribute defines the name of the ISO 8583 field containing the information returned. Type = Enum
|                |          |           | **Allowed Values:**                                                          |
|                |          |           | • Transaction Amount                                                        |
|                |          |           | • Settlement Date                                                            |
|                |          |           | • Authorization Identification Response                                        |
|                |          |           | • Response Code                                                              |
|                |          |           | • Additional Response Data                                                   |
|                |          |           | • Private Use Additional Data                                                |
|                |          |           | • Settlement Currency Code                                                   |
|                |          |           | • Cardholder Billing Currency Code                                           |
|                |          |           | • Additional Amounts                                                         |
|                |          |           | • Reserved Private                                                           |
|                |          |           | • Transaction Description                                                    |
|                |          |           | • Reserved For National Use                                                  |
|                |          |           | • Reserved For Private Use                                                   |
| fieldNumber    | Integer  | Yes       | This attribute defines the number of the ISO 8583 field containing the information returned. Different card networks may use different Field Numbers for the same information.
|                |          |           | **minLength = 1  maxLength = N/A**                                           |

**Child Elements:**

fieldValue, networkSubField

**Example:** networkField Structure with fieldValue

```xml
<networkField fieldNumber="4" fieldName="Transaction Amount">
  <fieldValue>000000001111</fieldValue>
</networkField>
```
Example: networkField Structure with networkSubField

```xml
<networkField fieldNumber="44" fieldName="Additional Response Data">
  <networkSubField fieldNumber="1">
    <fieldValue>7</fieldValue>
  </networkSubField>
  <networkSubField fieldNumber="2">
    <fieldValue>W</fieldValue>
  </networkSubField>
</networkField>
```
4.248 networkResponse

The networkResponse element is an optional child of the enhancedAuthResponse element. Its child elements provide a number of data points returned by the card networks in their ISO 8583 response messages.

Parent Elements:
enhancedAuthResponse

Attributes:
None

Child Elements:
endpoint, networkField

Example: networkResponse Structure

```xml
<networkResponse>
  <endpoint>VISA</endpoint>
  <networkField fieldNumber="4" fieldName="Transaction Amount">
    <fieldValue>000000000962</fieldValue>
  </networkField>
  <networkField fieldNumber="44" fieldName="Additional Response Data">
    <networkSubField fieldNumber="1">
      <fieldValue>7</fieldValue>
    </networkSubField>
    <networkSubField fieldNumber="2">
      <fieldValue>W</fieldValue>
    </networkSubField>
  </networkField>
</networkResponse>
```
### 4.249 networkSubField

The **networkSubField** element is a required child of the **networkField** element, when a subfield exists. Its child element provides the raw subfield data returned by the card networks in their ISO 8583 response messages.

**Parent Elements:**
- networkField

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldNumber</td>
<td>Integer</td>
<td>Yes</td>
<td>This attribute defines the number of the ISO 8583 field or subfield containing the information returned. Different card networks may use different Field Numbers for the same information. minLength = 1 maxLength = N/A</td>
</tr>
</tbody>
</table>

**Child Elements:**
- fieldValue
4.250 networkTransactionId

The `networkTransactionId` element is an optional child of the `authorizationResponse`, and `saleResponse` transactions, returned for Visa and Discover transactions. Visa and Discover use this value to link subsequent payments in a recurring/installment stream back to the initial transaction. You must include this value in the request message (`originalNetworkTransactionId` element) for subsequent recurring payments, if the payment method is either Visa or Discover.

**Type** = String; **minLength** = N/A; **maxLength** = 30

**NOTE:** In the initial transaction of the recurring/installment stream, you must set the `processingType` element to either `initialRecurring` or `initialInstallment`, as applicable.

**Parent Elements:**
- `authorizationResponse`
- `saleResponse`

**Attributes:**
- None

**Child Elements:**
- None
4.251 newAccountInfo

The `newAccountInfo` element is an optional child of the `accountUpdater` element, which contains child elements providing the updated information for the submitted account.

**Parent Elements:**
accountUpdater

**Attributes:**
None

**Child Elements:**
accType, accNum, routingNum

**Example: newAccountInfo Structure**

```xml
<newAccountInfo>
  <accType>Account Type</accType>
  <accNum>New Account Number</accNum>
  <routingNum>New Routing Number</routingNum>
</newAccountInfo>
```
4.252 newCardInfo

The `newCardInfo` element is an optional child of the `accountUpdater` element, which contains child elements providing the updated information for the submitted card.

**Parent Elements:**
- `accountUpdater`

**Attributes:**
None

**Child Elements:**
- `type`, `number`, `expDate`

**Example: newCardInfo Structure**

```xml
<newCardInfo>
    <type>Card Type</type>
    <number>New Account Number</number>
    <expDate>New Expiration Date</expDate>
</newCardInfo>
```
4.253 newCardTokenInfo

The `newCardTokenInfo` element is an optional child of the `accountUpdater` element, which contains child elements providing the updated token information for the submitted token.

**Parent Elements:**
- `accountUpdater`

**Attributes:**
- None

**Child Elements:**
- `litleToken`, `type`, `expDate`, `bin`

**Example: newCardInfo Structure**

```
<newCardTokenInfo>
  <litleToken>New Token</litleToken>
  <type>Card Type</type>
  <expDate>New Expiration Date</expDate>
  <bin>New Card BIN</bin>
</newCardTokenInfo>
```
4.254 newTokenInfo

The newTokenInfo element is an optional child of the accountUpdater element, which contains child elements providing the updated information for the submitted account. The system returns this information when processing a tokenized eCheck transactions and a change (NOC) is found against the account.

Parent Elements:
accountUpdater

Attributes:
None

Child Elements:
accType, litleToken, routingNum

Example: newAccountInfo Structure

```xml
<newTokenInfo>
  <accType>Account Type</accType>
  <litleToken>New Token Number</litleToken>
  <routingNum>New Routing Number</routingNum>
</newTokenInfo>
```
**4.255 nextRecycleTime**

**NOTE:** Worldpay no longer supports the recycling Advice feature.

The `nextRecycleTime` element is an optional child of the `recycleAdvice` element, which specifies the date and time (in GMT) recommended for the next recycle of the declined Authorization/Sale transaction. The format of the element is `YYYY-MM-DDTHH:MM:SSZ`. For example, `2011-04-21T11:00:00Z`.

**NOTE:** Per the ISO8601 standard, the Z appended to the end of the date/time stamp indicates the time is GMT.

Type = `dateTime`; `minLength` = `N/A`; `maxLength` = `20`

**Parent Elements:**

`recycleAdvice`

**Attributes:**

None

**Child Elements:**

None
4.256 number

The `number` element defines the account number associated with the transaction or the new/old account number associated with an update. This is a required child of the `card` element for card-not-present transactions.

**Type** = String; **minLength** = 13; **maxLength** = 25

**Parent Elements:**
accountInformation, card, newCardInfo, originalCardInfo

**Attributes:**
None

**Child Elements:**
None
4.257 numberOfPayments

The numberOfPayments element is defines the number of payments in a recurring billing plan including the initial payment. The timing of subsequent charges is defined by the planCode element. This element is an optional child of both the subscription, and createPlan elements. When submitted as a child of the subscription element, the value overrides the default value defined in the Plan.

NOTE: For an open-ended Plan, please omit the optional numberOfPayments element in createPlan.
For an open-ended subscription, please omit the optional numberOfPayments element in subscription and reference an open-ended Plan.

Type = Integer; minLength = 1; maxLength = 99

Parent Elements:
subscription, createPlan

Attributes:
None

Child Elements:
None
4.258 onlinePaymentCryptogram

The `onlinePaymentCryptogram` element is an optional child of the `applepayResponse` element and provides the BASE64 Encoded signature cryptogram associated with the Apple Pay transaction.

**Type** = Base 64 Encoded String; **minLength** = N/A; **maxLength** = 56

**Parent Elements:**
- applepayResponse

**Attributes:**
None

**Child Elements:**
None
4.259 orderDate

The orderDate element is an optional child of the enhancedData element, which specifies the date the order was placed. If you do not know the order date, do not include this element.

Type = Date; Format = YYYY-MM-DD

Parent Elements:
enhancedData

Attributes:
None

Child Elements:
None
4.260 orderId

The orderId element is a required child of several transaction types and defines a merchant-assigned value representing the order in the merchant’s system.

Type = String; minLength = N/A; maxLength = 25

**NOTE:** If you are using the orderId element as the transaction signature for the Recycling Engine, do not use the pipe character ("|") in the orderId. Use of the pipe character in this scenario will cause recycling errors.

**Parent Elements:**

accountUpdate, accountUpdateResponse, activate, authorization, authorizationResponse, balanceInquiry, credit, captureGivenAuth, deactivate, echeckCredit, echeckPreNoteCredit, echeckPreNoteSale, echeckSale, echeckVerification, forceCapture, giftCardCredit, load, registerTokenRequest, sale, saleResponse, unload, updateCardValidationNumOnToken

**Attributes:**

None

**Child Elements:**

None
4.261 orderSource

The orderSource element defines the order entry source for the type of transaction.

**Type** = Choice (enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
activate, authorization, balanceInquiry, captureGivenAuth, credit, deactivate, echeckCredit, echeckPreNoteCredit, echeckPreNoteSale, echeckSale, echeckVerification, forceCapture, load, sale, unload

**Attributes:**
None

**Child Elements:**
None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3dsAuthenticated</td>
<td>Use this value only if you authenticated the cardholder via an approved 3DS system such as Visa Verified By Visa and MasterCard SecureCode. This value applies to Visa and MasterCard transactions only.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> Your Merchant Profile must be configured to process 3DS type payments and accept this value.</td>
</tr>
<tr>
<td>3dsAttempted</td>
<td>Use this value only if you attempted to authenticate the cardholder via an approved 3DS system such as Visa VerifiedByVisa and MasterCard SecureCode, but either the Issuer or cardholder is not participating in the 3DS program. This value applies to Visa and MasterCard transactions only. If this is a MasterCard transaction, you must include the authenticationValue returned by MasterCard.</td>
</tr>
<tr>
<td></td>
<td><strong>NOTE:</strong> Your Merchant Profile must be configured to process 3DS type payments and accept this value.</td>
</tr>
<tr>
<td>echeckppd</td>
<td><em>(eCheck only)</em> Use this value for eCheck PPD transactions (Prearranged Payment and Deposit Entries). This type of transaction occurs when a merchants receives a written authorization, including a voided paper check, from a consumer so that the merchant can debit the consumer account. These transactions can be single entry or recurring debits to a consumer’s account.</td>
</tr>
</tbody>
</table>

**NOTE:** If you submit the wrong orderSource value, we return the response code 370 - Internal System Error - Contact Litle.

eCheckSale transactions must use an orderSource of one of the following: telephone, ecommerce, echeckppd, or recurringtel.
<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ecommerce</td>
<td>The transaction is an Internet or electronic commerce transaction.</td>
</tr>
<tr>
<td>installment</td>
<td>The transaction in an installment payment.</td>
</tr>
<tr>
<td>mailorder</td>
<td>The transaction is for a single mail order transaction.</td>
</tr>
<tr>
<td>recurring</td>
<td>The transaction is a recurring transaction. For Visa transactions, you can use this value for all transactions in a recurring stream including the initial transaction.</td>
</tr>
<tr>
<td>retail</td>
<td>The transaction is a Swiped or Keyed Entered retail purchase transaction.</td>
</tr>
<tr>
<td>telephone</td>
<td>The transaction is for a single telephone order.</td>
</tr>
<tr>
<td>recurringtel</td>
<td><em>(eCheck only)</em> The transaction is a recurring eCheck transaction initiated via telephone</td>
</tr>
<tr>
<td>applepay</td>
<td>The transaction uses the Apple Pay service.</td>
</tr>
<tr>
<td>androidpay</td>
<td>The transaction uses the Android Pay/Google Pay service.</td>
</tr>
</tbody>
</table>
4.262 originalAccountInfo

The originalAccountInfo element is an optional child of the accountUpdater element, which contains child elements providing the original information for the submitted account.

**Parent Elements:**
accountUpdater

**Attributes:**
None

**Child Elements:**
accType, accNum, routingNum

**Example: originalAccountInfo Structure**

```
<originalAccountInfo>
  <accType>Account Type</accType>
  <litleToken>Original Token Number</litleToken>
  <routingNum>Original Routing Number</routingNum>
</originalAccountInfo>
```
4.263 origAccountNumber

The origAccountNumber element is an optional child of the queryTransaction element and defines the account number of the credit/debit/gift card used in the original transaction.

**NOTE:** If you perform a query for an eCheck transaction, do not use this element to designate the checking/savings account number. Use this element for Credit/Debit/Gift card account numbers only.

Type = String; minLength = 13; maxLength = 25

Parent Elements:
queryTransaction

Attributes:
None

Child Elements:
None
4.264 origActionType

The `origActionType` element is a required child of the `queryTransaction` element and defines the transaction type of original transaction.

**Type** = String (Enum); **minLength** = 1; **maxLength** = 2 (Valid values shown below)

Parent Elements:
- `queryTransaction`

Attributes:
- None

Child Elements:
- None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Transaction Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>authorization</td>
</tr>
<tr>
<td>AR</td>
<td>activateReversal</td>
</tr>
<tr>
<td>D</td>
<td>deposit or sale</td>
</tr>
<tr>
<td>G</td>
<td>activate</td>
</tr>
<tr>
<td>I</td>
<td>unload</td>
</tr>
<tr>
<td>J</td>
<td>deactivate</td>
</tr>
<tr>
<td>L</td>
<td>load</td>
</tr>
<tr>
<td>LR</td>
<td>loadReversal</td>
</tr>
<tr>
<td>P</td>
<td>authReversal</td>
</tr>
<tr>
<td>R</td>
<td>credit</td>
</tr>
<tr>
<td>RR</td>
<td>refundReversal</td>
</tr>
<tr>
<td>S</td>
<td>echeckSale</td>
</tr>
<tr>
<td>T</td>
<td>echeckCredit</td>
</tr>
<tr>
<td>UR</td>
<td>unloadReversal</td>
</tr>
<tr>
<td>V</td>
<td>void</td>
</tr>
<tr>
<td>W</td>
<td>depositReversal</td>
</tr>
<tr>
<td>X</td>
<td>echeckVoid</td>
</tr>
</tbody>
</table>
4.265 origId

The `origId` element is a required child of the `queryTransaction` element and defines the `id` attribute used in the original transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 25

**Parent Elements:**
`queryTransaction`

**Attributes:**
None

**Child Elements:**
None
4.266 originalAmount

The `originalAmount` element is a required child of several gift card reversal transactions, as well as the `giftCardCapture` transaction, where it specifies the amount of the original gift card transaction. In the case of the `giftCardCapture` transaction, this is the amount associated with the authorization. In the case of a reversal transaction, it is the amount from the transaction you want to reverse. For example, for a `loadReversal` it specifies the amount from the `load` transaction.

Type = Integer; totalDigits = 6

Parent Elements:
activateReversal, deactivateReversal, depositReversal, giftCardAuthReversal, giftCardCapture, loadReversal, refundReversal, unloadReversal

Attributes:
None

Child Elements:
None
4.267 originalCard

The originalCard element is an optional child of the accountUpdateResponse element, which contains child elements providing the original information for the submitted card.

Parent Elements:
accountUpdateResponse

Attributes:
None

Child Elements:
type, number, expDate

Example: originalCard Structure

```xml
<originalCard>
  <type>Card Type</type>
  <number>Old Account Number</number>
  <expDate>Old Expiration Date</expDate>
</originalCard>
```
4.268 originalCardInfo

The `originalCardInfo` element is an optional child of the `accountUpdater` element, which contains child elements providing the original information for the submitted card.

**Parent Elements:**
accountUpdater

**Attributes:**
None

**Child Elements:**
type, number, expDate

**Example: originalCard Structure**

```xml
<originalCardInfo>
  <type>Card Type</type>
  <number>Old Account Number</number>
  <expDate>Old Expiration Date</expDate>
</originalCardInfo>
```
4.269 originalCardTokenInfo

The originalCardTokenInfo element is an optional child of the accountUpdater element, which contains child elements providing the original information for the submitted token.

Parent Elements:
accountUpdater

Attributes:
None

Child Elements:
litleToken, type, expDate, bin

Example: originalCard Structure

```
<originalCardTokenInfo>
  <litleToken>Old Token</litleToken>
  <type>Card Type</type>
  <expDate>Old Expiration Date</expDate>
  <bin>Old Card BIN</bin>
</originalCardTokenInfo>
```
4.270 originalNetworkTransactionId

The `originalNetworkTransactionId` element is an optional child of the `authorization`, `captureGivenAuth`, and `sale` transactions. It defines the `networkTransactionId` returned in the response messages for Visa and Discover Auth/Sale transactions. You must include this element and the original value returned for subsequent (after the initial) Visa or Discover recurring/installment payments.

Type = String; minLength = N/A; maxLength = 30

**NOTE:** In the initial transaction of the recurring/installment stream, you must set the `processingType` element to either `initialRecurring` or `initialInstallment`, as applicable.

If you fail to include this element/value for a recurring/installment Visa or Discover transaction, the card network rejects the transaction.

Parent Elements:

`authorization`, `captureGivenAuth`, `sale`

Attributes:

None

Child Elements:

None
### 4.271 originalRefCode

The `originalRefCode` element is a required child of several gift card transactions, where it specifies the value returned in the `refCode` element of the associated `giftCardResponse` element of the response messages.

**Type** = String; **minLength** = N/A; **maxLength** = 6

**Parent Elements:**
- activateReversal
- deactivateReversal
- depositReversal
- giftCardAuthReversal
- giftCardCapture
- loadReversal
- refundReversal
- unloadReversal

**Attributes:**
None

**Child Elements:**
None
4.272 originalSequenceNumber

The originalSequenceNumber element is a required child of several gift card reversal transactions, where it specifies the value returned in the sequenceNumber element of the associated giftCardResponse element of the response messages.

**Type** = String; **totalDigits** = 6; **Allowed Characters**: 0 - 9

**Parent Elements:**
activateReversal, deactivateReversal, depositReversal, giftCardAuthReversal, loadReversal, refundReversal, unloadReversal,

**Attributes:**
None

**Child Elements:**
None
4.273 originalSystemTraceId

The `originalSystemTraceId` element is a required child of several gift card reversal transactions, where it specifies the value returned in the `systemTraceId` element of the associated `giftCardResponse` element from the response messages of the transaction you wish to reverse. For example, for a `loadReversal` transaction, you use the value returned in the `loadResponse` message.

Type = Integer; totalDigits = 6

Parent Elements:
activateReversal, deactivateReversal, depositReversal, giftCardAuthReversal, loadReversal, refundReversal, unloadReversal

Attributes:
None

Child Elements:
None
4.274 originalToken

The originalToken element is an optional child of the accountUpdateResponse element, which contains child elements providing the original information for the submitted token.

Parent Elements:
accountUpdateResponse

Attributes:
None

Child Elements:
type, number, expDate, bin

Example: originalCard Structure

```xml
<originalToken>
  <litleToken>Old Token Number</litleToken>
  <expDate>Old Expiration Date</expDate>
  <type>Card Type</type>
  <bin>Card BIN</bin>
</originalToken>
```
4.275 originalTokenInfo

The `originalTokenInfo` element is an optional child of the `accountUpdater` element, which contains child elements providing the original token information for the submitted account. The system returns this information when processing a tokenized eCheck transactions and a change (NOC) is found against the account.

**Parent Elements:**
accountUpdater

**Attributes:**
None

**Child Elements:**
accType, litleToken, routingNum

**Example:** originalAccountInfo Structure

```
<originalTokenInfo>
  <accType>Account Type</accType>
  <litleToken>Old Account Number</litleToken>
  <routingNum>Old Routing Number</routingNum>
</originalTokenInfo>
```
4.276 originalTransactionAmount

The originalTransactionAmount element is an optional child of the authorization, captureGivenAuth, and sale transactions. It defines the amount from the initial transaction in a recurring/installment stream. You must include this element and the original amount in subsequent (after the initial) recurring/installment payments, if the payment method is Discover. You must also include the originalNetworkTransactionId element using the value from the networkTransactionId element returned in the initial transaction of the stream.

Type = Integer; totalDigits = 12

Parent Elements:
authorization, captureGivenAuth, sale

Attributes:
None

Child Elements:
None

NOTE: Currently, Discover does not require the inclusion of this element.

NOTE: In the initial transaction of the recurring/installment stream, you must set the processingType element to either initialRecurring or initialInstallment, as applicable.
4.277 originalTxnTime

The `originalTxnTime` element is a required child of several gift card transactions, which specifies the date and time the original transaction was processed by Worldpay. The system returns this value in the `txnTime` child of the `giftCardResponse` element. The format of the element is YYYY-MM-DDTHH:MM:SS. For example, 2016-11-21T11:00:00.

*Type* = `dateTime`; *minLength* = N/A; *maxLength* = 20

**Parent Elements:**

- `activateReversal`
- `deactivateReversal`
- `depositReversal`
- `giftCardAuthReversal`
- `giftCardCapture`
- `loadReversal`
- `refundReversal`
- `unloadReversal`

**Attributes:**

None

**Child Elements:**

None
4.278 origLitleTxnId

The \texttt{origLitleTxnId} element is an optional child of the \texttt{queryTransaction} element and defines the value of the \texttt{litleTxnId} element assigned to the original transaction and returned in the response message.

\textbf{Type} = \texttt{Long}; \hspace{0.5cm} \textbf{minLength} = \texttt{N/A}; \hspace{0.5cm} \textbf{maxLength} = 19

\textbf{Parent Elements:}

\texttt{queryTransaction}

\textbf{Attributes:}

None

\textbf{Child Elements:}

None
4.279 origOrderId

The origOrderId element is an optional child of the queryTransaction element and defines the merchant-assigned value for the orderId element submitted in the original transaction.

Type = String; minLength = N/A; maxLength = 25

Parent Elements:
queryTransaction

Attributes:
None

Child Elements:
None
4.280 password

The password element is a required child of the authentication element. It is used in combination with the user element to authenticate that the message is from a valid source.

Type = String; minLength = N/A; maxLength = 20

Parent Elements:
authentication

Attributes:
None

Child Elements:
None
4.281 payerId

The `payerId` element is a required child of the `paypal` element for all cases except for an Online Credit transaction, where you can choose between this element and the `payerEmail` element. This element specifies the Payer Id returned from PayPal.

**NOTE:** The value of the `<payerId>` element must match the PAYERID value returned by the GetExpressCheckout call operation to PayPal.

Type = String; minLength = 1; maxLength = 17

**Parent Elements:**

- `paypal`

**Attributes:**

- None

**Child Elements:**

- None
4.282 payFacCredit

The `payFacCredit` element is the parent element for the transaction type that a Payment Facilitator uses to distribute funds to themselves (i.e., from the PayFac Settlement Account to the PayFac Operating Account). If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**

`batchRequest`, `litleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td>customerld</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to true and the entire original response.</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `amount`, `fundingSubmerchantId`, `fundsTransferId`
4.283 payFacCreditResponse

The `payFacCreditResponse` element is the parent element for information returned to you in response to a `payFacCredit` transaction.

**Parent Elements:**
- batchResponse
- litleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>payFacCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1   maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the <code>payFacCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>payFacCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1   maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**
- Required: `litleTxnId`, `fundsTransferld`, `response`, `responseTime`, `message`
- Required (Online): `postDate`
4.284 payFacDebit

The `payFacDebit` element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the PayFac Operating Account back to the PayFac Settlement Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**

`batchRequest`, `litleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to <code>true</code> and the entire original response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `amount`, `fundingSubmerchantId`, `fundsTransferId`
4.285 payFacDebitResponse

The payFacDebitResponse element is the parent element for information returned to you in response to a payFacDebit transaction.

Parent Elements: batchRequest, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the payFacDebit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: litleTxnId, fundsTransferId, response, responseTime, message

Required (Online): postDate
4.286 paymentDataType

The `paymentDataType` element is an optional child of the `applepayResponse` element and specifies data type of the payment data associated with an Apple Pay transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**

`applepayResponse`

**Attributes:**

None

**Child Elements:**

None
4.287 paymentPurpose

The `paymentPurpose` element is an optional child of the `idealResponse` and `giropayResponse` elements where it specifies information (equivalent of Bill Descriptor for a credit card transaction) that appears on the consumer bank statement along with a reference string representing the transaction. The value has two components. The first part is a reference number assigned to the transaction. The second part is a default merchant descriptor established during implementation.

**Type** = String; **minLength** = N/A; **maxLength** = 256

**Parent Elements:**
- `idealResponse`
- `giropayResponse`

**Attributes:**
- None

**Child Elements:**
- None
4.288 paypage

The paypage element defines eProtect account information. It replaces the card or token elements in transactions using the eProtect feature of the Vault solution. When you submit the paypage element in a request, response messages will include token information.

Parent Elements:
authorization, captureGivenAuth, credit, forceCapture, sale, updateSubscription, fastAccessFunding

Attributes:
None

Child Elements:
Required: paypageRegistrationId
Optional: expDate, cardValidationNum, type

NOTE: Although the schema defines the expDate element as an optional child of the paypage element, you must submit a value for card-not-present transactions.

Example: Example: paypage Structure

```
<paypage>
  <paypageRegistrationId>Registration ID from eProtect</paypageRegistrationId>
  <expDate>Expiration Date</expDate>
  <cardValidationNum>Card Validation Number</cardValidationNum>
  <type>Method of Payment</type>
</paypage>
```
4.289 paypageRegistrationId

The `paypageRegistrationId` element is a required child of the `paypage` element, and specifies the Registration ID generated by securepaypage.litle.com (eProtect). You can also use it in a Register Token Request to obtain a token based on eProtect activity prior to submitting an Authorization or Sale transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 512

**Parent Elements:**
- `paypage`
- `registerTokenRequest`

**Attributes:**
- None

**Child Elements:**
- None
4.290 paypal

The `paypal` element defines PayPal account information. It replaces the card or token elements in transactions using PayPal as a payment method.

**Parent Elements:**
- authorization
- sale

**Attributes:**
- None

**Child Elements:**
- Required: payerId, transactionId
- Optional: token

**Example: paypal Structure**

```xml
<paypal>
  <payerId>PayPal Customer Identifier</payerId>
  <token>Token Value Returned</token>
  <transactionId>PayPal Transaction ID</transactionId>
</paypal>
```
4.291 **payPalNotes**

The `payPalNotes` element is an optional child of multiple transaction types. You use this field to record additional information about the PayPal transaction.

Type = String; Type = String; minLength = N/A; maxLength = 255

**Parent Elements:**

- authReversal
- capture
- credit
- sale

**Attributes:**

None

**Child Elements:**

None
4.292 PayPalOrderComplete

The `PayPalOrderComplete` element is an optional child of both the `capture` and `sale` elements, but is required to close a PayPal order. Set the value to `true` to close the order, when you have fulfilled the order and do not need to send any further auths or deposits against it. Set the value to `false` to keep the order open for additional auths or deposits.

Type = Boolean; Valid values = true or false

Parent Elements:
capture, sale

Attributes:
None

Child Elements:
None
4.293 phone

The `phone` element has two different uses in cnpAPI depending upon the parent element. When used as a child of either the `billToAddress` or `shipToAddress` elements, it defines the customer's phone number. When used as a child of the `customBilling` element, it defines the phone number of the merchant.

### 4.293.1 phone as a child of billToAddress and shipToAddress

The `phone` element defines the customer’s phone number in both the `billToAddress` and `shipToAddress` elements.

**NOTE:** When submitting an eCheck Verification, the string can only contain numbers (0 through 9). Letters and special characters are not allowed.

Type = String; minLength = N/A; maxLength = 20

**Parent Elements:**
- billToAddress, shipFromPostalCode

**Attributes:**
- None

**Child Elements:**
- None

### 4.293.2 phone as a child of customBilling

The `phone` element defines the merchant’s phone number. The string can only contain numbers (0 through 9). Letters and special characters are not allowed.

Type = String; minLength = N/A; maxLength = 13

**Parent Elements:**
- customBilling

**Attributes:**
- None

**Child Elements:** None
4.294 physicalCheckCredit

The `physicalCheckCredit` element is the parent element for the transaction type that a Payment Facilitator uses to distribute funds to a third party who issues physical checks on the Payment Facilitators behalf. (i.e., from the PayFac Settlement Account to the Third Party Account). If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**
batchRequest, littleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to <code>true</code> and the entire original response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

**Child Elements:**
Required: amount, fundingSubmerchantId, fundsTransferId
4.295 physicalCheckCreditResponse

The `physicalCheckCreditResponse` element is the parent element for information returned to you in response to a `physicalCheckCredit` transaction.

**Parent Elements:**
- `batchResponse`
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String | Yes       | The response returns the same value submitted in the `physicalCheckCredit` transaction.  
|                |        |           | `minLength = 1`  
|                |        |           | `maxLength = 36` |
| customerId     | String | No        | The response returns the same value submitted in the `physicalCheckCredit` transaction.  
|                |        |           | `minLength = N/A`  
|                |        |           | `maxLength = 50` |
| reportGroup    | String | Yes       | The response returns the same value submitted in the `physicalCheckCredit` transaction.  
|                |        |           | `minLength = 1`  
|                |        |           | `maxLength = 25` |

**Child Elements:**

Required: `litleTxnId`, `fundsTransferId`, `response`, `responseTime`, `message`

Required (Online): `postDate`
4.296 physicalCheckDebit

The `physicalCheckDebit` element is the parent element for the transaction type that a Payment Facilitator uses to move funds from a third party who issues physical checks on the Payment Facilitator’s behalf. (i.e., from the Third Party Account to the PayFac Settlement Account). If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

Parent Elements:
- `batchRequest`
- `litleOnlineRequest`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String     | Yes       | A unique identifier assigned by the presenter and mirrored back in the response.  
|                |            |           | `minLength = 1   maxLength = 36` |
| customerId     | String     | No        | A value assigned by the merchant to identify the consumer.  
|                |            |           | `minLength = N/A maxLength = 50` |
| reportGroup    | String     | Yes       | For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.  
|                |            |           | `minLength = 1   maxLength = 25` |
| duplicate      | boolean    | No        | If the request is a duplicate, the response includes the duplicate flag set to `true` and the entire original response.  
|                |            |           | **Note**: This attribute applies only to Online Dynamic Payout Funding Instruction responses. |

Child Elements:

Required: `amount`, `fundingSubmerchantId`, `fundsTransferId`
4.297 physicalCheckDebitResponse

The physicalCheckDebitResponse element is the parent element for information returned to you in response to a physicalCheckDebit transaction.

Parent Elements:
batchResponse, littleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the physicalCheckDebit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the physicalCheckDebit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the physicalCheckDebit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: littleTxnId, fundsTransferId, response, responseTime, message
Required (Online): postDate
The `pin` element is an optional child of the several transaction types, as well as the `card` element. It only applies to transactions involving closed-loop Gift Cards and defines the pin number associated with the Gift Card.

**Type** = String; **minLength** = 4; **maxLength** = 12

**Parent Elements:**
capture, card, credit, virtualGiftCardResponse

**Attributes:**
None

**Child Elements:**
None
4.299 planCode

The planCode element is the identifier of a defined recurring payment plan. You use it to specify the payment plan when submitting a recurring transaction to the Recurring Engine. For example, there could be a define plan called Monthly that instructs the Recurring Engine to bill the consumer the same amount every month for the number of months defined by the numberOfPayments element. This element is a required child of the subscription element.

Type = String; minLength = N/A; maxLength = 25

Parent Elements:
subscription, updateSubscription, createPlan, updatePlan, updatePlanResponse

Attributes:
None

Child Elements:
None
4.300 pos

The pos element contains child elements used to specify information required when submitting authorization, captureGivenAuth, credit, forceCapture, and sale transactions from point of sale terminals.

**Parent Elements:**
authorization, captureGivenAuth, credit, forceCapture, sale

**Attributes:**
None

**Child Elements:**
capability, entryMode, cardholderId, terminalId, catLevel

**Example:** pos Structure

```xml
<pos>
  <capability>Capability Enumeration</capability>
  <entryMode>Entry Mode Enumeration</entryMode>
  <cardholderId>Cardholder ID Enumeration</cardholderId>
  <terminalId>1234567890</terminalId>
  <catLevel>Capability of CAT Terminal</catLevel>
</pos>
```

**NOTE:** For CAT (Cardholder Activated Terminal) transactions, the **capability** element must be set to `magstripe`, the **cardholderId** element must be set to `nopin`, and the **catLevel** element must be set to `self service`. 
4.301 postDate

The `postDate` element defines the date the transaction was posted. The format is YYYY-MM-DD. It occurs only in response to Online transactions.

**NOTE:** Although the schema defines this element as optional in most response messages, the system returns it in the response for all Online transactions.

Type = Date; minLength = N/A; maxLength = 10

**Parent Elements:**
- activateResponse
- activateReversalResponse
- authorizationResponse
- authReversalResponse
- captureResponse
- captureGivenAuthResponse
- creditResponse
- deactivateResponse
- deactivateReversalResponse
- depositReversalResponse
- echeckCreditResponse
- echeckSalesResponse
- echeckVerificationResponse
- forceCaptureResponse
- loadResponse
- loadReversalResponse
- refundReversalResponse
- saleResponse
- unloadResponse
- unloadReversalResponse
- voidResponse
- fastAccessFundingResponse
- payFacCreditResponse
- payFacDebitResponse
- physicalCheckCreditResponse
- physicalCheckDebitResponse
- reserveCreditResponse
- reserveDebitResponse
- submerchantCreditResponse
- submerchantDebitResponse
- vendorCreditResponse
- vendorDebitResponse

**Attributes:**
- None

**Child Elements:**
- None
4.302 postDay

The `postDay` element is an optional child of the `accountUpdateFileRequestData` element that defines the date you submitted the Account Updater request. The format is YYYY-MM-DD.

**NOTE:** This is also the same date that the system created the Account Updater acknowledgment file.

*Type =* Date; *minLength =* N/A; *maxLength =* 10

**Parent Elements:**
- `accountUpdateFileRequestData`

**Attributes:**
- None

**Child Elements:**
- None
4.303 preapprovalNumber

The preapprovalNumber element is an optional child of the billMeLaterRequest element, which you use to specify the pre-approval number issued by PayPal Credit. If you include this element, the value must be 16 digits in length. Do not include this element to indicate there is no pre-approval. Internal pre-approval is indicated by using 1 as the first digit.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

**Type** = String;  
**minLength** = 13;  
**maxLength** = 25

**Parent Elements:**

billMeLaterRequest

**Attributes:**

None

**Child Elements:**

None
4.304 preferredLanguage

The preferredLanguage element is an optional child of the sepaDirectDebit element. This defines the language in which the merchant prefers the mandate to appear. While the merchant could be able to select any language, the mandate may not be available in the selected language. If the selected language is not available, the mandate appears in English. If you do not include this element, the preferred language defaults to the language indicated by the country of the IBAN, unless it is not available, in which case the language defaults to English.

Type = String (Enum); minLength = N/A; maxLength = 3

NOTE: The enumerations for this element are listed under <countryTypeEnum> in the cnpAPI Common XSD. The country names corresponding to the abbreviations can be found in the ISO 3166-1 standard.

All Country Codes are 2-character except for the United States, which accepts both US and USA.

Parent Elements:
sepaDirectDebit, ideal, giropay

Attributes:
None

Child Elements:
None
4.305 prepaid

The `prepaid` element is an optional child of the `filtering` element. How you choose to implement the Prepaid Filtering feature determines the use of the `prepaid` element. If your configuration filters all prepaid card transactions, you can disable the feature on selected transactions by including the `prepaid` element with a setting of `false`. If your configuration filters prepaid card transactions on a per transaction basis, you enable the filtering on a selected transaction by including the `prepaid` element with a setting of `true`.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
- `filtering`

**Attributes:**
- None

**Child Elements:**
- None
4.306 prepaidCardType

The `prepaidCardType` element is an optional child of the `enhancedAuthResponse` element, which specifies the type of prepaid card submitted in the Authorization or Sale transaction. For example, a few of the possible values are: GIFT, PAYROLL, and GENERAL_PREPAID.

Type = String; minLength = N/A; maxLength = 50

Parent Elements:

- fundingSource

Attributes:

None

Child Elements:

None
4.307 processingInstructions

The `processingInstructions` element contains a child element that allows you to specify whether or not the system performs velocity checking on the transaction.

**Parent Elements:** (optional for all)
- `authorization`
- `capture`
- `captureGivenAuth`
- `credit`
- `forceCapture`
- `sale`
- `void`

**Attributes:**
None

**Child Elements:**
- `bypassVelocityCheck`

**NOTE:** Please consult your Relationship Manager for additional information concerning Velocity Checking.

**Example:** processingInstructions Structure

```xml
<processingInstructions>
  <bypassVelocityCheck>true or false</bypassVelocityCheck>
</processingInstructions>
```
4.308 processingType

The processingType element is an optional child of several transaction types. You use this element to define a Visa transaction is intended to fund a host-based prepaid product, a brokerage account, or an escrow account. For Visa and Discover transactions, you also use it to define the initial transaction in a recurring or installment stream, or the initial, merchant initiated, or cardholder initiated Card on file transaction. For these cases, you must store the value of the networkTransactionId element returned in the response message. You use this value to populate the originalNetworkTransactionId element in subsequent recurring/installment or CoF transactions, so the networks can tie back to the original approval.

Type = String (Enum); minLength = N/A; maxLength = N/A

Parent Elements:
authorization, captureGivenAuth, forceCapture, sale

Attributes:
None

Child Elements:
None

Enumerations:

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accountFunding</td>
<td>Use this value to define a Visa transaction is intended to fund a host-based prepaid product, a brokerage account, or an escrow account.</td>
</tr>
<tr>
<td>initialRecurring</td>
<td>Use this value to specify the first in a series of recurring payment transactions.</td>
</tr>
<tr>
<td>initialInstallment</td>
<td>Use this value to specify the first in a series of installment payment transactions.</td>
</tr>
<tr>
<td>initialCOF</td>
<td>Use this value to specify an initial Card on File transaction.</td>
</tr>
<tr>
<td>merchantInitiatedCOF</td>
<td>Use this value to specify a merchant initiated Card on File transaction after the initial CoF transaction.</td>
</tr>
<tr>
<td>cardholderInitiatedCOF</td>
<td>Use this value to specify a cardholder initiated Card on File transaction after the initial CoF transaction.</td>
</tr>
</tbody>
</table>
4.309 productCode

The productCode element is an optional child of the lineItemData element, which specifies the product code of the purchased item. This value is a merchant defined description code of the product/service. This could be an inventory number, UPC, catalog number, or some other value that the merchant uses to define the specific product. Although an optional element, it is required by Visa and MasterCard when specifying line item data.

**Type** = String; **minLength** = 1; **maxLength** = 12

**Parent Elements:**

lineItemData

**Attributes:**

None

**Child Elements:**

None
4.310 publicKeyHash

The `publicKeyHash` element is a required child of the `header` element and provides the BASE64 Encoded string that is a hash of the merchant's certificate public key bytes associated with the Apple Pay transaction.

**Type** = Base 64 Encoded String; **minLength** = N/A; **maxLength** = 200

**Parent Elements:**
- header

**Attributes:**
- None

**Child Elements:**
- None
4.311 quantity

The quantity element is an optional child of the lineItemData element, which specifies the number of items purchased. Although an optional element, it is required by Visa and MasterCard when specifying line item data. The value must be greater than zero, but no more than 12 digits not including the decimal point.

**NOTE:** If you accidentally omit the quantity element, our system will submit the transaction using a default value of 1.

Type = Decimal; minInclusive = 0; totalDigits = 12

Parent Elements:

lineItemData

Attributes:

None

Child Elements:

None
4.312 queryTransaction

The `queryTransaction` element is the parent element for Query transactions. You use this transaction type to determine the status of a previously submitted transaction. You can submit this element only as an Online transaction.

**Parent Elements:**
- `litleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
</tbody>
</table>

**Child Elements:**

- Required: `origId`, `origActionType`
- Optional: `origLitleTxnId`, `origOrderId`, `origAccountNumber`
4.313 queryTransactionResponse

The queryTransactionResponse element is the parent element for the response to queryTransaction requests.

**Parent Elements:**
littleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1      maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A     maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the capture transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1      maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

All Required: response, responseTime, message, matchCount, results_Max10
**4.314 queryTransactionUnavailableResponse**

The `queryTransactionUnavailableResponse` element is an optional child of the `results_Max10` element. If the query results is a response code of 152 - Transaction found, but response not yet available, the `results_Max10` element will contain at least one `queryTransactionUnavailableResponse` child (see example `results_Max10 Element for 152 Response Code`).

**Parent Elements:**

`results_Max10`

**Attributes:**

None

**Child Elements:**

All Required: `litleTxnId`, `response`, `message`
4.315 recurringRequest

The recurringRequest element is the parent of several child elements that define the number of payments and plan type of recurring transaction to be handled by the Recurring Engine. It is an optional child of the Authorization and Sale transactions.

Parent Elements:
authorization, sale

Attributes:
None

Child Elements:
subscription

Example: recurringRequest Structure

```xml
<recurringRequest>
  <subscription>
    <planCode>Plan Reference Code</planCode>
    <numberOfPayments>1 to 99</numberOfPayments>
    <startDate>Start Date of Recurring Cycle</startDate>
    <amount>Amount of Recurring Payment</amount>
  </subscription>
</recurringRequest>
```
4.316 recurringResponse

The recurringResponse element is the parent element for the subscriptionId, responseCode, responseMessage, and recurringTxnId elements associated with a requested recurring payment. The system returns this element only when the sale transaction includes a recurringRequest element.

Parent Elements:
authorizationResponse, saleResponse

Attributes:
None

Child Elements:
subscriptionId, responseCode, responseMessage, recurringTxnId

Example: recurringResponse Structure

```xml
<recurringResponse>
  <subscriptionId>1234567890</subscriptionId>
  <responseMessage>Response Message</responseMessage>
</recurringResponse>
```
4.317 recurringTxnId

The `recurringTxnId` element is an optional child of the `recurringResponse` element used to identify the record of recurring, scheduled transactions.

**Type** = Long; **minLength** = N/A; **maxLength** = 19

**Parent Elements:**
- recurringResponse, litleInternalRecurringRequest

**NOTE:** Although the element is an optional child of the `recurringResponse` element, it will never be returned in the response to a merchant initiated transaction.

**Attributes:**
- None

**Child Elements:**
- None
4.318 recycleAdvice

**NOTE:** Worldpay no longer supports the Recycling Advice feature.

The `recycleAdvice` element contains a two child elements that either specifies the date and time (in GMT) recommended for the next recycle of the declined Authorization/Sale transaction or indicates that there is no additional recycling advice. The two children are mutually exclusive.

**Parent Elements:** (optional for all)
- recycling

**Attributes:**
- None

**Child Elements:**
- `nextRecycleTime`, `recycleAdviceEnd`

**NOTE:** The `recycleAdvice` element contains either a `nextRecycleTime` or `recycleAdviceEnd` element, but not both.

**Example:** `recycleAdvice` Structure - with recommended Date:Time

```xml
<recycleAdvice>
  <nextRecycleTime>2016-11-15T12:00:00</nextRecycleTime>
</recycleAdvice>
```

**Example:** `recycleAdvice` Structure - with end message

```xml
<recycleAdvice>
  <recycleAdviceEnd>End of Advice</recycleAdviceEnd>
</recycleAdvice>
```
4.319 recycleAdviceEnd

**NOTE:** Worldpay no longer supports the Recycling Advice feature.

The `recycleAdviceEnd` element is an optional child of the `recycleAdvice` element and signifies that no further recycling recommendations are available.

*Type* = String; *minLength* = N/A; *maxLength* = 20

**Parent Elements:**
recycleAdvice

**Attributes:**
None

**Child Elements:**
None
4.320 recycleBy

The recycleBy element is an optional child of the recyclingRequest element and determines the use of the Recycling Engine. The default setting is Litle, so omitting this element is the same as submitting a value of Litle.

NOTE: Also, if your Merchant Profile includes a preset percentage split of transactions between merchant and Worldpay controlled, then settings of Merchant and Litle are ignored; you can still use a setting of None to exclude the transaction.

Also, although the default setting is normally Litle, it can be altered in your merchant profile to a setting of Merchant.

Type = String (Enum); minLength = N/A; maxLength = N/A

Parent Elements:
recyclingRequest

Attributes:
None

Child Elements:
None

Enumerations:

<table>
<thead>
<tr>
<th>Enum</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merchant</td>
<td>This setting indicates that the merchant controls the recycling of the transaction. For A/B comparison testing, transactions using this setting will be counted as merchant controlled. After setting this value in the initial transaction, subsequent transactions should have same value. Any different value will be ignored.</td>
</tr>
<tr>
<td>Litle</td>
<td>This setting indicates either that the Recycling Engine controls the recycling of the transaction. For A/B comparison testing, transactions using this setting will be counted as Worldpay controlled. After setting this value in the initial transaction, subsequent transactions should have same value. Any different value will be ignored.</td>
</tr>
<tr>
<td>None</td>
<td>For A/B comparison testing, transactions using this setting are excluded from all counts. These transactions will not be counted as either merchant or Worldpay controlled.</td>
</tr>
</tbody>
</table>
4.321 recycleEngineActive

The recycleEngineActive element is an optional child of the recycling element that indicates whether or not the engine is recycling the declined transaction. This element is returned only if you are using the Recycling Engine.

Type = Boolean; Valid values = true or false

Parent Elements:
recycling

Attributes:
None

Child Elements:
None
4.322 recycleId

The recycleId element is an optional child of the recyclingRequest element. Merchants can use this identifier as part of the transaction signature used to track the recycling of a transaction. This element is an alternative to using the orderId element as part of the transaction signature.

**Type** = String; **minLength** = N/A; **maxLength** = 25

**Parent Elements:**

recyclingRequest

**Attributes:**

None

**Child Elements:**

None
4.323 recycling

The recycling element has two uses in the cnpAPI depending upon the parent. When used as a child of either the authorizationResponse or saleResponse elements, the recycling element contains a child element that specifies whether or not the engine is recycling the declined transaction.

When used as a child of the voidResponse, the recycling element contains a child providing the Transaction Id of the Credit transaction issued. This occurs only if a Void transaction is used to halt the recycling of a transaction by the recycling engine and the transaction has already been approved and captured (see Using Void to Halt Recycling Engine on page 77).

4.323.1 recycling Element as a Child of authorizationResponse or saleResponse

The recycling element contains child elements indicating that the Recycling Engine is active.

Parent Elements:
authorizationResponse, saleResponse

Attributes:
None

Child Elements:
recycleAdvice, recycleEngineActive

NOTE: Worldpay no longer supports the Recycling Advice feature.

Example: recycling Structure - with engine active flag

<recycling>
  <recycleEngineActive>true or false</recycleEngineActive>
</recycling>

4.323.2 recycling Element as a Child of voidResponse

The recycling element in an optional child of the voidResponse element and contains a child providing the Transaction Id of the Credit transaction issued. This element is present in the Void response only under the following circumstances (see Using Void to Halt Recycling Engine on page 77):

- You submitted a Void transaction to halt the recycling of a declined Sale transaction by the Recovery/Recycling Engine.
- The Sale transaction has already been approved and captured.
- Your Recovery/Recycling Engine configuration enables automatic refunds.
- The system has successfully submitted a Credit transaction on your behalf.
Parent Elements:

voidResponse

Attributes:

None

Child Elements:

creditLitleTxnId

Example: recycling Structure - child of voidResponse

<recycling>
    <creditLitleTxnId>1234567890123456789</creditLitleTxnId>
</recycling>
4.324 recyclingRequest

The recyclingRequest element is an optional child of the authorization and sale transactions, which contains a child element that specifies who is responsible for recycling the transaction. It also contains an optional element for an identifier assigned by the merchant to track the recycling of the transaction. This element only applies to merchants using the Recycling Engine.

Parent Elements:
authorization, sale

Attributes:
None

Child Elements:
recycleBy, recycleId

Example: recyclingRequest Structure

```xml
<recyclingRequest>
  <recycleBy>Merchant or Litle or None</recycleBy>
  <recycleId>abcdef1234567890</recycleId>
</recyclingRequest>
```
4.325 redirectToken

The redirectToken element is an optional child of the sepaDirectDebitResponse, idealResponse, and giropayResponse elements. It allows you to verify that the consumer accepted the Mandate by comparing it to the token_value parameter returned in the URL upon redirect from the Mandate site.

Type = String; minLength = N/A; maxLength = 128

Parent Elements:
sepDirectDebitResponse, idealResponse, giropayResponse

Attributes:
None

Child Elements:
None
4.326 redirectUrl

The redirectUrl element is an optional child of the sepaDirectDebit element and defines the URL that hosts the mandate, when Worldpay supplies the mandate. If you supply the mandate (<mandateProvider>Merchant</mandateProvider>), this element will not appear in the response.

**Type** = String; **minLength** = N/A; **maxLength** = 256

**Parent Elements:**

- sepaDirectDebitResponse
- idealResponse
- giropayResponse

**Attributes:**

None

**Child Elements:**

None
4.327 refCode

The `refCode` element is an optional child of the `giftCardResponse` element, where it specifies the authorization code of the gift card transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 6

**Parent Elements:**
- `giftCardResponse`

**Attributes:**
- None

**Child Elements:**
- None
4.328 refundReversal

The refundReversal element is the parent element for a Gift Card specific transaction type that reverses the a refund associated with a Gift Card.

Parent Elements:
- litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1   maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A   maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1   maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (Required)
- card, originalRefCode, originalAmount, originalTxnTime, originalSystemTraceId, originalSequenceNumber

Child Elements: (Optional)
- litleTxnId
### 4.329 refundReversalResponse

The `refundReversalResponse` element is the parent element for information returned to you in response to an `refundReversal` transaction.

#### Parent Elements:
- `litleOnlineResponse`

#### Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Refund Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Refund Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Refund Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1    maxLength = 25</code></td>
</tr>
</tbody>
</table>

#### Child Elements:

- Required: `litleTxnId`, `response`, `responseTime`, `message`, `giftCardResponse`
- Optional: `postDate`
4.330 registerTokenRequest

The registerTokenRequest element is the parent element for the Register Token transaction. You use this transaction type when you wish to submit an account number for tokenization, but there is no associated payment transaction.

You can use this element in either Online or Batch transactions.

NOTE: When submitting registerTokenRequest elements in a batchRequest, you must also include a numTokenRegistrations= attribute in the batchRequest element.

Parent Elements:
litleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute defining the merchant sub-group in the user interface where this transaction displays. Also see Coding for Report Groups on page 10 for information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: either accountNumber, echeckForToken, mpos, paypageRegistrationId, or applepay

Optional: orderld, cardValidationNum

NOTE: The use of the cardValidationNum element in the registerTokenRequest only applies when you submit an accountNumber element.
4.331 registerTokenResponse

The registerTokenResponse element is the parent element for the response to registerTokenRequest transactions. You receive this transaction type in response to the submission of an account number for tokenization in a registerTokenRequest transaction.

Parent Elements:
litleOnlineResponse, batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the registerTokenRequest transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the registerTokenRequest transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the registerTokenRequest transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: litleTxnId, response, message, responseTime

Optional: eCheckAccountSuffix, litleToken, bin, type, applepayResponse, androidpayResponse
4.332 reloadable

The `reloadable` element is an optional child of the `fundingSource` element and defines whether the prepaid card is reloadable.

**NOTE:** This element is never returned for American Express card transaction.

Type = String (Enum); Enumerations = YES, NO, or UNKNOWN

Parent Elements:

`fundingSource`

Attributes:

None

Child Elements:

None
**4.333 reserveCredit**

The *reserveCredit* element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the PayFac Settlement Account to the PayFac Reserve Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**
- batchRequest, litleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength = 1   maxLength = 36</strong></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength = N/A   maxLength = 50</strong></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>minLength = 1   maxLength = 25</strong></td>
</tr>
</tbody>
</table>

**Child Elements:**

- Required: amount, fundingSubmerchantId, fundsTransferId
4.334 reserveCreditResponse

The `reserveCreditResponse` element is the parent element for information returned to you in response to a `reserveCredit` transaction.

**Parent Elements:**
- `batchResponse`
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>payFacCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the <code>payFacCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the <code>reserveCredit</code> transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to <code>true</code> and the entire original response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `fundsTransferId`, `response`, `responseTime`, `message`

Required (Online): `postDate`
4.335 reserveDebit

The `reserveDebit` element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the PayFac Reserve Account to the PayFac Settlement Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**
batchRequest, littleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports. minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**
Required: amount, fundingSubmerchantId, fundsTransferId
4.336 reserveDebitResponse

The reserveDebitResponse element is the parent element for information returned to you in response to a reserveDebit transaction.

Parent Elements:
batchResponse, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the payFacCredit transaction. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the reserveDebit transaction. minLength = 1  maxLength = 25</td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to true and the entire original response.  Note: This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

Child Elements:
Required: litleTxnId, fundsTransferId, response, responseTime, message
Required (Online): postDate
4.337 residenceStatus

The `residenceStatus` element is an optional child of the `customerInfo` element and defines the type of domicile in which the customer resides. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = String (Enum); Enumerations = Own, Rent, or Other

**Parent Elements:**

customerInfo

**Attributes:**

None

**Child Elements:**

None
4.338 response

The `response` element contains a three digit numeric code which specifies either that the transaction is approved (000 code) or declined. The `message` element provides a brief definition of the response code.

For a complete list of response codes and associated messages, please refer to Payment Transaction Response Codes on page 786.

Type = String; `minLength` = N/A; `maxLength` = 3

Parent Elements:

Attributes:
None

Child Elements:
None
4.339 responseCode

The responseCode element contains a three digit numeric code which along with the responseMessage element specifies either acceptance by the Recurring Engine or the reason the recurring Engine was unable to schedule subsequent payments.

Type = String; minLength = N/A; maxLength = 3

Parent Elements:
recurringResponse

Attributes:
None

Child Elements:
None
4.340 responseMessage

The `responseMessage` element contains a brief definition of the `responseCode` returned for the recurring transaction.

Type = String; minLength = N/A; maxLength = 512

Parent Elements:
`recurringResponse`

Attributes:
None

Child Elements:
None
### 4.341 responseTime

The `responseTime` element provides a date/time stamp of the response. The format of the element is YYYY-MM-DDTHH:MM:SS. For example, 2009-12-21T11:37:04.

**Type** = `dateTime`; **minLength** = N/A; **maxLength** = 19

**Parent Elements:**


**Attributes:**

None

**Child Elements:**

None
4.342 results_Max10

The results_Max10 element is a required child of the queryTransactionResponse. Any original transaction responses that match the criteria submitted in the queryTransaction appear as children of this element in the response. The system can return a maximum of ten responses as children of the results_Max10 element. The value for the matchCount element reflects the number of found transactions.

If the system does not find any transactions matching the query criteria, the results_Max10 element will be empty.

If the query results is a response code of 152 - Transaction found, but response not yet available, the results_Max10 element will contain at least one queryTransactionUnavailableResponse child and may contain other found responses.

Parent Elements:
queryTransactionResponse

Attributes:
None

Child Elements:

Example: results_Max10 Element with One Found Transaction

```xml
<results_max10>
  <authorizationResponse id="GiftCardAuth" reportGroup="Mer5PM1" customerId="1">
    <litleTxnId>82827170811986124</litleTxnId>
    <orderId>150330_GCAuth</orderId>
    <response>000</response>
    <responseTime>2015-04-06T16:40:04</responseTime>
    <postDate>2015-04-06</postDate>
    <message>Approved</message>
    <authCode>111115</authCode>
    <fraudResult>
      <avsResult>30</avsResult>
      <cardValidationResult>M</cardValidationResult>
    </fraudResult>
    <giftCardResponse>
```

<availableBalance>125</availableBalance>
</giftCardResponse>
</authorizationResponse>
</results_max10>

Example: results_max10 Element with Multiple Found Transactions

<results_max10>
<authorizationResponse id="DupeId" reportGroup="Mer5PM1">
<litleTxnId>82827170811986215</litleTxnId>
<orderId>150331_DupeAuth2</orderId>
<response>000</response>
<responseTime>2015-04-06T16:40:12</responseTime>
<postDate>2015-04-06</postDate>
<message>Approved</message>
<authCode>055858</authCode>
<fraudResult>
<avsResult>32</avsResult>
<cardValidationResult>M</cardValidationResult>
</fraudResult>
</authorizationResponse>
<authorizationResponse id="DupeId" reportGroup="Mer5PM1">
<litleTxnId>82827170811986207</litleTxnId>
<orderId>150331_DupeAuth1</orderId>
<response>000</response>
<postDate>2015-04-06</postDate>
<message>Approved</message>
<authCode>111111</authCode>
<fraudResult>
<avsResult>00</avsResult>
<cardValidationResult>M</cardValidationResult>
</fraudResult>
</authorizationResponse>
</results_max10>

Example: results_max10 Element for 152 Response Code

<results_max10>
<queryTransactionUnavailableResponse>
  <litleTxnId>82827170811986124</litleTxnId>
  <response>152</response>
  <message>Original transaction found but response not yet available</message>
</queryTransactionUnavailableResponse>
4.343 RFRRequest

The `RFRRequest` element is an optional child of a `litleRequest` element. You can use this type of request in one of two ways.

- To request a session response from a previously processed `litleRequest`, include the `litleSessionId` child. The resulting RFR response will duplicate the original session response (Authorization, Credit, Capture, or Sale response) associated with the `litleSessionId`. The session ID returned in the response will be the session ID of the original session.
- To request an Account Updater completion response file, include the `accountUpdateFileRequestData` element. If the completion file is ready, it is returned. If the completion file is not ready, you receive an RFRResponse message with the response attribute set to 1 and the message attribute reading, “The account Update file is not ready yet. Please try again later.”

**Parent Elements:**
- `litleRequest`

**Attributes:**
- None

**Child Elements:** (Choice of)
- `litleSessionId` or `accountUpdateFileRequestData`

**Example:** RFRRequest Structure - Batch

```xml
<RFRRequest>
  <litleSessionId>Session ID</litleSessionId>
</RFRRequest>
```

**Example:** RFRRequest Structure - Account Updater

```xml
<RFRRequest>
  <accountUpdateFileRequestData>
    <merchantId>Merchant ID</merchantId>
    <postDay>Post Date</postDay>
  </accountUpdateFileRequestData>
</RFRRequest>
```
4.344 RFRResponse

The RFRResponse element is an optional child of a littleResponse element returned in response to a RFRRequest.

Parent Elements:
littleResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>response</td>
<td>String</td>
<td>Yes</td>
<td>The RFR Response Code indicating the result of the RFR request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 3</td>
</tr>
<tr>
<td>message</td>
<td>String</td>
<td>Yes</td>
<td>A brief definition of the response code returned for this transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 512</td>
</tr>
</tbody>
</table>

Child Elements:
None
4.345 routingNum

The routingNum element is a required child of the echeck, originalAccountInfo, and newAccountInfo elements defining the routing number of the Echeck account.

**Type** = String; **minLength** = 8; **maxLength** = 9

**Parent Elements:**
echeck, newAccountInfo, originalAccountInfo, newTokenInfo, originalTokenInfo, accountInfo

**Attributes:**
None

**Child Elements:**
None

**NOTE:** If you submit an invalid routing number, we return the XML Response Code 900 - Invalid Bank Routing Number.
4.346 RxAmount

The RxAmount element is an optional child of the healthcareAmounts element and defines the healthcare amount used for the purchased medications. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**

Optional: healthcareAmounts

**Attributes:**

None

**Child Elements:**

None
**4.347 sale**

The `sale` element is the parent element for all Sale transactions. A Sale transaction is a combination of Authorization and Capture transaction. You can use this element in either Online or Batch transactions.

**Parent Elements:**
- litleOnlineRequest, batchRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1  maxLength = 25</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `orderId`, `amount`, `orderSource`, (choice of) `card`, `paypal`, `paypage`, `mpos`, `token`, `applepay`, `ideal`, `sepaDirectDebit`, `giropay`, or `sofort`

**NOTE:** The cardholderAuthentication child element is required only for 3-D Secure transactions and for BML ecommerce transactions.

The `fraudCheck` element has been deprecated; use the `cardholderAuthentication` element instead.

Optional: `litleTxnId`, `customerInfo`, `billToAddress`, `shipToAddress`, `billMeLaterRequest`, `cardholderAuthentication`, `customBilling`, `taxType`, `enhancedData`, `processingInstructions`, `pos`, `paypalNotes`, `paypalOrderComplete`, `amexAggregatorData`, `allowPartialAuth`, `healthcareIIAS`, `merchantData`, `recyclingRequest`, `fraudFilterOverride`, `secondaryAmount`, `surchargeAmount`, `recurringRequest`, `litleInternalRecurringRequest`, `debtRepayment`, `advancedFraudChecks`, `wallet`, `processingType`, `origOrderId`, `password`, `originalNetworkTransactionId`, `originalTransactionAmount`
4.348 saleResponse

The `saleResponse` element is the parent element for information returned in response to a Sale transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

**Parent Elements:**

`litleOnlineResponse`, `batchResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Sale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Sale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A  maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Sale transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1  maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `orderId`, `response`, `responseTime`, `message`

Optional: `postDate`, `cardProductId` (see Note below), `authCode`, `authorizationResponseSubCode` (see Note below), `approvedAmount`, `accountInformation`, `fraudResult`, `billMeLaterResponseData`, `tokenResponse`, `enhancedAuthResponse`, `accountUpdater`, `recycling`, `recurringResponse`, `giftCardResponse`, `applepayResponse`, `cardSuffix`, `androidpayResponse`, `sepaDirectDebitResponse`, `networkTransactionId`, `idealResponse`, `giropayResponse`, `sofortResponse`

**NOTE:** The system returns the `postDate` child element only in responses to Online transactions.

The `cardProductId` element returns a raw code referencing the card type. Please consult your Relationship Manager for additional information.

The `authorizationResponseSubCode` element is not used at this time.
**4.349 salesTax**

The `salesTax` element defines the amount of sales tax included in the transaction amount. Although the schema defines it as an optional child of the `enhancedData` element, it is required to receive the best interchange rate for Level II and Level III corporate purchases. The decimal is implied. Example: 500 = $5.00.

**NOTE:** For a non-taxable transaction, use 0 as the value. In this case you must also set the `taxExempt` element to true.

If you provide `detailTax` data, the `salesTax` should be the sum of the `detailTax`.

Type = Integer; totalDigits = 8

Parent Elements:
- `enhancedData`

Attributes:
- None

Child Elements:
- None
4.350 secondaryAmount

The `secondaryAmount` element defines the principal portion of the total amount when a convenience fee applied to the transaction by the merchant. For example, if the total charge is $105, with the principal amount being $100 and the convenience fee being $5, you must use $100 as the value for the `secondaryAmount` element. Supply the value in cents without a decimal point. For example, a value of 400 signifies $4.00.

**NOTE:** Worldpay supports convenience fees for MasterCard and Visa card brands only. Attempting to use this element in a Discover or American Express transaction results in a decline with the response code of 381 - This method of payment does not support secondary amount.

**Type** = Integer; **totalDigits** = 12

**Parent Elements:**
- authorization
- capture
- credit
- captureGivenAuth
- echeckCredit
- echeckSale
- forceCapture
- sale

**Attributes:**
None

**Child Elements:**
None
4.351 sellerId

The `sellerId` element is a required child of the `amexAggregatorData` element, which defines the Seller Id as assigned by American Express.

**Type** = String; **minLength** = 1; **maxLength** = 16

**Parent Elements:**
- `amexAggregatorData`

**Attributes:**
None

**Child Elements:**
None
4.352 sellerMerchantCategoryCode

The sellerMerchantCategoryCode element is a required child of the amexAggregatorData element, which defines the Merchant Category Code as assigned by American Express.

Type = String; minLength = 1; maxLength = 4

Parent Elements:

amexAggregatorData

Attributes:
None

Child Elements:
None
4.353 sepaDirectDebit

The sepaDirectDebit element is a child of the sale transaction that, through its child elements, defines information needed to process a SEPA Direct Debit transaction. At this time, you can use the SEPA Direct Debit method of payment in Online transactions only.

Parent Elements:

sale

Attributes:

None

Child Elements:

Required: iban, mandateProvider, sequenceType
Optional: mandateReference, mandateURL, mandateSignatureDate, preferredLanguage

Example: sepaDirectDebit Structure

```xml
<sepaDirectDebit>
  <mandateProvider>Merchant or Vantiv</mandateProvider>
  <sequenceType>One Time or Type of Recurring</sequenceType>
  <mandateReference>Ref to First Payment of Recurring Stream</mandateReference>
  <mandateUrl>URL of Merchant Hosted Mandtae</mandateUrl>
  <mandateSignatureDate>2016-06-12</mandateSignatureDate>
  <iban>Consumer Account Number</iban>
  <preferredLanguage>Country Code of Language</preferredLanguage>
</sepaDirectDebit>
```
4.354 sepaDirectDebitResponse

The `sepaDirectDebitResponse` element is a child of the `saleResponse` transaction when the method of payment in the request is `sepaDirectDebit`. It contains child elements that you should store for future use/reference.

**Parent Elements:**

- `saleResponse`

**Attributes:**

None

**Child Elements (all Optional):**

- `mandateReference`
- `redirectUrl`
- `redirectToken`

**Example: sepaDirectDebitResponse Structure**

```xml
<sepaDirectDebitResponse>
  <redirectUrl>URL of Worldpay Supplied Mandate</redirectUrl>
  <redirectToken>Map Mandate to litleTxnId</redirectToken>
  <mandateReference>Ref Number for Subsequent Payments</mandateReference>
</sepaDirectDebitResponse>
```
4.355 sequenceNumber

The `sequenceNumber` element is an optional child of the `giftCardResponse` element, which specifies a Worldpay generated sequence number associated with the transaction in our systems. You should retain this value for possible future use in gift card reversal transactions.

**Type** = String; **totalDigits** = 6; **Allowed Characters**: 0 - 9

**Parent Elements:**
giftCardResponse

**Attributes:**
None

**Child Elements:**
None
4.356 sequenceType

The `sequenceType` element is a required child of the `sepaDirectDebit` element and defines the purchase in terms of a one-time buy or a member of a recurring stream of debits. If the value of this element is either `SubsequentRecurring`, or `FinalRecurring`, you must include the mandateReference element in the request, specifying the value returned in the initial `sepaDirectDebitResponse`.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
- `sepaDirectDebit`

**Attributes:**
- None

**Child Elements:**
- None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OneTime</td>
<td>The purchase is a one-time buy (i.e., not part of a recurring stream).</td>
</tr>
<tr>
<td>FirstRecurring</td>
<td>This purchase is the initial buy of a recurring stream.</td>
</tr>
<tr>
<td>SubsequentRecurring</td>
<td>This purchase is part of a recurring stream of purchases, but not the first or last in the stream.</td>
</tr>
<tr>
<td>FinalRecurring</td>
<td>This purchase is the final buy of a recurring stream.</td>
</tr>
</tbody>
</table>
4.357 shipFromPostalCode

The `shipFromPostalCode` element defines the postal code from which the product ships in the `enhancedData` element.

**Type** = String;  **minLength** = N/A;  **maxLength** = 20

**NOTE:** Although the schema specifies the `maxLength` of the `<shipFromPostalCode>` element as 20 characters, in practice you should never exceed 10 characters in your submissions.

**Parent Elements:**
- `enhancedData`

**Attributes:**
- None

**Child Elements:**
- None
4.358 shippingAmount

The `shippingAmount` element defines shipping cost for the order. Although the schema defines it as an optional child of the `enhancedData` element, it is required by Visa for Level III interchange rates. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**

`enhancedData`

**Attributes:**

None

**Child Elements:**

None
4.359 shipToAddress

The `shipToAddress` element contains several child elements that define the postal mailing address (and telephone number) used for shipping purposes.

**Parent Elements:**
authorization, captureGivenAuth, fraudCheck, sale

**Attributes:**
None

**Child Elements: (all Optional)**
name, addressLine1, addressLine2, addressLine3, city, state, zip, country, email, phone

**Example: shipToAddress Structure**

```xml
<shipToAddress>
  <name>Customer’s Full Name</name>
  <addressLine1>Address Line 1</addressLine1>
  <addressLine2>Address Line 2</addressLine2>
  <addressLine3>Address Line 3</addressLine3>
  <city>City</city>
  <state>State Abbreviation</state>
  <zip>ZIP Code</zip>
  <country>Country Code</country>
  <email>Email Address</email>
  <phone>Telephone Number</phone>
</shipToAddress>
```
4.360 signature

The signature element is a required child of the applepay element. It is the BASE64 encoded string signature of the payment and header data from the PKPaymentToken.

Type = String; minLength = N/A; maxLength = 10000

Parent Elements:
applepay

Attributes:
None

Child Elements:
None
The **sofort** element is a child of the **sale** transaction that, through its child elements, defines information needed to process an SOFORT (Real-time Bank Transfer) transaction. At this time, you can use the iDeal method of payment in Online transactions only.

**Parent Elements:**

**sale**

**Attributes:**

None

**Child Elements:**

Optional: **preferredLanguage**

**Example:** sofort Structure

```xml
<sofort>
  <preferredLanguage>Country Code of Language</preferredLanguage>
</sofort>
```
4.362 sofortResponse

The sofortResponse element is a child of the saleResponse transaction when the method of payment in the request is sofort. It contains child elements that you should store for future use/reference.

Parent Elements:

saleResponse

Attributes:

None

Child Elements (all Optional):

paymentPurpose, redirectUrl, redirectToken

Example: sofortResponse Structure

```xml
<sofortResponse>
  <redirectUrl>URL of Worldpay Supplied Mandate</redirectUrl>
  <redirectToken>Map Mandate to littleTxnId</redirectToken>
  <paymentPurpose>Reference Number + Billing Descriptor</paymentPurpose>
</sofortResponse>
```
The ssn element is an optional child of the customerInfo element. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = Pattern; **minLength** = 4 (last four digits of SSN); **maxLength** = 9 (full SSN)

**NOTE:** In order for a BML transaction to succeed, you must include this element if:

- the customer does not have a BML account
- the customer has a BML account, but the account has not been authenticated.

You do not need to include this element if the BML account has been authenticated.

**Parent Elements:**
customerInfo

**Attributes:**
None

**Child Elements:**
None
4.364 startDate

The `startDate` element is a optional child of the `subscription` element, which specifies the date the recurring billing should begin. It is also an optional child of both the `createAddOn` and `createDiscount` element, where it specifies either the starting date of the Add On charge or the starting date of the discount.

**Type** = Date; **Format** = YYYY-MM-DD

**Parent Elements:**
- subscription
- createAddOn
- createDiscount
- updateAddOn
- updateDiscount

**Attributes:**
None

**Child Elements:**
None
4.365 state

The state element defines the customer’s state name in the billToAddress, shipToAddress, taxBilling and elements.

Type = String; minLength = N/A; maxLength = 2

NOTE: Although the schema defines the maxLength for this element as 30, the best practice is to use the 2 character abbreviation. When submitting an eCheck Verification transaction, you must use the 2 character abbreviation or the transaction will be rejected with a 370 reason code.

Parent Elements:
billToAddress, shipToAddress, taxExempt

Attributes:
None

Child Elements:
None
4.366 submerchantCredit

The submerchantCredit element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the PayFac Settlement Account to the Sub-merchant Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

Parent Elements:
- batchRequest
- litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
- required: accountInfo, amount, fundingSubmerchantId, fundsTransferId, submerchantName, customIdentifier
4.367 submerchantCreditResponse

The submerchantCreditResponse element is the parent element for information returned to you in response to a submerchantCredit transaction.

Parent Elements:
batchResponse, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction. minLength = 1  maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the payFacCredit transaction. minLength = N/A  maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the submerchantCredit transaction. minLength = 1  maxLength = 25</td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to true and the entire original response. Note: This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

Child Elements:
Required: litleTxnId, fundsTransferId, response, responseTime, message
Required (Online): postDate
4.368 submerchantDebit

The submerchantDebit element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the Sub-merchant Account to the PayFac Settlement Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

Parent Elements:
- batchRequest, litleOnlineRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
- Required: accountInfo, amount, fundingSubmerchantId, fundsTransferId, submerchantName, customIdentifier
### 4.369 submerchantDebitResponse

The `submerchantDebitResponse` element is the parent element for information returned to you in response to a `submerchantDebit` transaction.

**Parent Elements:**
- batchResponse
- litleOnlineResponse

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the payFacCredit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the submerchantDebit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
<tr>
<td>duplicate</td>
<td>boolean</td>
<td>No</td>
<td>If the request is a duplicate, the response includes the duplicate flag set to true and the entire original response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Note:</strong> This attribute applies only to Online Dynamic Payout Funding Instruction responses.</td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: litleTxnId, fundsTransferId, response, responseTime, message

Required (Online): postDate
**4.370 submerchantName**

The `submerchantName` element is a required child of the `submerchantCredit` element and specifies the name of the Sub-merchant.

**Type** = String; **minLength** = 1; **maxLength** = 256

**Parent Elements:**

`submerchantCredit`, `submerchantDebit`, `fastAccessFunding`

**Attributes:**

None

**Child Elements:**

None
4.371 subscription

The subscription element is a required child of the recurringRequest element and the parent of several child element that define information about the recurring transaction stream to be handled by the Recurring Engine.

Parent Elements:
recurringRequest

Attributes:
None

Child Elements (required):
planCode

Child Elements (optional):
numberOfPayments, startDate, amount, createDiscount, createAddOn

NOTE: If you include the numberOfPayments child element, the value submitted overrides the default value defined in the Plan.

Example: subscription Structure

```xml
<subscription>
  <planCode>Plan Reference Code</planCode>
  <numberOfPayments>1 to 99</numberOfPayments>
  <startDate>Start Date of recurring Cycle</startDate>
  <amount>Amount of Recurring Payment</amount>
  <createDiscount>
    <discountCode>Discount Reference Code</discountCode>
    <name>Name of Discount</name>
    <amount>Amount of Discount</amount>
    <startDate>Start Date of Discount</startDate>
    <endDate>End Date of Discount</endDate>
  </createDiscount>
  <createAddOn>
    <name>Name of Add On</name>
    <amount>Amount of Add On</amount>
    <startDate>Start Date of Add On Charge</startDate>
</subscription>
```
<endDate>End Date of Add On Charge</endDate>
</createAddOn>
</subscription>
4.372 subscriptionId

The `subscriptionId` element is a required child of the `recurringResponse` element and defines the assigned identifier for the sequence of recurring billing transactions. You also use this element in the `updateSubscription` and `cancelSubscription` transactions to identify the subscription for changes/cancellation.

**Type** = Long; **minLength** = N/A; **maxLength** = 19

**Parent Elements:**
- recurringResponse, littleInternalRecurringRequest, cancelSubscription, updateSubscription, updateSubscriptionResponse, cancelSubscriptionResponse

**Attributes:**
None

**Child Elements:**
None
4.373 surchargeAmount

The `surchargeAmount` element defines the amount of the surcharge applied to the transaction by the merchant. Supply the value in cents without a decimal point. For example, a value of 400 signifies $4.00.

**Type** = Integer; **totalDigits** = 12

**NOTE:** Use of the `surchargeAmount` element applies to Visa or MasterCard credit card payments only. Also, you are required to notify the card networks and us of your intent to applying surcharges at least 30 days prior to implementing the surcharges. Please consult your Relationship Manager if you have additional questions.

**Parent Elements:**
authorization, authReversal, capture, credit, captureGivenAuth, forceCapture, sale

**Attributes:**
None

**Child Elements:**
None
4.374 systemTraceId

The `systemTraceId` element is an optional child of the `giftCardResponse` element, which specifies a Worldpay generated identifier associated with the transaction in our systems. You should retain this value for possible future use in gift card reversal transactions.

**Type** = Integer; **totalDigits** = 6

**Parent Elements:**
giftCardResponse

**Attributes:**
None

**Child Elements:**
None
4.375 taxAmount

The taxAmount element is a required child of the detailTax element and an optional child of the lineItemData element and defines the detail tax amount on the purchased good or service. The decimal is implied. Example: 500 = $5.00.

Type = Integer; totalDigits = 8

Parent Elements:
Required: detailTax
Optional: lineItemData

NOTE: If you include taxAmount as a child of lineItemData along with detailTax, the lineItemData taxAmount should be the sum of the taxAmount children from detailTax children.

Attributes:
None

Child Elements:
None
### 4.376 taxExempt

The `taxExempt` element is an optional child of the `enhancedData` element and specifies whether or not the transaction is exempt from sales tax. If you do not include this element, the value defaults to `false`.

| **NOTE:** You must set this element to true, if you set the `salesTax` element to 0. |

**Type** = Boolean; **Valid values** = true or false

**Parent Elements:**

`enhancedData`

**Attributes:**

None

**Child Elements:**

None
The `taxIncludedInTotal` element is an optional child of the `detailTax` element and defines whether or not the tax is included in the total purchase amount.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
- `detailTax`

**Attributes:**
- None

**Child Elements:**
- None
4.378 taxRate

The `taxRate` element is an optional child of the `detailTax` element and defines the tax rate applied to this specific taxable amount.

**Type** = Decimal; **totalDigits** = 5

**Parent Elements:**

- `detailTax`

**Attributes:**

None

**Child Elements:**

None
4.379 taxType

The `taxType` element is an optional child of several transaction types that designates the transaction as either a convenience fee or tax payment for merchants using the Visa Tax Payment Program or the MasterCard Convenience Fee Program.

**Type** = String (enum); **minLength** = N/A; **maxLength** = 1; **Valid Values** = payment or fee

**Parent Elements:**
authorization, captureGivenAuth, credit, forceCapture, sale

**Attributes:**
None

**Child Elements:**
None
4.380 taxTypeIdentifier

The `taxTypeIdentifier` element is an optional child of the `detailTax` element and defines the type of tax collected on this specific tax amount. If the tax type identifier is unknown, do not include this element.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = 2

**Parent Elements:**
detailTax

**Attributes:**
None

**Child Elements:**
None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Unknown</td>
</tr>
<tr>
<td>01</td>
<td>Federal/National Sales Tax</td>
</tr>
<tr>
<td>02</td>
<td>State Sales Tax</td>
</tr>
<tr>
<td>03</td>
<td>City Sales Tax</td>
</tr>
<tr>
<td>04</td>
<td>Local Sales Tax</td>
</tr>
<tr>
<td>05</td>
<td>Municipal Sales Tax</td>
</tr>
<tr>
<td>06</td>
<td>Other Tax</td>
</tr>
<tr>
<td>10</td>
<td>Value Added Tax (VAT)</td>
</tr>
<tr>
<td>11</td>
<td>Goods and Services Tax (GST)</td>
</tr>
<tr>
<td>12</td>
<td>Provincial Sales Tax (PST)</td>
</tr>
<tr>
<td>13</td>
<td>Harmonized Sales Tax (HST)</td>
</tr>
<tr>
<td>14</td>
<td>Quebec Sales Tax (QST)</td>
</tr>
<tr>
<td>20</td>
<td>Room Tax</td>
</tr>
<tr>
<td>21</td>
<td>Occupancy Tax</td>
</tr>
<tr>
<td>22</td>
<td>Energy Tax</td>
</tr>
</tbody>
</table>
4.381 terminalId

The `terminalId` element is an optional child of the `pos` element and defines the identifier of the terminal used at the point of sale.

**NOTE:** For MasterCard POS transactions, you must include the `terminalId` element.

*Type* = String; *minLength* = N/A; *maxLength* = 8 (No special characters allowed.)

**Parent Elements:**

- `pos`

**Attributes:**

- None

**Child Elements:**

- None
4.382 termsAndConditions

The termsAndConditions element is an optional child of the billMeLaterRequest element and defines the specific lending terms of the PayPal Credit account.

NOTE: As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = Integer; totalDigits = 5

Parent Elements:

billMeLaterRequest

Attributes:

None

Child Elements:

None
4.383 threatMetrixSessionId

The threatMetrixSessionId element is an optional child of the advancedFraudChecks element.

Type = String;  
Allowed Characters = a-z, A-Z, 0-9, -, _;  
minLength = 1;  
maxLength = 128

NOTE: While generated by you at the time the consumer accesses your page, each threatMetrixSessionId must include a 5-character prefix, supplied by your Implementation Consultant, followed by a dash ("-"). The remainder of the Id must be unique for each instance of the customer accessing your page.

Parent Elements:
advancedFraudChecks

Attributes:
None

Child Elements:
None
4.384 token

The token element has two uses depending upon whether the element concerns a Worldpay generated token (for tokenized merchants) or a PayPal generated token.

4.384.1 token (Worldpay generated card number replacement)

In this case, the token element replaces the card element in tokenized card transactions or the echeck element in eCheck transactions, and defines the tokenized payment card/account information.

Parent Elements:
authorization, captureGivenAuth, credit, forceCapture, sale, accountUpdate, updateSubscription, fastAccessFunding

Attributes:
None

Child Elements:
Required: litleToken
Optional: expDate, cardValidationNum, type, checkoutId

Example: token Structure with CVV

```xml
<token>
  <litleToken>Token</litleToken>
  <expDate>Card Expiration Date</expDate>
  <cardValidationNum>Card Validation Number</cardValidationNum>
  <type>Method of Payment</type>
</token>
```

Example: token Structure with checkoutId instead of CVV

```xml
<token>
  <litleToken>Token</litleToken>
  <expDate>Card Expiration Date</expDate>
  <type>Method of Payment</type>
</token>
```

NOTE: Although not a required element, Worldpay recommends you include the expDate element. If you converted PAN information to tokens using the registerTokenRequest transaction, we do not have the expDate value stored, so cannot add it to the transaction. Transactions without expDate have a high likelihood of decline.
<checkoutId>Low Value Token for CVV</checkoutId>
</token>

4.384.2 token (PayPal generated)

In this case, the token element is the token generated by PayPal.

Type = String; minLength = N/A; maxLength = N/A

Parent Elements:
paypal

Attributes:
None

Child Elements:
None
4.385 tokenMessage

The tokenMessage element provides a short, human-readable explanation of the tokenResponseCode (see Table 4-3).

Type = String; minLength = N/A; maxLength = N/A

Parent Elements:
tokenResponse

Attributes:
None

Child Elements:
None
4.386 tokenResponse

The `tokenResponse` element is the parent element for several children defining the registered token, as well as the either card type and BIN, or last three characters of the account number in the case of eChecks. This element appears in the response only if a tokenized merchant submits card or eCheck account information in the transaction request.

**Parent Elements:**

- `authorizationResponse`
- `captureGivenAuthResponse`
- `creditResponse`
- `echeckCreditResponse`
- `echeckRedepositResponse`
- `echeckSalesResponse`
- `echeckVerificationResponse`
- `forceCaptureResponse`
- `saleResponse`
- `updateSubscriptionResponse`

**Attributes:**

None

**Child Elements:**

**Required:** `tokenResponseCode`, `tokenMessage`

**Optional:** `litleToken`, `type`, `bin`, `eCheckAccountSuffix`

**Example:** tokenResponse Structure

```
<tokenResponse>
  <litleToken>Token</litleToken>
  <tokenResponseCode>Response Code</tokenResponseCode>
  <tokenMessage>Response Message</tokenMessage>
  <type>Method of Payment</type>
  <bin>BIN</bin>
  <eCheckAccountSuffix>Last 3 of Account Number</eCheckAccountSuffix> (returned for eCheck account tokens)
</tokenResponse>
```
4.387 tokenResponseCode

The `tokenResponseCode` element provides a 3-digit code (see Table 4-3) indicating the results of a transaction involving the conversion or attempted conversion of an account number to a token. The `tokenMessage` element contains a short, human-readable explanation of the `tokenResponseCode`.

**Type** = String; **minLength** = N/A; **maxLength** = 3

**Parent Elements:**
tokenResponse

**Attributes:**
None

**Child Elements:**
None

**TABLE 4-3**  tokenResponseCode and tokenMessage Values

<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>801</td>
<td>Account number was successfully registered</td>
</tr>
<tr>
<td>802</td>
<td>Account number was previously registered</td>
</tr>
<tr>
<td>820</td>
<td>Credit card number was invalid</td>
</tr>
<tr>
<td>821</td>
<td>Merchant is not authorized for tokens</td>
</tr>
<tr>
<td>822</td>
<td>Token was not found</td>
</tr>
<tr>
<td>823</td>
<td>Token was Invalid</td>
</tr>
<tr>
<td>898</td>
<td>Generic token registration error</td>
</tr>
<tr>
<td>899</td>
<td>Generic token use error</td>
</tr>
</tbody>
</table>
4.388 totalHealthcareAmount

The `totalHealthcareAmount` element is a required child of the `healthcareAmounts` element and defines the total amount of healthcare related purchases. This value must be the greater than or equal to the sum of the values applied to the following elements: `RxAmount`, `visionAmount`, `clinicOtherAmount`, and `dentalAmount`. For a Visa transaction, do not include the `visionAmount` in the total. The decimal is implied. Example: 500 = $5.00.

**NOTE:** You must include a value greater than 0, except for a Visa transaction that only includes a vision component.

Type = Integer; totalDigits = 8

Parent Elements:

Optional: `healthcareAmounts`

Attributes:

None

Child Elements:

None
4.389 track

The track element is child of the card element, which is required for card-present transactions. The contents of the track element is the data read from the magnetic stripe.

Type = String; minLength = 1; maxLength = 256

Parent Elements:

* card

Attributes:

None

Child Elements:

None
4.390 track1Status

The `track1Status` element is a required child of the `mpos` element. This element indicates whether the device read track 1 from the magnetic stripe. A value of 0 indicates a successful read, while a value of 1 indicates a failure.

Type = Integer; minInclusive = 0; maxInclusive = 1028

Parent Elements:
- mpos

Attributes:
- None

Child Elements:
- None
4.391 track2Status

The `track2Status` element is a required child of the `mpos` element. This element indicates whether the device read track 2 from the magnetic stripe. A value of 0 indicates a successful read, while a value of 1 indicates a failure.

**Type** = Integer; **minInclusive** = 0; **maxInclusive** = 1028

**Parent Elements:**
- `mpos`

**Attributes:**
- None

**Child Elements:**
- None
4.392 transactionAmount

The transactionAmount element is an optional child of the applepayResponse element and specifies the amount of the transaction. The decimal is implied. Example: 500 = $5.00.

Type = Integer; totalDigits = 12

Parent Elements:
applepayResponse

Attributes:
None

Child Elements:
None
4.393 transactionId

The `transactionId` element is used in two locations: in PayPal transactions, as a child of the `paypal` element and in Apple Pay transactions as a child of the `header` element.

4.393.1 transactionId as a Child of the `paypal` element

The `transactionId` element is a required child of the `paypal` element, specifying the transaction Id returned from PayPal.

**NOTE:** The value of the `transactionId` element must match the TRANSACTIONID value returned by the DoExpressCheckoutPayment call operation to PayPal.

Type = String; **minLength** = N/A; **maxLength** = N/A

Parent Elements:

`paypal`

Attributes:
None

Child Elements:
None

4.393.2 transactionId as a Child of the `header` element

The `transactionId` element is a required child of the `header` element and provides the hexadecimal transaction identifier generated on the device for an Apple Pay transaction.

Type = Hex Encoded String; **minLength** = N/A; **maxLength** = 250

Parent Elements:

`header`

Attributes:
None

Child Elements:
None
### 4.394 trialIntervalType

The `trialIntervalType` element is an optional child of the `createPlan` element and defines the interval period of a trial associated with the Plan. The overall length of a trial period is defined by the `trialIntervalType` combined with the `trialNumberOfIntervals` element.

**Type** = String (enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**
- `createPlan`

**Attributes:**
- None

**Child Elements:**
- None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MONTH</td>
<td>The trial interval one month.</td>
</tr>
<tr>
<td>DAY</td>
<td>The trial interval one day.</td>
</tr>
</tbody>
</table>
4.395 trialNumberOfIntervals

The trialNumberOfIntervals element is an optional child of the createPlan element and defines the number of trial intervals (trialIntervalType) associated with the Plan. The overall length of a trial period is defined by the trialIntervalType combined with the trialNumberOfIntervals element.

Type = Integer; minLength = 1; maxLength = 99

Parent Elements:
createPlan

Attributes:
None

Child Elements:
None
The `triggeredRule` element is an optional child of the `advancedFraudResult` element. It can appear multiple times in the response, once for each triggered rule from the ThreatMetrix policy. A triggered rule is one where the threshold is exceeded.

**Type** = String; **minLength** = N/A; **maxLength** = 64

**Parent Elements:**
advancedFraudResults

**Attributes:**
None

**Child Elements:**
None
4.397 txnTime

The `txnTime` element is an optional child of the `giftCardResponse` element, which specifies the date and time the transaction was processed by Worldpay. You should retain this value for possible future use in other gift card transactions, such as `giftCardCapture` and most reversal transactions. The format of the element is `YYYY-MM-DDTHH:MM:SS`. For example, `2016-11-21T11:00:00`.

Type = dateTime; minLength = N/A; maxLength = 20

Parent Elements:
`giftCardResponse`

Attributes:
None

Child Elements:
None
4.398 type

The type element has two uses in the cnpAPI depending upon the parent. In one case it defines the type of account used in the transaction in terms of association, company, PayPal, or eCheck. When used as a child of the FundingSource element, it defines the card type in terms of prepaid, credit, debit, FSA, or unknown.

4.398.1 type Element as a Child of the parent elements listed below

This type element defines the type of account used in the transaction in terms of card association, card company, PayPal, or eCheck.

Type = String (Enum); minLength = N/A; maxLength = 2

Parent Elements:

accountInformation, newCardInfo, newCardTokenInfo, originalCard, originalCardInfo, originalCardTokenInfo, originalToken, updatedCard, updatedToken, registerTokenResponse, tokenResponse, card, paypage, token

Attributes:

None

Child Elements:

None

Enumerations:

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC</td>
<td>MasterCard</td>
</tr>
<tr>
<td>VI</td>
<td>Visa</td>
</tr>
<tr>
<td>AX</td>
<td>American Express</td>
</tr>
<tr>
<td>DC</td>
<td>Diner’s Club (see Table B-1)</td>
</tr>
<tr>
<td>DI</td>
<td>Discover</td>
</tr>
<tr>
<td>PP</td>
<td>PayPal</td>
</tr>
<tr>
<td>JC</td>
<td>JCB (Japanese Credit Bureau)</td>
</tr>
<tr>
<td>BL</td>
<td>PayPal Credit</td>
</tr>
<tr>
<td>EC</td>
<td>eCheck</td>
</tr>
<tr>
<td>GC</td>
<td>Gift Card</td>
</tr>
<tr>
<td>&quot;&quot; (empty)</td>
<td>Card type unknown or undefined</td>
</tr>
</tbody>
</table>
4.398.2 type Element as a Child of fundingSource

This type element defines the card type in terms of prepaid, credit, debit, FSA, or unknown.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**

fundingSource

**Attributes:**

None

**Child Elements:**

None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNKNOWN</td>
<td>The card type can not be determined.</td>
</tr>
<tr>
<td>PREPAID</td>
<td>This is a prepaid card.</td>
</tr>
<tr>
<td>CREDIT</td>
<td>This is a credit card.</td>
</tr>
<tr>
<td>DEBIT</td>
<td>This is a debit card.</td>
</tr>
<tr>
<td>FSA</td>
<td>This is a Flexible Spending Account card. Cards of this type can be used only for IRS-approved healthcare items.</td>
</tr>
</tbody>
</table>

**NOTE:** The fundingSource element and its child elements, type and availableBalance are associated with the Insights features (see Issuer Insights on page 24.)

Please consult your Relationship Manager for additional information.
4.399 unitCost

The unitCost element is an optional child of the lineItemData element, which specifies the price of one unit of the item purchased. Although the schema defines it as an optional child of the enhancedData element, it is required by Visa for Level III interchange rates. The value must be greater than or equal to 0.

Type = Decimal; minInclusive value = 0, totalDigits = 12

Parent Elements:
lineItemData

Attributes:
None

Child Elements:
None
4.400 unitOfMeasure

The `unitOfMeasure` element is an optional child of the `lineItemData` element, which specifies the unit of measure of the purchased item. For example, each, kit, pair, gallon, and month would all be valid values. Although an optional element, it is required by Visa and MasterCard when specifying line item data.

**Type** = String; **minLength** = 1; **maxLength** = 12

**Parent Elements:**
- `lineItemData`

**Attributes:**
- None

**Child Elements:**
- None
4.401 unload

The unload element is the parent element for the transaction type that removes funds from a Gift Card.

Parent Elements:
littleOnlineRequest, batchRequest

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (all Required)
amount, orderSource, card
4.402 unloadResponse

The `unloadResponse` element is the parent element for information returned to you in response to an unload transaction. It can be a child of either a `litleOnlineResponse` element or a `batchResponse` element.

Parent Elements:
- `litleOnlineResponse`
- `batchResponse`

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Unload transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Unload transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Unload transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:

Required: `litleTxnId`, `response`, `responseTime`, `message`

Optional: `postDate`, `fraudResult`, `giftCardResponse`
4.403 unloadReversal

The `unloadReversal` element is the parent element for the transaction type that reverses the unloading of a Gift Card.

**Parent Elements:**
- litleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements: (Required)**
- card, originalRefCode, originalAmount, originalTxnTime, originalSystemTraceId, originalSequenceNumber

**Child Elements: (Optional)**
- litleTxnId
4.404 unloadReversalResponse

The `unloadReversalResponse` element is the parent element for information returned to you in response to an `unloadReversal` transaction.

**Parent Elements:**
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Unload Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> \ <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the Unload Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> \ <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the Unload Reversal transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code> \ <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`, `response`, `responseTime`, `message`

Optional: `postDate`, `giftCardResponse`
4.405 updateAddOn

The `updateAddOn` element is the parent of several child elements used to modify an additional charge added to an existing subscription.

**Parent Elements:**

`updateSubscription`

**Attributes:**

None

**Child Elements (all Required):**

`addOnCode`, `name`, `amount`, `startDate`, `endDate`

**Example: updateAddOn Structure**

```xml
<updateAddOn>
  <name>Name of Add On</name>
  <amount>Amount of Add On</amount>
  <startDate>Start Date of Add On Charge</startDate>
  <endDate>End Date of Add On Charge</endDate>
</updateAddOn>
```
4.406 updatedCard

The updatedCard element is an optional child of the accountUpdateResponse element, which contains child elements providing the updated information for the submitted card.

Parent Elements:
accountUpdateResponse

Attributes:
None

Child Elements:
type, number, expDate

Example: updatedCard Structure

```xml
<updatedCard>
  <type>Card Type</type>
  <number>New Account Number</number>
  <expDate>New Expiration Date</expDate>
</updatedCard>
```
4.407 updateCardValidationNumOnToken

The `updateCardValidationNumOnToken` element is the parent element for the transaction type used to update a CVV2/CVC2/CID code stored temporarily on the platform. You should only use this transaction type if you had previously submitted the account number and security code in a `registerTokenRequest` transaction and now need to change the CVV2/CVC2/CID value.

**Parent Elements:**
- `litleOnlineRequest`
- `batchRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleToken`, `cardValidationNum`

Optional: `orderId`
4.408 updateTimeCardValidationOnTokenResponse

The updateTimeCardValidationOnTokenResponse element is the parent element for the response to updateTimeCardValidationOnToken transactions.

Parent Elements:
littleOnlineResponse, batchResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the updateTimeCardValidationOnToken transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the updateTimeCardValidationOnToken transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the updateTimeCardValidationOnToken transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements:
Required: littleTxnId, response, message, responseTime
4.409 updateDiscount

The `updateDiscount` element is the parent of several child elements used to define updates to a discount applied to an existing subscription.

**Parent Elements:**

`updateSubscription`

**Attributes:**

None

**Child Elements (all Required):**

`discountCode`, `name`, `amount`, `startDate`, `endDate`

**Example:** customerInfo Structure

```xml
<updateDiscount>
  <discountCode>Discount Reference Code</discountCode>
  <name>Name of Discount</name>
  <amount>Amount of Discount</amount>
  <startDate>Start Date of Discount</startDate>
  <endDate>End Date of Discount</endDate>
</updateDiscount>
```
4.410 updatePlan

The *updatePlan* element is the parent of the transaction used to activate/deactivate Plans associated with recurring payments. When you deactivate a Plan, you can no longer reference that Plan for use with subscriptions. Existing subscriptions making use of the deactivated Plan will continue to use the Plan until either modified or completed. You can also reactivate a deactivated Plan by updating the Plan and setting the *active* flag to true.

**Parent Elements:**
littleOnlineRequest, littleRequest

**Attributes:**
None

**Child Elements:**
planCode, active

**Example:** createPlan Structure

```
<updatePlan>
  <planCode>Plan Reference Code</planCode>
  <active>true or false</active>
</updatePlan>
```
4.411 updatePlanResponse

The updatePlanResponse element is the parent of the response message to the updatePlan transaction used to deactivate Plans associated with recurring payments.

Parent Elements:
litleOnlineResponse, batchResponse

Attributes:
None

Child Elements:
litleTxnId, response, message, responseTime, planCode

Example: updatePlan Structure

```xml
<updatePlan>
  <litleTxnId>Transaction ID</litleTxnId>
  <response>Response Reason Code</response>
  <message>Response Message</message>
  <responseTime>Date and Time in GMT</responseTime>
  <planCode>Plan Reference Code</planCode>
</updatePlan>
```
4.412 updateSubscription

The `updateSubscription` element is the parent element for the transaction that updates the subscription information associated with a recurring payment. Using this transaction type you can change the plan, card, billing information, and/or billing date. You can also create, update, or delete a Discount and/or an Add On.

**Parent Elements:**
- litleOnlineRequest, batchRequest

**Attributes:**
- None

**Child Elements:**
- Required: `subscriptionId`
- Optional: `planCode`, `billToAddress`, (choice of) `card`, `paypage`, or `token`, `billingDate`, `createDiscount`, `deleteDiscount`, `updateDiscount`, `createAddOn`, `updateAddOn`, `deleteAddOn`

**Example: updateSubscription - Change Plan**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <planCode>New Plan Code</planCode>
</updateSubscription>
```

**Example: updateSubscription - Change Card**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <card>
    <type>Card Type Abbreviation</type>
    <number>Account Number</number>
    <expDate>Expiration Date</expDate>
  </card>
</updateSubscription>
```

**Example: updateSubscription - Change Billing Date**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <billingDate>New Billing Date</billingDate>
</updateSubscription>
```
**Example: updateSubscription - Change Billing Info**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <billToAddress>
    <name>Customer’s Full Name</name>
    <companyName>Company’s Name</companyName>
    <addressLine1>Address Line 1</addressLine1>
    <addressLine2>Address Line 2</addressLine2>
    <addressLine3>Address Line 3</addressLine3>
    <city>City</city>
    <state>State Abbreviation</state>
    <zip>Postal Code</zip>
    <country>Country Code</country>
    <email>Email Address</email>
    <phone>Telephone Number</phone>
  </billToAddress>
</updateSubscription>
```

**Example: updateSubscription - Create Discount**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <createDiscount>
    <discountCode>Discount Reference Code</discountCode>
    <name>Name of Discount</name>
    <amount>Amount of Discount</amount>
    <startDate>Start Date of Discount</startDate>
    <endDate>End Date of Discount</endDate>
  </createDiscount>
</updateSubscription>
```

**Example: updateSubscription - Create Add On**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <createAddOn>
    <name>Name of Add On</name>
    <amount>Amount of Add On</amount>
  </createAddOn>
</updateSubscription>
```
<startDate>Start Date of Add On Charge</startDate>
<endDate>End Date of Add On Charge</endDate>
</createAddOn>
</updateSubscription>

**Example: updateSubscription - Update Discount**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <updateDiscount>
    <discountCode>Discount Reference Code</discountCode>
    <name>Name of Discount</name>
    <amount>Amount of Discount</amount>
    <startDate>Start Date of Discount</startDate>
    <endDate>End Date of Discount</endDate>
  </updateDiscount>
</updateSubscription>
```

**Example: updateSubscription - Update Add On**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <updateAddOn>
    <name>Name of Add On</name>
    <amount>Amount of Add On</amount>
    <startDate>Start Date of Add On Charge</startDate>
    <endDate>End Date of Add On Charge</endDate>
  </updateAddOn>
</updateSubscription>
```

**Example: updateSubscription - Delete Discount**

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <deleteDiscount>
    <discountCode>Discount Reference Code</discountCode>
  </deleteDiscount>
</updateSubscription>
```
Example: updateSubscription - Delete Add On

```xml
<updateSubscription>
  <subscriptionId>Subscription Id</subscriptionId>
  <deleteAddOn>
  </deleteAddOn>
</updateSubscription>
```
4.413 updateSubscriptionResponse

The updateSubscriptionResponse element is the parent element for the response to an updateSubscription transaction.

**Parent Elements:**
litleOnlineResponse, batchResponse

**Attributes:**
None

**Child Elements:**
subscriptionId, litleTxnId, response, message, responseTime, tokenResponse
4.414 updatedToken

The updatedToken element is an optional child of the accountUpdateResponse element, which contains child elements providing the updated information for the submitted token.

**Parent Elements:**
accountUpdateResponse

**Attributes:**
None

**Child Elements:**
type, number, expDate, bin

**Example: originalCard Structure**

```xml
<updatedToken>
  <litleToken>New Token Number</litleToken>
  <expDate>New Expiration Date</expDate>
  <type>Card Type</type>
  <bin>Card BIN</bin>
</updatedToken>
```
4.415 url

The `url` element is an optional child of the `customBilling` element. You use it to designate your customer service web site instead of providing a customer service phone number. This element may include any of the following characters: A-Z, a-z, 0-9, /, \, -, ., or _.

**Type** = String; **minLength** = N/A; **maxLength** = 13

**NOTE:** Please consult your Relationship Manager prior to attempting to use the `<url>` element. This contents of this element are discarded unless you are specifically enabled to use in your cnpAPI submissions.

**Parent Elements:**
- `customBilling`

**Attributes:**
- None

**Child Elements:**
- None
4.416 user

The user element is a required child of the authentication element. It is a unique identifier of the user/merchant used to authenticate that the message is from a valid source.

**Type** = String; **minLength** = N/A; **maxLength** = 20

**Parent Elements:**
- authentication

**Attributes:**
None

**Child Elements:**
None
### 4.417 vendorCredit

The `vendorCredit` element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the PayFac Settlement Account the Vendor Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**

- `batchRequest`, `littleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code>  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

- Required: `accountInfo`, `amount`, `fundingSubmerchantId`, `fundsTransferId`, `vendorName`
4.418 vendorCreditResponse

The `vendorCreditResponse` element is the parent element for information returned to you in response to a `vendorCredit` transaction.

**Parent Elements:**
- `batchResponse`
- `litleOnlineResponse`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String    | Yes       | The response returns the same value submitted in the `payFacCredit` transaction.  
|                |           |           | `minLength = 1`  
|                |           |           | `maxLength = 36` |
| customerId     | String    | No        | The response returns the same value submitted in the `payFacCredit` transaction.  
|                |           |           | `minLength = N/A`  
|                |           |           | `maxLength = 50` |
| reportGroup    | String    | Yes       | The response returns the same value submitted in the `submerchantCredit` transaction.  
|                |           |           | `minLength = 1`  
|                |           |           | `maxLength = 25` |
| duplicate      | boolean   | No        | If the request is a duplicate, the response includes the duplicate flag set to `true` and the entire original response.  
|                |           |           | **Note:** This attribute applies only to Online Dynamic Payout Funding Instruction responses. |

**Child Elements:**

Required: `litleTxnId`, `fundsTransferId`, `response`, `responseTime`, `message`

Required (Online): `postDate`
### 4.419 vendorDebit

The `vendorDebit` element is the parent element for the transaction type that a Payment Facilitator uses to move funds from the Vendor Account to the PayFac Settlement Account. If you use V11.3 or above, you can submit this transaction type either in a Batch or Online.

**Parent Elements:**
- batchRequest
- littleOnlineRequest

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>   <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = N/A</code> <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>For Payment Facilitators, this attribute does not segregate transactions in iQ. This field does appear in various SSR reports.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><code>minLength = 1</code>   <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: accountInfo, amount, fundingSubmerchantId, fundsTransferId, vendorName
4.420 vendorDebitResponse

The vendorDebitResponse element is the parent element for information returned to you in response to a vendorDebit transaction.

Parent Elements:
batchResponse, litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
</table>
| id             | String| Yes       | The response returns the same value submitted in the payFacCredit transaction.  
|                |       |           | minLength = 1  maxLength = 36 |
| customerId     | String| No        | The response returns the same value submitted in the payFacCredit transaction.  
|                |       |           | minLength = N/A  maxLength = 50 |
| reportGroup    | String| Yes       | The response returns the same value submitted in the submerchantCredit transaction.  
|                |       |           | minLength = 1  maxLength = 25 |
| duplicate      | boolean| No       | If the request is a duplicate, the response includes the duplicate flag set to true and the entire original response.  
|                |       |           | Note: This attribute applies only to Online Dynamic Payout Funding Instruction responses. |

Child Elements:

Required: litleTxnId, fundsTransferId, response, responseTime, message

Required (Online): postDate
4.421 vendorName

The `vendorName` element is a required child of both the `vendorCredit` and `vendorDebit` elements and specifies the name of the vendor involved in the funding instructions.

Type = String; minLength = 1; maxLength = 256

Parent Elements:
vendorCredit, vendorDebit

Attributes:
None

Child Elements:
None
### 4.422 verificationCode

**NOTE:** This element is not used at this time.

The verificationCode element is an optional child of the echeckSaleResponse element. It specifies the verification code from the associated eCheck Sale transaction.

**Type** = String; **minLength** = N/A; **maxLength** = 6

**Parent Elements:**
- echeckSalesResponse

**Attributes:**
- None

**Child Elements:**
- None
4.423 verify

The `verify` element is an optional child of the `echeckSale` element, which allows you to specify to perform an eCheck Verification prior to processing the sale. If the account fails the verification operation, the system does not process the sale.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**

- `echeckSale`

**Attributes:**

None

**Child Elements:**

None
4.424 version

The `version` element is a required child of the `applepay` element and provides version information about the payment token.

**Type** = String; **minLength** = 5; **maxLength** = 20

**Parent Elements:**
- `applepay`

**Attributes:**
- None

**Child Elements:**
- None
The `virtualAccountNumber` element is an optional child of the `enhancedAuthResponse` element and indicates if the card number used for the transaction corresponds to a virtual account number.

**Type** = Boolean; **Valid Values** = true or false

**Parent Elements:**
`enhancedAuthResponse`

**Attributes:**
None

**Child Elements:**
None
4.426 virtualAuthenticationKeyData

The `virtualAuthenticationKeyData` is an optional child of the `billMeLaterRequest` element.

**Type** = String; **minLength** = N/A; **maxLength** = 4

**Parent Elements:**
- `billMeLaterRequest`

**Attributes:**
- None

**Child Elements:**
- None
**4.427 virtualAuthenticationKeyPresenceIndicator**

The `virtualAuthenticationKeyPresenceIndicator` is an optional child of the `billMeLaterRequest` element.

**Type** = String; **minLength** = N/A; **maxLength** = 1

**Parent Elements:**
- `batchRequest`

**Attributes:**
None

**Child Elements:**
None
4.428 virtualGiftCard

The virtualGiftCard element is an optional child of the activate transaction. You include this element when you are requesting a Virtual Gift Card.

NOTE: In an early iteration of schema V8.22 issued in September of 2013 the accountNumberLength and giftCardBin child elements were defined as optional. If you coded to the earlier version, be aware that these elements are now required children of virtualGiftCard. If you do not include these elements, the transaction will fail XML validation.

Parent Elements:
activate

Attributes:
None

Child Elements (all required):
accountNumberLength, giftCardBin

Example: virtualGiftCard Structure

```xml
<virtualGiftCard>
  <accountNumberLength>Length of Virtual Card Number</accountNumberLength>
  <giftCardBin>Requested BIN of Virtual Gift Card</giftCardBin>
</virtualGiftCard>
```
4.429 virtualGiftCardBin

The `VirtualGiftCardBin` element is an optional child of the `activateReversal` element defining the BIN of the virtual Gift Card.

**Type** = String; **minLength** = N/A; **maxLength** = 10

**Parent Elements:**
- `activateReversal`

**Attributes:**
- None

**Child Elements:**
- None
4.430 virtualGiftCardResponse

The virtualGiftCardResponse element is an optional child of the activateResponse transaction. This element is returned when you request a Virtual Gift Card number via an activate transaction and through its children, defines the virtual gift Card number, as well as the Card Validation number.

Parent Elements:
activateResponse

Attributes:
None

Child Elements (all optional):
accountNumber, cardValidationNum, pin

Example: virtualGiftCardResponse Structure

```xml
<virtualGiftCardResponse>
    <accountNumber>Virtual Card Number</accountNumber>
    <cardValidationNum>Validation Number</cardValidationNum>
    <pin>Pin Number</pin>
</virtualGiftCardResponse>
```
**4.431 visionAmount**

The `visionAmount` element is an optional child of the `healthcareAmounts` element and defines the healthcare amount used for vision related purchases. The decimal is implied. Example: 500 = $5.00.

**Type** = Integer; **totalDigits** = 8

**Parent Elements:**
Optional: `healthcareAmounts`

**Attributes:**
None

**Child Elements:**
None
4.432 void

The `void` element is the parent element for all Void transactions. You can use this element only in Online transactions. If you use this Recycling Engine, you can use the `void` transaction to halt the recycling of a sale transaction.

**NOTE:** Before submitting a Void, please allow a minimum of 60 seconds to elapse after submitting the transaction you wish to void. This timing ensures our system fully records the first (to be voided) transaction in our database.

**Parent Elements:**
`litleOnlineRequest`

**Attributes:**

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>A unique identifier assigned by the presenter and mirrored back in the response. \n  <code>minLength = 1</code> \n  <code>maxLength = 36</code></td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>A value assigned by the merchant to identify the consumer. \n  <code>minLength = N/A</code> \n  <code>maxLength = 50</code></td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>Required attribute that defines the merchant sub-group in the user interface where this transaction will be displayed. Please refer to Coding for Report Groups on page 10 for additional information. \n  <code>minLength = 1</code> \n  <code>maxLength = 25</code></td>
</tr>
</tbody>
</table>

**Child Elements:**

Required: `litleTxnId`

Optional: `processingInstructions`
4.433 voidResponse

The voidResponse element is the parent element for information returned to you in response to a Void transaction.

Parent Elements:

litleOnlineResponse

Attributes:

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Type</th>
<th>Required?</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the void transaction. minLength = 1 maxLength = 36</td>
</tr>
<tr>
<td>customerId</td>
<td>String</td>
<td>No</td>
<td>The response returns the same value submitted in the void transaction. minLength = N/A maxLength = 50</td>
</tr>
<tr>
<td>reportGroup</td>
<td>String</td>
<td>Yes</td>
<td>The response returns the same value submitted in the void transaction. minLength = 1 maxLength = 25</td>
</tr>
</tbody>
</table>

Child Elements: (Required)
litleTxnId, response, responseTime, message, postDate

Child Elements: (Optional)

recycling
4.434 wallet

The wallet element is an optional child of the authorization and sale transactions. You must use this element along with its child elements, when the consumer uses MasterPass or Visa Checkout to make a purchase.

Parent Elements:
authorization, sale

Attributes:
None

Child Elements:
walletSourceType, walletSourceTypeId

Example: wallet Structure

```xml
<wallet>
  <walletSourceType>MasterPass or VisaCheckout</walletSourceType>
  <walletSourceTypeId>MasterPass ID or VCIND</walletSourceTypeId>
</wallet>
```
**4.435 walletSourceType**

The `walletSourceType` element is a required child of the `wallet` element, which defines the source of the transaction information. You must submitted this element with the transaction when the consumer uses MasterPass or Visa Checkout.

**Type** = String (Enum); **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**

- wallet

**Attributes:**

- None

**Child Elements:**

- None

**Enumerations:**

<table>
<thead>
<tr>
<th>Enumeration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterPass</td>
<td>The origin of the information used in the transaction is from MasterPass.</td>
</tr>
<tr>
<td>VisaCheckout</td>
<td>The origin of the information used in the transaction is from Visa Checkout.</td>
</tr>
</tbody>
</table>
4.436 walletSourceType

The `walletSourceType` element is a required child of the `wallet` element. For MasterPass transactions, the value of this element is returned from MasterPass. For Visa Checkout transactions, set this value to VCIND.

**Type** = String; **minLength** = N/A; **maxLength** = N/A

**Parent Elements:**

`wallet`

**Attributes:**

None

**Child Elements:**

None
4.437 yearsAtEmployer

The `yearsAtEmployer` element is an optional child of the `customerInfo` element and defines the number of years the customer has worked for their current employer. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = Integer; totalDigits = 2

Parent Elements:

- `customerInfo`

Attributes:

None

Child Elements:

None
4.438 yearsAtResidence

The `yearsAtResidence` element is an optional child of the `customerInfo` element and defines the number of years the customer has resided in their current domicile. It is used in combination with several other elements to provide required information for some PayPal Credit transactions.

**NOTE:** As of September, 2016, the Worldpay eCommerce platform no longer supports PayPal Credit for new merchants or existing merchants not already using this payment method. Existing merchants already using PayPal Credit should consult their Worldpay Relationship Manager.

Type = Integer; totalDigits = 2

**Parent Elements:**
- `customerInfo`

**Attributes:**
- None

**Child Elements:**
- None
**4.439 zip**

The `zip` element defines the customer’s postal code in both the `billToAddress` and `shipToAddress` elements.

**Type** = String; **minLength** = N/A; **maxLength** = 20

```
NOTE: Although the schema specifies the maxLength of the zip element as 20 characters, in practice you should never exceed 10 characters in your submissions.

If including the zip element for eCheck Verification, do not exceed 9 characters and do not use dashes.
```

**Parent Elements:**

`billToAddress`, `shipToAddress`

**Attributes:**

None

**Child Elements:**

None
Payment Transaction Response Codes

This appendix provides reference material regarding the codes that are returned in a cnpAPI response for a payment transaction. This appendix contains the following sections:

- Payment Transaction Response Codes
- 3DS Authentication Result Codes
- AVS Response Codes
- AAVS Response Codes
- Card Validation Response Codes
- Advanced Fraud Tools Triggered Rules
- XML Validation Error Messages
- Additional Response Header Error Messages
- ACH Return Reason Codes
- ACH NoC Change Codes
- Canadian eCheck Return Codes
A.1 Payment Transaction Response Codes

This section contains a list of codes and messages that the system can return in the response message for a payment transaction.

**NOTE:** For information concerning Chargeback Response Code, see the Worldpay eComm Chargeback API Reference Guide.

Table A-1 shows all possible values for the `<response>` and `<message>` elements. You should code appropriately to handle all codes applicable to the transactions you use.

- The Response Code value appears in the `<response>` element.
- The Response Message value appears in the `<message>` element.

### TABLE A-1 Valid Values for the Response and Message Elements

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>Transaction Received</td>
<td>Info</td>
<td>This is sent to acknowledge that the submitted transaction has been received.</td>
</tr>
<tr>
<td>000</td>
<td>Approved</td>
<td>Approved</td>
<td>No action required.</td>
</tr>
<tr>
<td>010</td>
<td>Partially Approved</td>
<td>Approved</td>
<td>The authorized amount is less than the requested amount.</td>
</tr>
<tr>
<td>011</td>
<td>Offline Approval</td>
<td>Approved</td>
<td>Offline approval issued while the terminal is unable to communicate with the issuer.</td>
</tr>
<tr>
<td>013</td>
<td>Offline Approval (unable to go online)</td>
<td>Approved</td>
<td>Offline approval issued while the terminal is unable to communicate with the issuer.</td>
</tr>
<tr>
<td>100</td>
<td>Processing Network Unavailable</td>
<td>Soft Decline</td>
<td>There is a problem with the card network. Contact the network for more information.</td>
</tr>
<tr>
<td>101</td>
<td>Issuer Unavailable</td>
<td>Soft Decline</td>
<td>There is a problem with the issuer network. Please contact the issuing bank.</td>
</tr>
<tr>
<td>102</td>
<td>Re-submit Transaction</td>
<td>Soft Decline</td>
<td>There is a temporary problem with your submission. Please re-submit the transaction.</td>
</tr>
<tr>
<td>108</td>
<td>Try again later</td>
<td>Soft</td>
<td>Returned if the eProtect token is not immediately available, when submitting an Auth or Sale transaction.</td>
</tr>
<tr>
<td>110</td>
<td>Insufficient Funds</td>
<td>Soft Decline</td>
<td>The card does not have enough funds to cover the transaction.</td>
</tr>
<tr>
<td>111</td>
<td>Authorization amount has already been depleted</td>
<td>Hard Decline</td>
<td>The total amount of the original Authorization has been used. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>120</td>
<td>Call Issuer</td>
<td>Referral or Soft</td>
<td>There is an unspecified problem, contact the issuing bank.</td>
</tr>
<tr>
<td>121</td>
<td>Call AMEX</td>
<td>Referral</td>
<td>There is an unspecified problem; contact AMEX.</td>
</tr>
<tr>
<td>122</td>
<td>Call Diners Club</td>
<td>Referral</td>
<td>There is an unspecified problem; contact Diners Club.</td>
</tr>
<tr>
<td>123</td>
<td>Call Discover</td>
<td>Referral</td>
<td>There is an unspecified problem; contact Discover.</td>
</tr>
<tr>
<td>124</td>
<td>Call JBS</td>
<td>Referral</td>
<td>There is an unspecified problem; contact JBS.</td>
</tr>
<tr>
<td>125</td>
<td>Call Visa/MasterCard</td>
<td>Referral</td>
<td>There is an unspecified problem; contact Visa or MasterCard.</td>
</tr>
<tr>
<td>126</td>
<td>Call Issuer - Update Cardholder Data</td>
<td>Referral</td>
<td>Some data is out of date; contact the issuer to update this information.</td>
</tr>
<tr>
<td>127</td>
<td>Exceeds Approval Amount Limit</td>
<td>Hard Decline</td>
<td>This transaction exceeds the daily approval limit for the card.</td>
</tr>
<tr>
<td>130</td>
<td>Call Indicated Number</td>
<td>Referral</td>
<td>There is an unspecified problem; contact the phone number provided.</td>
</tr>
<tr>
<td>140</td>
<td>Update Cardholder Data</td>
<td>Referral</td>
<td>Cardholder data is incorrect; contact the issuing bank.</td>
</tr>
<tr>
<td>150</td>
<td>Original transaction found.</td>
<td>Info</td>
<td>A Query transaction response indicating that the original transaction was found.</td>
</tr>
<tr>
<td>151</td>
<td>Original transaction not found.</td>
<td>Info</td>
<td>A Query transaction response indicating that the original transaction was not found.</td>
</tr>
<tr>
<td>152</td>
<td>Original transaction found, but response not yet available.</td>
<td>Info</td>
<td>A Query transaction response indicating that the original transaction was found, but the final response information is not yet available.</td>
</tr>
<tr>
<td>153</td>
<td>Query transaction not enabled.</td>
<td>Info</td>
<td>A Query transaction response indicating that you are not enabled for use of the Query transaction.</td>
</tr>
<tr>
<td>191</td>
<td>The merchant is not registered in the update program.</td>
<td>N/A</td>
<td>This is an Account Updater response indicating a set-up problem that must be resolved prior to submitting another request file. Escalate this to your Relationship Manager.</td>
</tr>
<tr>
<td>192</td>
<td>Merchant not certified/enabled for IIAS</td>
<td>Hard Decline</td>
<td>Your organization is not certified or enabled for IIAS/FSA transactions.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>206</td>
<td>Issuer Generated Error</td>
<td>Soft Decline</td>
<td>An unspecified error was returned by the issuer. Please retry the transaction and if the problem persist, contact the issuing bank.</td>
</tr>
<tr>
<td>207</td>
<td>Pickup card - Other than Lost/Stolen</td>
<td>Hard Decline</td>
<td>The issuer indicated that the gift card should be removed from use.</td>
</tr>
<tr>
<td>209</td>
<td>Invalid Amount</td>
<td>Hard Decline</td>
<td>The specified amount is invalid for this transaction.</td>
</tr>
<tr>
<td>211</td>
<td>Reversal Unsuccessful</td>
<td>Hard Decline</td>
<td>The reversal transaction was unsuccessful.</td>
</tr>
<tr>
<td>212</td>
<td>Missing Data</td>
<td>Hard Decline</td>
<td>Contact your Relationship Manager.</td>
</tr>
<tr>
<td>213</td>
<td>Pickup Card - Lost Card</td>
<td>Hard Decline</td>
<td>The submitted card was reported as lost and should be removed from use.</td>
</tr>
<tr>
<td>214</td>
<td>Pickup Card - Stolen Card</td>
<td>Hard Decline</td>
<td>The submitted card was reported as stolen and should be removed from use.</td>
</tr>
<tr>
<td>215</td>
<td>Restricted Card</td>
<td>Hard Decline</td>
<td>The specified Gift Card is not available for use.</td>
</tr>
<tr>
<td>216</td>
<td>Invalid Deactivate</td>
<td>Hard Decline</td>
<td>The Deactivate transaction is invalid for the specified card.</td>
</tr>
<tr>
<td>217</td>
<td>Card Already Active</td>
<td>Hard Decline</td>
<td>The submitted card is already active.</td>
</tr>
<tr>
<td>218</td>
<td>Card Not Active</td>
<td>Hard Decline</td>
<td>The submitted card has not been activated.</td>
</tr>
<tr>
<td>219</td>
<td>Card Already Deactivate</td>
<td>Hard Decline</td>
<td>The submitted card has already been deactivated.</td>
</tr>
<tr>
<td>221</td>
<td>Over Max Balance</td>
<td>Hard Decline</td>
<td>The activate or load amount exceeds the maximum allowed for the specified gift Card.</td>
</tr>
<tr>
<td>222</td>
<td>Invalid Activate</td>
<td>Hard Decline</td>
<td>The activate transaction is not valid or can no longer be reversed.</td>
</tr>
<tr>
<td>223</td>
<td>No transaction Found for Reversal</td>
<td>Hard Decline</td>
<td>The transaction referenced in the reversal transaction does not exist.</td>
</tr>
<tr>
<td>226</td>
<td>Incorrect CVV</td>
<td>Hard Decline</td>
<td>The transaction was declined because it was submitted with the incorrect security code.</td>
</tr>
<tr>
<td>229</td>
<td>Illegal Transaction</td>
<td>Hard Decline</td>
<td>The transaction would violate the law.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------</td>
<td>----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>251</td>
<td>Duplicate Transaction</td>
<td>Hard Decline</td>
<td>The transaction is a duplicate of a previously submitted transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>252</td>
<td>System Error</td>
<td>Hard Decline</td>
<td>Contact your Relationship Manager.</td>
</tr>
<tr>
<td>253</td>
<td>Deconverted BIN</td>
<td>Hard Decline</td>
<td>The BIN is no longer valid.</td>
</tr>
<tr>
<td>254</td>
<td>Merchant Depleted</td>
<td>Hard Decline</td>
<td>No balance remains on gift Card.</td>
</tr>
<tr>
<td>255</td>
<td>Gift Card Escheated</td>
<td>Hard Decline</td>
<td>The Gift Card has been seized by the government while resolving an estate.</td>
</tr>
<tr>
<td>256</td>
<td>Invalid Reversal Type for Credit Card Transaction</td>
<td>Hard Decline</td>
<td>You attempted to use a Closed Loop Gift Card reversal transaction to reverse</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a credit card transaction. For example, you cannot use a Deposit Reversal</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>transaction to reverse a Capture. To reverse a credit card Capture transaction, use a Credit transaction.</td>
</tr>
<tr>
<td>257</td>
<td>System Error (message format error)</td>
<td>Hard Decline</td>
<td>Issuer reported message format is incorrect. Contact your Relationship Manager.</td>
</tr>
<tr>
<td>258</td>
<td>System Error (cannot process)</td>
<td>Hard Decline</td>
<td>Issuer reported transaction could not be processed. Contact your Relationship Manager.</td>
</tr>
<tr>
<td>271</td>
<td>Refund rejected due to pending deposit status</td>
<td>Soft Decline</td>
<td>The refund is tied to a deposit that is still in pending state or the state is in doubt. You can retry the refund at a later time.</td>
</tr>
<tr>
<td>272</td>
<td>Refund rejected due to declined deposit status</td>
<td>Hard Decline</td>
<td>The refund is tied to a deposit that failed.</td>
</tr>
<tr>
<td>273</td>
<td>Refund rejected by the processing network</td>
<td>Soft Decline</td>
<td>The refund is tied to a deposit that succeeded, but was declined by PayPro.</td>
</tr>
<tr>
<td>284</td>
<td>Capture, Credit and AuthReversal tags cannot be used for Gift Card Transactions</td>
<td>Hard Decline</td>
<td>You must use the Gift Card version of these transactions for Gift Cards (i.e., giftCardCapture, giftCardCredit, and giftCardAuthReversal).</td>
</tr>
<tr>
<td>301</td>
<td>Invalid Account Number</td>
<td>Hard Decline</td>
<td>The account number is not valid; contact the cardholder to confirm information or inquire about another form of payment.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>302</td>
<td>Account Number Does Not Match Payment Type</td>
<td>Hard Decline</td>
<td>The payment type was selected as one card type (e.g. Visa), but the card number indicates a different card type (e.g. MasterCard).</td>
</tr>
<tr>
<td>303</td>
<td>Pick Up Card</td>
<td>Hard Decline</td>
<td>This is a card present response, but in a card not present environment. Do not process the transaction and contact the issuing bank.</td>
</tr>
<tr>
<td>304</td>
<td>Lost/Stolen Card</td>
<td>Hard Decline</td>
<td>The card has been designated as lost or stolen; contact the issuing bank.</td>
</tr>
<tr>
<td>305</td>
<td>Expired Card</td>
<td>Hard Decline</td>
<td>The card is expired.</td>
</tr>
<tr>
<td>306</td>
<td>Authorization has expired; no need to reverse</td>
<td>Hard Decline</td>
<td>The original Authorization is no longer valid, because it has expired. You can not perform an Authorization Reversal for an expired Authorization. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>307</td>
<td>Restricted Card</td>
<td>Hard Decline</td>
<td>The card has a restriction preventing approval for this transaction. Please contact the issuing bank for a specific reason. You may also receive this code if the transaction was declined due to Prior Fraud Advice Filtering and you are using a schema version V8.10 or older.</td>
</tr>
<tr>
<td>308</td>
<td>Restricted Card - Chargeback</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the Prior Chargeback Card Filtering Service or the card has a restriction preventing approval if there are any chargebacks against it.</td>
</tr>
<tr>
<td>309</td>
<td>Restricted Card - Prepaid Card Filtering Service</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the Prepaid Card Filtering service.</td>
</tr>
<tr>
<td>310</td>
<td>Invalid track data</td>
<td>Hard Decline</td>
<td>The track data is not valid.</td>
</tr>
<tr>
<td>311</td>
<td>Deposit is already referenced by a chargeback</td>
<td>Hard Decline</td>
<td>The deposit is already referenced by a chargeback; therefore, a refund cannot be processed against the original transaction. Appears in Declined Transaction report.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>312</td>
<td>Restricted Card - International Card Filtering Service</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the International Card Filtering Service.</td>
</tr>
<tr>
<td>313</td>
<td>International filtering for issuing card country &lt;country&gt;</td>
<td>Hard Decline</td>
<td>This is returned when the transaction involves a US based merchant processing Canadian transactions has a transaction that uses a US card.</td>
</tr>
<tr>
<td>315</td>
<td>Restricted Card - Auth Fraud Velocity Filtering Service</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the Auth Fraud Velocity Filtering Service.</td>
</tr>
<tr>
<td>316</td>
<td>Automatic Refund Already Issued</td>
<td>Hard Decline</td>
<td>This refund transaction is a duplicate for one already processed automatically by the Fraud Chargeback Prevention Service (FCPS). Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>318</td>
<td>Restricted Card - Auth Fraud Advice Filtering Service</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the Auth Fraud Advice Filtering Service.</td>
</tr>
<tr>
<td>319</td>
<td>Restricted Card - Fraud AVS Filtering Service</td>
<td>Hard Decline</td>
<td>This transaction is being declined due the operation of the Auth Fraud AVS Filtering Service.</td>
</tr>
<tr>
<td>320</td>
<td>Invalid Expiration Date</td>
<td>Hard Decline</td>
<td>The expiration date is invalid.</td>
</tr>
<tr>
<td>321</td>
<td>Invalid Merchant</td>
<td>Hard Decline</td>
<td>The card is not allowed to make purchases from this merchant (e.g. a Travel only card trying to purchase electronics).</td>
</tr>
<tr>
<td>322</td>
<td>Invalid Transaction</td>
<td>Hard Decline</td>
<td>The transaction is not permitted; contact the issuing bank.</td>
</tr>
</tbody>
</table>

**Note:** If you are enabled for Transaction Filtering, but have not upgraded to use schema version 8.3 or above, the system returns this code for transactions filtered by the Prepaid or International Card Filtering Service. If you are enabled for Velocity Fraud Filtering, but have not upgraded to V8.9, you will receive this code for filtered transactions. If you are enabled for AVS Fraud Filtering, but have not upgraded to V8.13, you will receive this code for filtered transactions.
### Payment Transaction Response Codes

**TABLE A-1** Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>323</td>
<td>No such issuer</td>
<td>Hard Decline</td>
<td>The card number references an issuer that does not exist. Do not process the transaction.</td>
</tr>
<tr>
<td>324</td>
<td>Invalid Pin</td>
<td>Hard Decline</td>
<td>The PIN provided is invalid. Appears in Declined Transaction report</td>
</tr>
<tr>
<td>325</td>
<td>Transaction not allowed at terminal</td>
<td>Hard Decline</td>
<td>The transaction is not permitted; contact the issuing bank.</td>
</tr>
<tr>
<td>326</td>
<td>Exceeds number of PIN entries</td>
<td>Hard Decline</td>
<td>(Referring to a debit card) The incorrect PIN has been entered excessively and the card is locked.</td>
</tr>
<tr>
<td>327</td>
<td>Cardholder transaction not permitted</td>
<td>Hard Decline</td>
<td>Merchant does not allow that card type or specific transaction.</td>
</tr>
<tr>
<td>328</td>
<td>Cardholder requested that recurring or installment payment be stopped</td>
<td>Hard Decline</td>
<td>Recurring/Installment Payments no longer accepted by the card issuing bank.</td>
</tr>
<tr>
<td>330</td>
<td>Invalid Payment Type</td>
<td>Hard Decline</td>
<td>This payment type is not accepted by the issuer.</td>
</tr>
<tr>
<td>331</td>
<td>Invalid POS Capability for Cardholder Authorized Terminal Transaction</td>
<td>Hard Decline</td>
<td>For a Cardholder Authorized Terminal Transaction the POS capability must be set to magstripe.</td>
</tr>
<tr>
<td>332</td>
<td>Invalid POS Cardholder ID for Cardholder Authorized Terminal Transaction</td>
<td>Hard Decline</td>
<td>For a Cardholder Authorized Terminal Transaction the POS Cardholder ID must be set to nopin.</td>
</tr>
<tr>
<td>335</td>
<td>This method of payment does not support authorization reversals</td>
<td>Hard Decline</td>
<td>You can not perform an Authorization Reversal transaction for this payment type.</td>
</tr>
<tr>
<td>336</td>
<td>Reversal amount does not match Authorization amount.</td>
<td>Hard Decline</td>
<td>For a merchant initiated reversal against an American Express authorization, the reversal amount must match the authorization amount exactly.</td>
</tr>
<tr>
<td>337</td>
<td>Transaction did not convert to Pinless</td>
<td>Soft Decline</td>
<td>Retry the transaction.</td>
</tr>
<tr>
<td>340</td>
<td>Invalid Amount</td>
<td>Hard Decline</td>
<td>The transaction amount is invalid (too high or too low). For example, less than 0 for an authorization, or less than .01 for other payment types.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>341</td>
<td>Invalid Healthcare Amounts</td>
<td>Hard Decline</td>
<td>The amount submitted with this FSA/Healthcare transaction is invalid. The FSA amount must be greater than 0, and cannot be greater than the transaction amount.</td>
</tr>
<tr>
<td>346</td>
<td>Invalid billing descriptor prefix</td>
<td>Hard Decline</td>
<td>The billing descriptor prefix submitted is not valid. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>347</td>
<td>Invalid billing descriptor</td>
<td>Hard Decline</td>
<td>The billing descriptor is not valid because you are not authorized to send transactions with custom billing fields. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>348</td>
<td>Invalid Report Group</td>
<td>Hard Decline</td>
<td>The Report Group specified in the transaction is invalid, because it is either not in the defined list of acceptable Report Groups or there is a mis-match between the Report Group and the defined Billing Descriptor.</td>
</tr>
<tr>
<td>349</td>
<td>Do Not Honor</td>
<td>Soft/Decline</td>
<td>The issuing bank has put a temporary hold on the card.</td>
</tr>
</tbody>
</table>
| 350           | Generic Decline                                       | Soft/Hard Decline | There is an unspecified problem; contact the issuing bank for more details.  
Note: This code can be a hard or soft decline, depending on the method of payment, and other variables. |
| 351           | Decline - Request Positive ID                         | Hard Decline  | Card Present transaction that requires a picture ID match.                                            |
| 352           | Decline CVV2/CID Fail                                 | Hard Decline  | The CVV2/CID is invalid.                                                                               |
| 354           | 3-D Secure transaction not supported by merchant      | Hard Decline  | You are not certified to submit 3-D Secure transactions.                                              |
| 356           | Invalid purchase level III, the transaction contained bad or missing data | Soft/Decline | Submitted Level III data is bad or missing.                                                            |
| 357           | Missing healthcareIIAS tag for an FSA transaction      | Hard Decline  | The FSA Transactions submitted does not contain the `<healthcareIIAS>` data element.                  |
| 358           | Restricted by Vantiv due to security code mismatch.   | Hard Decline  | The transaction was declined due to the security code (CV2, CID, etc) not matching.                   |
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>360</td>
<td>No transaction found with specified Transaction Id</td>
<td>Hard Decline</td>
<td>There were no transactions found with the specified Transaction Id. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>361</td>
<td>Authorization no longer available</td>
<td>Hard Decline</td>
<td>The authorization for this transaction is no longer available. Either the authorization has already been consumed by another capture, or the authorization has expired. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>362</td>
<td>Transaction Not Voided - Already Settled</td>
<td>Hard Decline</td>
<td>This transaction cannot be voided; it has already been delivered. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>363</td>
<td>Auto-void on refund</td>
<td>Hard Decline</td>
<td>This transaction (both capture and refund) has been voided. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>364</td>
<td>Invalid Account Number - original or NOC updated eCheck account required</td>
<td>Hard Decline</td>
<td>The submitted account number is invalid. Confirm the original account number or check NOC for new account number.</td>
</tr>
<tr>
<td>365</td>
<td>Total credit amount exceeds capture amount</td>
<td>Hard Decline</td>
<td>The amount of the credit is greater than the capture, or the amount of this credit plus other credits already referencing this capture are greater than the capture amount. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>366</td>
<td>Exceed the threshold for sending redeposits</td>
<td>Hard Decline</td>
<td>NACHA rules allow two redeposit attempts within 180 days of the settlement date of the initial deposit attempt. This threshold has been exceeded. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>367</td>
<td>Deposit has not been returned for insufficient/non-sufficient funds</td>
<td>Hard Decline</td>
<td>NACHA rules only allow redeposit attempts against deposits returned for Insufficient or Uncollected Funds. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>368</td>
<td>Invalid check number</td>
<td>Soft Decline</td>
<td>The check number is invalid.</td>
</tr>
<tr>
<td>369</td>
<td>Redeposit against invalid transaction type</td>
<td>Hard Decline</td>
<td>The redeposit attempted against an invalid transaction type. Appears in Declined Transaction report.</td>
</tr>
</tbody>
</table>
### Table A-1 Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>370</td>
<td>Internal System Error - Call Vantiv</td>
<td>Hard Decline</td>
<td>There is a problem with the system. Contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a>.</td>
</tr>
<tr>
<td>371</td>
<td>Original Transaction has been Processed - Future Redeposits Canceled</td>
<td>Hard Decline</td>
<td>Do not send additional redeposit transactions, since the original transaction was processed. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>372</td>
<td>Soft Decline - Auto Recycling In Progress</td>
<td>Soft Decline</td>
<td>The transaction was intercepted because it is being auto recycled by the Recycling Engine.</td>
</tr>
<tr>
<td>373</td>
<td>Hard Decline - Auto Recycling Complete</td>
<td>Hard Decline</td>
<td>The transaction was intercepted because auto recycling has completed with a final decline.</td>
</tr>
<tr>
<td>375</td>
<td>Merchant is not enabled for surcharging</td>
<td>Hard Decline</td>
<td>The submitted transaction contained a surcharge and the merchant is not enabled for surcharging.</td>
</tr>
<tr>
<td>376</td>
<td>This method of payment does not support surcharging</td>
<td>Hard Decline</td>
<td>The use of a surcharge is only allowed for Visa and MasterCard methods of payment.</td>
</tr>
<tr>
<td>377</td>
<td>Surcharge is not valid for debit or prepaid cards</td>
<td>Hard Decline</td>
<td>You cannot apply a surcharge to a transaction using a debit or prepaid card.</td>
</tr>
<tr>
<td>378</td>
<td>Surcharge cannot exceed 4% of the sale amount</td>
<td>Hard Decline</td>
<td>The surcharge in the submitted transaction exceeded 4% maximum allowed for a surcharge.</td>
</tr>
<tr>
<td>380</td>
<td>Secondary amount cannot exceed the sale amount</td>
<td>Hard Decline</td>
<td>The secondary amount exceeded the sale amount in the submitted transaction.</td>
</tr>
<tr>
<td>381</td>
<td>This method of payment does not support secondary amount</td>
<td>Hard Decline</td>
<td>The submitted method of payment does not allow the use of Convenience Fees.</td>
</tr>
<tr>
<td>382</td>
<td>Secondary amount cannot be less than zero</td>
<td>Hard Decline</td>
<td>The secondary amount must be a positive integer.</td>
</tr>
<tr>
<td>383</td>
<td>Partial transaction is not supported when including a secondary amount</td>
<td>Hard Decline</td>
<td>Transactions set to allow partial authorizations cannot include a secondary amount.</td>
</tr>
<tr>
<td>384</td>
<td>Secondary amount required on partial refund when used on deposit</td>
<td>Hard Decline</td>
<td>If the associated sale or capture transaction included a secondary amount, an associated partial refund must include a secondary amount.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>385</td>
<td>Secondary amount not allowed on refund if not included on deposit</td>
<td>Hard Decline</td>
<td>If the associated sale or capture transaction did not include a secondary amount, you cannot include a secondary amount on an associated refund.</td>
</tr>
<tr>
<td>401</td>
<td>Invalid E-mail</td>
<td>Hard Decline</td>
<td>The e-mail address provided is not valid. Verify that it was entered correctly.</td>
</tr>
<tr>
<td>469</td>
<td>Invalid Recurring Request - See Recurring Response for Details</td>
<td>Hard Decline</td>
<td>The Recurring Request was invalid, which invalidated the transaction. The Response Code and Message in the Recurring Response contains additional information.</td>
</tr>
<tr>
<td>470</td>
<td>Approved - Recurring Subscription Created</td>
<td>Approved</td>
<td>The recurring request was processed successfully.</td>
</tr>
<tr>
<td>471</td>
<td>Parent Transaction Declined - Recurring Subscription Not Created</td>
<td>Hard Decline</td>
<td>The original payment transaction was declined, so the recurring payments have not been scheduled. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>472</td>
<td>Invalid Plan Code</td>
<td>Hard Decline</td>
<td>The plan specified in the recurring request was invalid.</td>
</tr>
<tr>
<td>473</td>
<td>Scheduled Recurring Payment Processed</td>
<td>Approved</td>
<td>The scheduled recurring payment has been processed successfully.</td>
</tr>
<tr>
<td>475</td>
<td>Invalid Subscription Id</td>
<td>Hard Decline</td>
<td>The referenced subscription Id does not exist. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>477</td>
<td>Duplicate Add On Codes in Requests</td>
<td>Hard Decline</td>
<td>Multiple createAddOn requests submitted with the same Add On Code.</td>
</tr>
<tr>
<td>480</td>
<td>No Matching Discount Code for the Subscription</td>
<td>Hard Decline</td>
<td>The Discount Code supplied in the updateDiscount or deleteDiscount transaction does not exist. Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>481</td>
<td>Duplicate Discount Codes in Request</td>
<td>Hard Decline</td>
<td>Multiple createDiscount requests submitted with the same Discount Code.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>482</td>
<td>Invalid Start Date</td>
<td>Hard Decline</td>
<td>The supplied Start Date is invalid.</td>
</tr>
<tr>
<td>483</td>
<td>Merchant Not Registered for Recurring Engine</td>
<td>Hard Decline</td>
<td>You are not registered for the use of the Recurring Engine.</td>
</tr>
<tr>
<td>500</td>
<td>The account number was changed</td>
<td>Hard Decline</td>
<td>An Account Updater response indicating the Account Number changed from the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>original number.</td>
</tr>
<tr>
<td>501</td>
<td>The account was closed</td>
<td>Hard Decline</td>
<td>An Account Updater response indicating the account was closed. Contact the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>cardholder directly for updated information.</td>
</tr>
<tr>
<td>502</td>
<td>The expiration date was changed</td>
<td>N/A</td>
<td>An Account Updater response indicating the Expiration date for the card has</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>changed.</td>
</tr>
<tr>
<td>503</td>
<td>The issuing bank does not participate in the update</td>
<td>N/A</td>
<td>An Account Updater response indicating the issuing bank does not participate</td>
</tr>
<tr>
<td></td>
<td>program</td>
<td></td>
<td>in the update program</td>
</tr>
<tr>
<td>504</td>
<td>Contact the cardholder for updated information</td>
<td>N/A</td>
<td>An Account Updater response indicating you should contact the cardholder</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>directly for updated information.</td>
</tr>
<tr>
<td>505</td>
<td>No match found</td>
<td>N/A</td>
<td>An Account Updater response indicating no match was found in the updated</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>information.</td>
</tr>
<tr>
<td>506</td>
<td>No changes found</td>
<td>N/A</td>
<td>An Account Updater response indicating there have been no changes to the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>account information.</td>
</tr>
<tr>
<td>507</td>
<td>The cardholder has opted out of the update program</td>
<td>N/A</td>
<td>The cardholder requested that no updates be included for their account.</td>
</tr>
<tr>
<td>521</td>
<td>Soft Decline - Card reader decryption service is not</td>
<td>Soft Decline</td>
<td>The connection to the decryption service is currently unavailable. Please</td>
</tr>
<tr>
<td></td>
<td>available</td>
<td></td>
<td>retry the transaction and/or contact your Relationship Manager.</td>
</tr>
<tr>
<td>523</td>
<td>Soft Decline - Decryption failed</td>
<td>Soft Decline</td>
<td>Our attempt to decrypt the card information failed. Please retry the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>transaction.</td>
</tr>
<tr>
<td>524</td>
<td>Hard Decline - Input data is invalid.</td>
<td>Hard Decline</td>
<td>The submitted data is invalid.</td>
</tr>
<tr>
<td>530</td>
<td>Apple Pay Key Mismatch</td>
<td>Hard Decline</td>
<td>The submitted publicKeyHash element does not match any configured entries.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Contact your Implementation Consultant.</td>
</tr>
<tr>
<td>531</td>
<td>Apple Pay Decryption Failed</td>
<td>Hard Decline</td>
<td>Worldpay was unable to decrypt the submitted information.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>550</td>
<td>Advanced Fraud Filter Score Below Threshold</td>
<td>Hard Decline</td>
<td>The transaction was declined because the resulting Fraud Filter Score was below the acceptable threshold set in the merchant's policy.</td>
</tr>
<tr>
<td>560</td>
<td>System Error - Contact Worldpay representative</td>
<td>Soft Decline</td>
<td>There was an unspecified problem with the transaction. Please contact your Worldpay Relationship Manager.</td>
</tr>
<tr>
<td>561</td>
<td>Amazon Pay - Amazon Unavailable</td>
<td>Soft Decline</td>
<td>Amazon was unavailable. Please retry the transaction.</td>
</tr>
<tr>
<td>562</td>
<td>Amazon Pay - Amazon Declined</td>
<td>Hard Decline</td>
<td>Amazon declined the transaction.</td>
</tr>
<tr>
<td>563</td>
<td>Amazon Pay - Invalid Token</td>
<td>Hard Decline</td>
<td>The submitted Amazon token is invalid. Please correct the token value before resubmitting the transaction.</td>
</tr>
<tr>
<td>564</td>
<td>Merchant not enabled for Amazon Pay</td>
<td>Hard Decline</td>
<td>Your organization is not enabled for the use of Amazon Pay. Please contact your Relationship Manager.</td>
</tr>
<tr>
<td>601</td>
<td>Soft Decline - Primary Funding Source Failed</td>
<td>Soft Decline</td>
<td>A PayPal response indicating the transaction failed due to an issue with primary funding source (e.g. expired Card, insufficient funds, etc.).</td>
</tr>
</tbody>
</table>

**NOTE:** The Response Message associated with Response Code 602 is inaccurate due to a remapping of PayPal Response Codes. Please read the Description below for the recommended action when receiving Response Code 602.

<table>
<thead>
<tr>
<th>602</th>
<th>Soft Decline - Buyer has alternate funding source</th>
<th>Soft Decline</th>
<th>The transaction could not be completed for one of the following reasons:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>• The billing address associated with the financial Instrument could not be confirmed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• The transaction exceeds the card limit.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• The transaction was denied by the card issuer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>You should establish error handling logic that directs the customer to contact PayPal to resolve the issue with their account.</td>
</tr>
<tr>
<td>610</td>
<td>Hard Decline - Invalid Billing Agreement Id</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the Billing Agreement ID is invalid.</td>
</tr>
<tr>
<td>611</td>
<td>Hard Decline - Primary Funding Source Failed</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the issuer is unavailable.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>612</td>
<td>Hard Decline - Issue with Paypal Account</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the transaction failed due to an issue with the buyer account.</td>
</tr>
<tr>
<td>613</td>
<td>Hard Decline - PayPal authorization ID missing</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the need to correct the authorization ID before resubmitting.</td>
</tr>
<tr>
<td>614</td>
<td>Hard Decline - confirmed email address is not available</td>
<td>Hard Decline</td>
<td>A PayPal response indicating your account is configured to decline transactions without a confirmed address. request another payment method or contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a> to modify your account settings.</td>
</tr>
<tr>
<td>615</td>
<td>Hard Decline - PayPal buyer account denied</td>
<td>Hard Decline</td>
<td>A PayPal response indicating account unauthorized payment risk.</td>
</tr>
<tr>
<td>616</td>
<td>Hard Decline - PayPal buyer account restricted</td>
<td>Hard Decline</td>
<td>A PayPal response indicating PayPal is unable to process the payment. Buyer should contact PayPal with questions.</td>
</tr>
<tr>
<td>617</td>
<td>Hard Decline - PayPal order has been voided, expired, or completed</td>
<td>Hard Decline</td>
<td>A PayPal response indicating no further authorizations/captures can be processed against this order. A new order must be created.</td>
</tr>
<tr>
<td>618</td>
<td>Hard Decline - issue with PayPal refund</td>
<td>Hard Decline</td>
<td>A PayPal response indicating one of these potential refund related issues: duplicate partial refund must be less than or equal to original or remaining amount, past time limit, not allowed for transaction type, consumer account locked/inactive, or complaint exists - only a full refund of total/remaining amount allowed. Contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a> for specific details.</td>
</tr>
<tr>
<td>619</td>
<td>Hard Decline - PayPal credentials issue</td>
<td>Hard Decline</td>
<td>A PayPal response indicating you do not have permissions to make this API call.</td>
</tr>
<tr>
<td>620</td>
<td>Hard Decline - PayPal authorization voided or expired</td>
<td>Hard Decline</td>
<td>A PayPal response indicating you cannot capture against this authorization. You need to perform a brand new authorization for the transaction.</td>
</tr>
<tr>
<td>621</td>
<td>Hard Decline - required PayPal parameter missing</td>
<td>Hard Decline</td>
<td>A PayPal response indicating missing parameters are required. Contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a> for specific details.</td>
</tr>
</tbody>
</table>
## TABLE A-1 Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>622</td>
<td>Hard Decline - PayPal transaction ID or auth ID is invalid</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the need to check the validity of the authorization ID prior to reattempting the transaction.</td>
</tr>
<tr>
<td>623</td>
<td>Hard Decline - Exceeded maximum number of PayPal authorization attempts</td>
<td>Hard Decline</td>
<td>A PayPal response indicating you should capture against a previous authorization.</td>
</tr>
<tr>
<td>624</td>
<td>Hard Decline - Transaction amount exceeds merchant's PayPal account limit.</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the transaction amount exceeds the merchant's account limit. Contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a> to modify your account settings.</td>
</tr>
<tr>
<td>625</td>
<td>Hard Decline - PayPal funding sources unavailable.</td>
<td>Hard Decline</td>
<td>A PayPal response indicating the buyer needs to add another funding sources to their account.</td>
</tr>
<tr>
<td>626</td>
<td>Hard Decline - issue with PayPal primary funding source.</td>
<td>Hard Decline</td>
<td>A PayPal response indicating there are issues with the buyer's primary funding source.</td>
</tr>
<tr>
<td>627</td>
<td>Hard Decline - PayPal profile does not allow this transaction type.</td>
<td>Hard Decline</td>
<td>Contact your Relationship Manager to adjust your PayPal merchant profile preferences.</td>
</tr>
<tr>
<td>628</td>
<td>Internal System Error with PayPal - Contact Vantiv</td>
<td>Hard Decline</td>
<td>There is a problem with your username and password. Contact <a href="mailto:eCommerceSupport@vantiv.com">eCommerceSupport@vantiv.com</a>.</td>
</tr>
<tr>
<td>629</td>
<td>Hard Decline - Contact PayPal consumer for another payment method</td>
<td>Hard Decline</td>
<td>A PayPal response indicating you should contact the consumer for another payment method.</td>
</tr>
<tr>
<td>637</td>
<td>Invalid terminal Id</td>
<td>Hard Decline</td>
<td>The terminal Id submitted with the POS transaction is invalid.</td>
</tr>
<tr>
<td>701</td>
<td>Under 18 years old</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating the customer is under 18 years of age based upon the date of birth.</td>
</tr>
<tr>
<td>702</td>
<td>Bill to outside USA</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating the billing address is outside the United States.</td>
</tr>
<tr>
<td>703</td>
<td>Bill to address is not equal to ship to address</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating that the billing address does not match the shipping address.</td>
</tr>
<tr>
<td>704</td>
<td>Declined, foreign currency, must be USD</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating the transaction is declined, because it is not in US dollars.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>705</td>
<td>On negative file</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating the account is on the negative file.</td>
</tr>
<tr>
<td>706</td>
<td>Blocked agreement</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating a blocked agreement account status.</td>
</tr>
<tr>
<td>707</td>
<td>Insufficient buying power</td>
<td>Other</td>
<td>A PayPal Credit response indicating that the account holder does not have sufficient credit available for the transaction amount.</td>
</tr>
<tr>
<td>708</td>
<td>Invalid Data</td>
<td>Hard Decline</td>
<td>A PayPal Credit response indicating that there are one or more problems with the submitted data.</td>
</tr>
</tbody>
</table>
| 709           | Invalid Data - data elements missing | Hard Decline | A PayPal Credit response indicating one or more required data elements are missing. 

Also, returned for an eCheck transaction that is missing a required data element. For example, failure to include the name element in an `echeckSale` or `echeckCredit` transaction would result in this code being returned. |
| 710           | Invalid Data - data format error | Hard Decline | A PayPal Credit response indicating that some data was formatted incorrectly. |
| 711           | Invalid Data - Invalid T&C version | Hard Decline  | A PayPal Credit response indicating the T&C version is invalid. |
| 712           | Duplicate transaction | Hard Decline  | A PayPal Credit response indicating that the transaction is a duplicate. |
| 713           | Verify billing address | Hard Decline  | A PayPal Credit response indicating that you should verify the billing address. |
| 714           | Inactive Account  | Hard Decline  | A PayPal Credit response indicating the customer account is inactive. |
| 716           | Invalid Auth      | Hard Decline  | A PayPal Credit response indicating that the referenced authorization is invalid. |
| 717           | Authorization already exists for the order | Hard Decline  | A PayPal Credit response indicating that an authorization already exists for the transaction. |
| 801           | Account number was successfully registered | Approved | The card number was successfully registered and a token number was returned. |
| 802           | Account number was previously registered | Approved | The card number was previously registered for tokenization. |
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>805</td>
<td>Card Validation Number Updated</td>
<td>Approved</td>
<td>The stored value for CVV2/CVC2/CID has been successfully updated.</td>
</tr>
<tr>
<td>820</td>
<td>Credit card number was invalid</td>
<td>Hard Decline</td>
<td>The card number submitted for tokenization is invalid.</td>
</tr>
<tr>
<td>821</td>
<td>Merchant is not authorized for tokens</td>
<td>Hard Decline</td>
<td>Your organization is not authorized to use tokens.</td>
</tr>
<tr>
<td>822</td>
<td>Token was not found</td>
<td>Hard Decline</td>
<td>The token number submitted with this transaction was not found.</td>
</tr>
<tr>
<td>823</td>
<td>Token Invalid</td>
<td>Hard Decline</td>
<td>The submitted token is invalid.</td>
</tr>
<tr>
<td>825</td>
<td>Merchant not authorized for eCheck tokens</td>
<td>Hard Decline</td>
<td>Your organization is not authorized for eCheck tokenization.</td>
</tr>
<tr>
<td>826</td>
<td>Checkout Id was invalid</td>
<td>Soft Decline</td>
<td>The submitted checkoutId is invalid.</td>
</tr>
<tr>
<td>827</td>
<td>Checkout Id was not found</td>
<td>Soft Decline</td>
<td>The submitted checkoutId was not found.</td>
</tr>
<tr>
<td>828</td>
<td>Generic Checkout Id error</td>
<td>Soft Decline</td>
<td>An unknown error caused the use of checkoutId to fail.</td>
</tr>
<tr>
<td>835</td>
<td>Capture amount can not be more than authorized amount</td>
<td>Hard Decline</td>
<td>The amount in the submitted Capture exceeds 115% of the authorized amount.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Appears in Declined Transaction report.</td>
</tr>
<tr>
<td>850</td>
<td>Tax Billing only allowed for MCC 9311</td>
<td>Hard Decline</td>
<td>Tax Billing elements are allowed only for MCC 9311.</td>
</tr>
<tr>
<td>851</td>
<td>MCC 9311 requires taxType element</td>
<td>Hard Decline</td>
<td>Missing taxType element</td>
</tr>
<tr>
<td>852</td>
<td>Debt Repayment only allowed for VI transactions on MCCs 6012 and 6051</td>
<td>Hard Decline</td>
<td>You must be either MCC 6012 or 6051 to designate a Visa transaction as Debt Repayment (debtRepayment element set to true).</td>
</tr>
<tr>
<td>861</td>
<td>Routing Number did not match one on file for token</td>
<td>Soft Decline</td>
<td>The routing number submitted does not match the number submitted when the token was created. Verify the routing number and resubmit the transaction.</td>
</tr>
<tr>
<td>877</td>
<td>Invalid Pay Page Registration Id</td>
<td>Hard Decline</td>
<td>An eProtect response indicating that the Registration ID submitted is invalid.</td>
</tr>
</tbody>
</table>
### TABLE A-1 Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>878</td>
<td>Expired Pay Page Registration Id</td>
<td>Hard Decline</td>
<td>An eProtect response indicating that the Registration ID has expired (Registration IDs expire 24 hours after being issued).</td>
</tr>
<tr>
<td>879</td>
<td>Merchant is not authorized for Pay Page</td>
<td>Hard Decline</td>
<td>Your organization is not authorized to use eProtect.</td>
</tr>
<tr>
<td>898</td>
<td>Generic token registration error</td>
<td>Soft Decline</td>
<td>There is an unspecified token registration error; contact your Relationship Manager</td>
</tr>
<tr>
<td>899</td>
<td>Generic token use error</td>
<td>Soft Decline</td>
<td>There is an unspecified token use error; contact your Relationship Manager</td>
</tr>
<tr>
<td>900</td>
<td>Invalid Bank Routing Number</td>
<td>Hard Decline</td>
<td>The eCheck routing number submitted with this transaction has failed validation.</td>
</tr>
<tr>
<td>901</td>
<td>Missing Name</td>
<td>Hard Decline</td>
<td>The customer name is required for SEPA transactions.</td>
</tr>
<tr>
<td>902</td>
<td>Invalid Name</td>
<td>Hard Decline</td>
<td>The customer name must be a minimum of two characters for SEPA transactions.</td>
</tr>
<tr>
<td>904</td>
<td>Invalid IBAN</td>
<td>Hard Decline</td>
<td>The submitted International Bank Account number is invalid. Please correct the number and resubmit the transaction.</td>
</tr>
<tr>
<td>905</td>
<td>Missing Email Address</td>
<td>Hard Decline</td>
<td>The customer email address is required for SEPA transactions.</td>
</tr>
<tr>
<td>940</td>
<td>This Funding Instruction results in a negative account balance</td>
<td>Soft Decline</td>
<td>There are insufficient funds in the FBO Settlement Account to cover the Funding Instruction. Wait for additional funds to settle to the account and then resubmit the transaction.</td>
</tr>
<tr>
<td>941</td>
<td>Account balance information unavailable at this time.</td>
<td>Soft Decline</td>
<td>Typically, this response occurs only for new FBO Settlement accounts that do not yet have any settled transactions and were not pre-funded (i.e., no balance yet recorded for the account). Wait for additional funds to settle to the account or pre-fund the account and then resubmit the transaction.</td>
</tr>
<tr>
<td>942</td>
<td>The submitted card is not eligible for Fast Access Funding.</td>
<td>Hard Decline</td>
<td>The card you submitted in the Fast Access Funding instruction cannot receive funds using this method.</td>
</tr>
</tbody>
</table>
### TABLE A-1  Valid Values for the Response and Message Elements (Continued)

<table>
<thead>
<tr>
<th>Response Code</th>
<th>Response Message</th>
<th>Response Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>917</td>
<td>Invalid Billing Country</td>
<td>Hard Decline</td>
<td>Some of the alternative payment methods restrict the allowed consumer’s billing country codes. For example, iDEAL only allows country code NL.</td>
</tr>
<tr>
<td>950</td>
<td>Decline - Negative Information on File</td>
<td>Hard Decline</td>
<td>An eCheck response indicating the account is on the negative file.</td>
</tr>
<tr>
<td>951</td>
<td>Absolute Decline</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that this transaction was declined.</td>
</tr>
<tr>
<td>952</td>
<td>The Merchant Profile does not allow the requested operation</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that your Merchant Profile does not allow the requested operation. Contact your Relationship Manager for additional information.</td>
</tr>
<tr>
<td>953</td>
<td>The account cannot accept ACH transactions</td>
<td>Hard Decline</td>
<td>An eCheck response indicating the customer’s checking account does not accept ACH transactions.</td>
</tr>
<tr>
<td>954</td>
<td>The account cannot accept ACH transactions or site drafts</td>
<td>Hard Decline</td>
<td>An eCheck response indicating the customer’s checking account does not accept ACH transactions or site drafts.</td>
</tr>
<tr>
<td>955</td>
<td>Amount greater than limit specified in the Merchant Profile</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that the dollar amount of this transaction exceeds the maximum amount specified in your Merchant Profile. Contact your Relationship Manager for additional information.</td>
</tr>
<tr>
<td>956</td>
<td>Merchant is not authorized to perform eCheck Verification transactions</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that your organization is not authorized to perform eCheck verifications. Contact your Relationship Manager for additional information.</td>
</tr>
<tr>
<td>957</td>
<td>First Name and Last Name required for eCheck Verifications</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that the first and last name of the customer is required for eCheck verifications.</td>
</tr>
<tr>
<td>958</td>
<td>Company Name required for corporate account for eCheck Verifications</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that the company name is required for verifications on corporate accounts.</td>
</tr>
<tr>
<td>959</td>
<td>Phone number required for eCheck Verifications</td>
<td>Hard Decline</td>
<td>An eCheck response indicating that the phone number of the customer is required for eCheck verifications.</td>
</tr>
<tr>
<td>961</td>
<td>Card Brand token not supported</td>
<td>Hard Decline</td>
<td>This code is returned if the merchant submits a Visa generated token.</td>
</tr>
<tr>
<td>Response Code</td>
<td>Response Message</td>
<td>Response Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------</td>
<td>-------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>962</td>
<td>Private Label Card not supported</td>
<td>Hard Decline</td>
<td>This code is returned if the transaction involves a Visa Private Label card.</td>
</tr>
</tbody>
</table>
## A.2 3DS Authentication Result Codes

Table A-2 contains a list of valid authentication result codes returned by Visa for the Verified by Visa service or MasterCard for the MasterCard SecureCode service. It specifies which authentication result values apply to which order sources.

**NOTE:** The Discover ProtectBuy service only returns one of the following Authentication Result Codes: 0, 1, or 2.

<table>
<thead>
<tr>
<th>Authentication Result Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Order Source - Ecommerce</strong></td>
<td></td>
</tr>
<tr>
<td>Blank</td>
<td>Standard ecommerce or non-ecommerce transactions, not an authentication or attempted authentication. CAVV not present</td>
</tr>
<tr>
<td><strong>Order Source - any</strong></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>CAVV passed verification, but no liability shift because a) ECI was not 5 or 6 or b) the card type is an excluded (e.g., Commercial Card)</td>
</tr>
<tr>
<td><strong>Order Source - 3DSAuthenticated or 3DRequested</strong></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>CAVV data field not properly formatted; verification cannot be performed. Or (for MasterCard) Downgrade to non-3DS transaction, Missing or base64 encoded AAV does not start with j (AAV is invalid for 3DSAuthenticated transaction)</td>
</tr>
<tr>
<td>6</td>
<td>CAVV not verified because Issuer has requested no verification. VisaNet processes as if CAVV is valid.</td>
</tr>
<tr>
<td><strong>Order Source - 3DSAuthenticated</strong></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CAVV failed verification Or (for Visa) TAVV (token authentication verification value) cryptogram failed validation Or (for MasterCard) Downgrade to 3DS Attempted transaction, AAV starts with h and not j</td>
</tr>
<tr>
<td>2</td>
<td>CAVV passed verification Or (for Visa) TAVV cryptogram passed validation</td>
</tr>
</tbody>
</table>
**TABLE A-2**  Authentication Result Codes (Continued)

<table>
<thead>
<tr>
<th>Authentication Result Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Issuer elected to return CAVV verification results and Field 44.13 blank. Value is set by VisaNet; means CAVV Results are valid.</td>
</tr>
</tbody>
</table>

**Order Source - 3DSAttempted**

<table>
<thead>
<tr>
<th>Authentication Result Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 3                          | CAVV passed verification  
or (for Visa)  
DTVV (dynamic token verification value) or Visa-defined format cryptogram failed validation |
| 4                          | CAVV failed verification  
or (for Visa)  
DTVV or Visa-defined format cryptogram passed validation |
| 5                          | Not used |
| 7                          | CAVV failed verification |
| 8                          | CAVV passed verification |
| 9                          | CAVV failed verification; Visa generated CAVV because Issuer ACS was not available. |
| A                          | CAVV passed verification; Visa generated CAVV because Issuer Access Control Server (ACS) was not available. |
| B                          | CAVV passed verification but no liability shift because a) ECI was not 5 or 6 or b) the card type is an excluded (e.g., Commercial Card) |
| C                          | Issuer elected to return CAVV verification results and Field 44.13 blank. Value is set by VisaNet; means CAVV Results are valid |
## A.3 AVS Response Codes

Table A-3 contains a list of AVS response codes that can be returned in the response for a payment transaction. There are some codes that you may never receive. Code your system to expect codes from this list. The description is not included in the response.

### TABLE A-3 AVS Response Codes

<table>
<thead>
<tr>
<th>AVS Response Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>5-Digit zip and address match</td>
</tr>
<tr>
<td>01</td>
<td>9-Digit zip and address match</td>
</tr>
<tr>
<td>02</td>
<td>Postal code and address match</td>
</tr>
<tr>
<td>10</td>
<td>5-Digit zip matches, address does not match</td>
</tr>
<tr>
<td>11</td>
<td>9-Digit zip matches, address does not match</td>
</tr>
<tr>
<td>12</td>
<td>Zip does not match, address matches</td>
</tr>
<tr>
<td>13</td>
<td>Postal code does not match, address matches</td>
</tr>
<tr>
<td>14</td>
<td>Postal code matches, address not verified</td>
</tr>
<tr>
<td>20</td>
<td>Neither zip nor address match</td>
</tr>
<tr>
<td>30</td>
<td>AVS service not supported by issuer</td>
</tr>
<tr>
<td>31</td>
<td>AVS system not available</td>
</tr>
<tr>
<td>32</td>
<td>Address unavailable</td>
</tr>
<tr>
<td>33</td>
<td>General error</td>
</tr>
<tr>
<td>34</td>
<td>AVS not performed</td>
</tr>
<tr>
<td>40</td>
<td>Address failed Worldpay edit checks</td>
</tr>
</tbody>
</table>
A.4 AAVS Response Codes

Table A-4 contains a list of American Express Advanced AVS response codes that can be returned as verification of information supplied in the `<name>`, `<phone>` and/or `<email>` child elements of the `<billToAddress>` element. The system returns the AAVS response code in the `<advancedAVSResult>` child of the `<fraudResult>` element.

The code returned has the following format:

- **1st position** - name match
- **2nd position** - phone match
- **3rd position** - email match

Each position can have one of the following values:

- 0 - No Match (failure)
- 1 - Match
- 2 - Not Sent
- 3 - No Response (unchecked, retry, or service not allowed)

For example, a code of 210 would indicate that the name was not sent, the phone matches, and the email does not match.

You should code your system to parse all codes from this list. The description is not included in the response.

**Table A-4** Advances AVS Response Codes

<table>
<thead>
<tr>
<th>AAVS Response Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>No Match</td>
</tr>
<tr>
<td>001</td>
<td>Email matches, name and phone do not match</td>
</tr>
<tr>
<td>002</td>
<td>Name and phone do not match, email not sent</td>
</tr>
<tr>
<td>003</td>
<td>Name and phone do not match, no response for email</td>
</tr>
<tr>
<td>010</td>
<td>Phone matches, name and email do not match</td>
</tr>
<tr>
<td>011</td>
<td>Phone and email match, name does not match</td>
</tr>
<tr>
<td>012</td>
<td>Phone matches, name does not match, email not sent</td>
</tr>
<tr>
<td>013</td>
<td>Phone matches, name does not match, no response for email</td>
</tr>
<tr>
<td>020</td>
<td>Name and email do not match, phone not sent</td>
</tr>
<tr>
<td>021</td>
<td>Email matches, name does not match, phone not sent</td>
</tr>
<tr>
<td>030</td>
<td>Name and email do not match, no response for phone</td>
</tr>
<tr>
<td>031</td>
<td>Email matches, name does not match, no response for phone</td>
</tr>
</tbody>
</table>
### TABLE A-4  Advances AVS Response Codes

<table>
<thead>
<tr>
<th>AAVS Response Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>033</td>
<td>Name does not match, no response for phone or email</td>
</tr>
<tr>
<td>100</td>
<td>Name matches, phone and email do not match</td>
</tr>
<tr>
<td>101</td>
<td>Name and email match, phone does not match</td>
</tr>
<tr>
<td>102</td>
<td>Name matches, phone does not match, email not sent</td>
</tr>
<tr>
<td>103</td>
<td>Name matches, phone does not match, no response for email</td>
</tr>
<tr>
<td>110</td>
<td>Name and phone match, no match for email</td>
</tr>
<tr>
<td>111</td>
<td>Full match</td>
</tr>
<tr>
<td>112</td>
<td>Name and phone match, email not sent</td>
</tr>
<tr>
<td>113</td>
<td>Name and phone match, no response for email</td>
</tr>
<tr>
<td>120</td>
<td>Name matches, email does not match, phone not sent</td>
</tr>
<tr>
<td>121</td>
<td>Name and email match, phone not sent</td>
</tr>
<tr>
<td>130</td>
<td>Name matches, email does not match, no response for phone</td>
</tr>
<tr>
<td>131</td>
<td>Name and email match, no response for phone</td>
</tr>
<tr>
<td>133</td>
<td>Name matches, no response for phone or email</td>
</tr>
<tr>
<td>200</td>
<td>Name not sent, phone and email do not match</td>
</tr>
<tr>
<td>201</td>
<td>Email matches, phone does not match, name not sent</td>
</tr>
<tr>
<td>202</td>
<td>Phone does not match, name and email not sent</td>
</tr>
<tr>
<td>203</td>
<td>Phone does not match, name not sent, no response for email</td>
</tr>
<tr>
<td>210</td>
<td>Phone matches, email does not match, name not sent</td>
</tr>
<tr>
<td>211</td>
<td>Phone and email match, name not sent</td>
</tr>
<tr>
<td>212</td>
<td>Phone matches, name and email not sent</td>
</tr>
<tr>
<td>213</td>
<td>Phone matches, name not sent, no response for email</td>
</tr>
<tr>
<td>220</td>
<td>Email does not match, name and phone not sent</td>
</tr>
<tr>
<td>221</td>
<td>Email matches, name and phone not sent</td>
</tr>
<tr>
<td>230</td>
<td>Email does not match, name not sent, no response for phone</td>
</tr>
<tr>
<td>231</td>
<td>Email matches, name not sent, no response for phone</td>
</tr>
<tr>
<td>233</td>
<td>Name not sent, no response for phone and email</td>
</tr>
<tr>
<td>AAVS Response Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>300</td>
<td>Phone and email do not match, no response for name</td>
</tr>
<tr>
<td>301</td>
<td>Email matches, phone does not match, no response for name</td>
</tr>
<tr>
<td>302</td>
<td>Phone does not match, no response for name, email not sent</td>
</tr>
<tr>
<td>303</td>
<td>Phone does not match, no response for name and email</td>
</tr>
<tr>
<td>310</td>
<td>Phone matches, email does not match, no response for name</td>
</tr>
<tr>
<td>311</td>
<td>Phone and email match, no response for name</td>
</tr>
<tr>
<td>312</td>
<td>Phone matches, email not sent, no response for name</td>
</tr>
<tr>
<td>313</td>
<td>Phone matches, no response for name and email</td>
</tr>
<tr>
<td>320</td>
<td>Email does not match, phone not sent, no response for name</td>
</tr>
<tr>
<td>321</td>
<td>Email matches, phone not sent, no response for name</td>
</tr>
<tr>
<td>330</td>
<td>Email does not match, no response for name and phone</td>
</tr>
<tr>
<td>331</td>
<td>Email matches, no response for name and phone</td>
</tr>
<tr>
<td>333</td>
<td>No response</td>
</tr>
</tbody>
</table>
A.5 Card Validation Response Codes

Table A-5 contains a normalized list of response codes that can be returned when requesting a card validation check.

- CVV2
- CVC2
- CID

The description is not included in the response.

**NOTE:** For American Express transactions, if the submitted security code does not match, the transaction is declined with a Response Reason Code of 352 - Decline CVV2/CID Fail.

**TABLE A-5** Card Validation Response Codes

<table>
<thead>
<tr>
<th>CVV2/CVC2/CID Response Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Match</td>
</tr>
<tr>
<td>N</td>
<td>No Match</td>
</tr>
<tr>
<td>P</td>
<td>Not Processed</td>
</tr>
<tr>
<td>S</td>
<td>CVV2/CVC2/CID should be on the card, but the merchant has indicated CVV2/CVC2/CID is not present</td>
</tr>
<tr>
<td>U</td>
<td>Issuer is not certified for CVV2/CVC2/CID processing</td>
</tr>
<tr>
<td>&quot;&quot; (empty response)</td>
<td>Check was not done for an unspecified reason</td>
</tr>
</tbody>
</table>
A.6 Advanced Fraud Tools Triggered Rules

This section provides definitions of the triggered rules returned in the Advanced Fraud Results (advancedFraudResults element) section of the response message (see Example below). ThreatMetrix uses the rules triggered by each advanced fraud check to determine the device reputation score, which in turn determines the final review status: Pass, Review, or Fail.

**NOTE:** The rules/descriptions in this document reflect those used in the generic merchant policy. Depending upon the policy configured in your merchant profile, some rules may not apply to you, or additional rules, not defined here, may appear in your results.

**Example:** advancedFraudResults Structure

```
<advancedFraudResults>
  <deviceReviewStatus>pass, fail, review, etc.</deviceReviewStatus>
  <deviceReputationScore>Score Returned from ThreatMetrix</deviceReputationScore>
  <triggeredRule>Triggered Rule #1</triggeredRule>
  .
  .
  .
  <triggeredRule>Triggered Rule #N</triggeredRule>
</advancedFraudResults>
```

**TABLE A-6** Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10PaymentsOnDeviceLocalDay</td>
<td>This device submitted 10 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>10PaymentsOnDeviceLocalHour</td>
<td>This device has submitted 10 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>10PaymentsOnFuzzyDeviceLocalDay</td>
<td>This device appears to have submitted 10 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>10PaymentsOnFuzzyDeviceLocalHour</td>
<td>This device appears to have submitted 10 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>10PaymentsOnTrueIPLocalDay</td>
<td>This True IP submitted 10 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10PaymentsOnTrueIPLocalHour</td>
<td>This True IP submitted 10 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>10PaymentsWithEmailAddressLocalDay</td>
<td>This email address submitted 10 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>15PaymentsWithCustomerIDLocalDay</td>
<td>This customer ID submitted 15 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>15PaymentsWithPaymentCardLocalDay</td>
<td>This payment card submitted 15 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>100PaymentsOnDeviceLocalMonth</td>
<td>This device submitted 100 or more payments in the previous month. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>100PaymentsOnFuzzyDeviceLocalMonth</td>
<td>This device appears to have submitted 100 or more payments in the previous month. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>100PaymentsOnTrueIPLocalMonth</td>
<td>This True IP submitted 100 or more payments in the previous month. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>2ScreenResolutionsPerDeviceGlobalDay</td>
<td>This device used 2 or more screen resolutions in the past day. This is atypical and may indicate misuse.</td>
</tr>
<tr>
<td>20PaymentsOnDeviceLocalDay</td>
<td>This device submitted 20 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>20PaymentsOnFuzzyDeviceLocalDay</td>
<td>This device appears to have submitted 20 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>20PaymentsOnTrueIPLocalDay</td>
<td>This True IP submitted 20 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>20PaymentsWithCustomerIDLocalWeek</td>
<td>This customer ID submitted 20 or more payments in the previous week. This is atypical and may be an indicator of misuse.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20PaymentsWithPaymentCardLocalWeek</td>
<td>This payment card submitted 20 or more payments in the previous week. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>3CustomerIDsPerDeviceLocalDay</td>
<td>This device submitted transactions using 3 or more distinct customer IDs in the previous 24 hours. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3CustomerIDsPerDeviceLocalWeek</td>
<td>This device submitted transactions using 3 or more distinct customer IDs in the previous 7 days. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3DevicesPerCustomerIDLocalDay</td>
<td>Three or more devices have been used to submit transactions with this customer ID in the previous 24 hours. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3DevicesPerCustomerIDLocalWeek</td>
<td>Three or more devices have been used to submit transactions with this customer ID in the previous 7 days. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3DevicesPerEmailGlobalDay</td>
<td>Three or more devices have been used to submit transactions with this email address in the previous 24 hours. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3DevicesPerEmailGlobalWeek</td>
<td>Three or more devices have been used to submit transactions with this email address in the previous 7 days. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3DevicesPerPaymentCardGlobalDay</td>
<td>Three or more devices have been used to submit transactions with this payment card in the previous 24 hours. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3DevicesPerPaymentCardGlobalWeek</td>
<td>Three or more devices have been used to submit transactions with this payment card in the previous 7 days. This may be an indicator of misuse.</td>
</tr>
<tr>
<td>3EmailsPerDeviceGlobalDay</td>
<td>This device has submitted transactions using 3 or more distinct email addresses in the previous 24 hours. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
</tbody>
</table>
# Payment Transaction Response Codes

**TABLE A-6** Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3EmailsPerDeviceGlobalWeek</td>
<td>This device has submitted transactions using 3 or more distinct email addresses in the previous 7 days. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3EmailsPerFuzzyDeviceLocalHour</td>
<td>This device appears to have submitted transactions using 3 or more distinct email addresses in the previous 60 minutes. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3PaymentCardsPerDeviceGlobalDay</td>
<td>This device has submitted transactions using 3 or more distinct payment cards in the previous 24 hours. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3PaymentCardsPerDeviceGlobalWeek</td>
<td>This device has submitted transactions using 3 or more distinct payment cards in the previous 7 days. This is abnormal and may be an indicator of a card-testing attack or free-trial abuse.</td>
</tr>
<tr>
<td>3PaymentsOnDeviceLocalHour</td>
<td>This device has submitted 3 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>3PaymentsOnFuzzyDeviceLocalHour</td>
<td>This device appears to have submitted 3 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>3PaymentsOnTrueIPLocalHour</td>
<td>This True IP submitted 3 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>3ProxiesPerDeviceGlobalDay</td>
<td>This device submitted transactions through 3 or more distinct IP proxies in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>4ScreenResolutionsPerDeviceGlobalDay</td>
<td>This device has used 4 or more screen resolutions in the past day. This is atypical and may indicate misuse.</td>
</tr>
<tr>
<td>5PaymentsOnDeviceLocalDay</td>
<td>This device has submitted 5 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsOnDeviceLocalHour</td>
<td>This device has submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>Triggered Rule Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>5PaymentsOnFuzzyDeviceLocalDay</td>
<td>This device appears to have submitted 5 or more payments in the previous 24 hours. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsOnFuzzyDeviceLocalHour</td>
<td>This device appears to have submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsOnTrueIPLocalDay</td>
<td>This True IP has submitted 5 or more payments in the previous day. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsOnTrueIPLocalHour</td>
<td>This True IP submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsWithCustomerIDLocalHour</td>
<td>This customer ID submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsWithEmailAddressLocalHour</td>
<td>This email address has submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>5PaymentsWithPaymentCardLocalHour</td>
<td>This payment card submitted 5 or more payments in the previous hour. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>50PaymentsOnDeviceLocalWeek</td>
<td>This device has submitted 50 or more payments in the previous week. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>50PaymentsOnFuzzyDeviceLocalWeek</td>
<td>This device appears to have submitted 50 or more payments in the previous week. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>50PaymentsOnTrueIPLocalMonth</td>
<td>This True IP has submitted 50 or more payments in the previous month. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>20PaymentsWithCustomerIDLocalMonth</td>
<td>This customer ID submitted 20 or more payments in the previous month. This is atypical and may be an indicator of misuse.</td>
</tr>
</tbody>
</table>
| 20PaymentsWithPaymentCardLocalWeek              | This payment card submitted 20 or more payments in the previous week. This is atypical and may be an indicator of misuse.
<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnonymousProxy</td>
<td>This transaction was submitted through an anonymous web proxy, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>AnonymousProxyIP</td>
<td>This transaction was submitted through an anonymous proxy IP Address, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>BINCustomerAddressGeolocationMismatch</td>
<td>The customer's bill-to address country does not match that of the payment card's issuing bank. This may be an indicator of a fraud attack.</td>
</tr>
<tr>
<td>ComputerGeneratedEmail</td>
<td>This email address may have been automatically generated by a computer. Fraudsters frequently employ automated bots that create email addresses programmatically to enable their fraud attacks.</td>
</tr>
<tr>
<td>CookiesDisabled</td>
<td>The browser used to submit this transaction has disabled cookies. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>CookiesJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled cookies and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>CustomAttribute1OnLocalBlacklist</td>
<td>Custom attribute 1 for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomAttribute1OnLocalWhitelist</td>
<td>Custom attribute 1 for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomAttribute2OnLocalBlacklist</td>
<td>Custom attribute 2 for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomAttribute2OnLocalWhitelist</td>
<td>Custom attribute 2 for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomAttribute3OnLocalBlacklist</td>
<td>Custom attribute 3 for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomAttribute3OnLocalWhitelist</td>
<td>Custom attribute 3 for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomAttribute4OnLocalBlacklist</td>
<td>Custom attribute 4 for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>Triggered Rule Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CustomAttribute4OnLocalWhitelist</td>
<td>Custom attribute 4 for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomAttribute5OnLocalBlacklist</td>
<td>Custom attribute 5 for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomAttribute5OnLocalWhitelist</td>
<td>Custom attribute 5 for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomerIDOnLocalBlacklist</td>
<td>The customer ID for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomerIDOnLocalWhitelist</td>
<td>The customer ID for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>CustomerNameOnLocalBlacklist</td>
<td>The customer name for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>CustomerNameOnLocalWhitelist</td>
<td>The customer name for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>DeviceCountriesNotAllowed</td>
<td>This transaction originated from an IP address located in a country on ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>DeviceIDOnThreatMetrixGlobalBlacklist</td>
<td>The originating device is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>DeviceGlobalAgeLessThanOneHour</td>
<td>The originating device was first seen across the entire ThreatMetrix global network within the past hour. This is uncommon and may point to a fraudster simulating a new device through advanced techniques in an attempt to avoid detection.</td>
</tr>
<tr>
<td>DeviceLocalAgeLessThanOneHour</td>
<td>The originating device was first seen by Worldpay within the past hour. This may point to a fraudster simulating a new device through advanced techniques in an attempt to avoid detection.</td>
</tr>
<tr>
<td>DeviceNotFingerprinted</td>
<td>ThreatMetrix could not fingerprint the originating device. This is atypical and may indicate a deliberate attempt by the user to cloak his or her identity.</td>
</tr>
<tr>
<td>DeviceOnLocalBlacklist</td>
<td>The originating device is on ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>DeviceOnLocalWhitelist</td>
<td>The originating device is on ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>Triggered Rule Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DeviceOnThreatMetrixGlobalBlacklist</td>
<td>The originating device is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>DeviceRejectedByNetwork10Times</td>
<td>The originating device has been rejected by one of ThreatMetrix's customers and/or partners 10 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceRejectedByNetwork25Times</td>
<td>The originating device has been rejected by one of ThreatMetrix's customers and/or partners 25 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceRejectedByNetwork5Times</td>
<td>The originating device has been rejected by one of ThreatMetrix's customers and/or partners 5 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceRejectedByNetworkInLastWeek</td>
<td>The originating device has been rejected by one of ThreatMetrix's customers and/or partners in the last week on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceReviewedByNetwork5Times</td>
<td>The originating device has been reviewed by one of ThreatMetrix's customers and/or partners 5 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceReviewedByNetwork10Times</td>
<td>The originating device has been reviewed by one of ThreatMetrix's customers and/or partners 10 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>DeviceReviewedByNetwork25Times</td>
<td>The originating device has been reviewed by one of ThreatMetrix's customers and/or partners 25 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>EmailDistanceTraveled</td>
<td>This email address has been associated with transactions originating from locations at least 1,000 miles apart in the last hour. This is a red flag and warrants caution.</td>
</tr>
<tr>
<td>EmailHostnameTooLong</td>
<td>The hostname portion (i.e. to the right of &quot;@&quot;) of this email address exceeds 30 characters. Email addresses associated with suspicious domain names are often used as part of attacks. Overly long hostnames are a common marker of such domains.</td>
</tr>
<tr>
<td>EmailHostnameWithNonLetters</td>
<td>The hostname portion (i.e. to the right of &quot;@&quot;) of this email address contains non-letter characters (e.g. numbers and special characters). Email addresses associated with suspicious domain names are often used as part of attacks. Hostnames with non-letter characters are a common marker of such domains.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EmailOnLocalBlacklist</td>
<td>The customer email address for this transaction is on Worldpay’s ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>EmailOnLocalWhitelist</td>
<td>The customer email address for this transaction is on Worldpay’s ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>EmailOnThreatMetrixGlobalBlacklist</td>
<td>This email address is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>EmailRejectedByNetwork10TimesInLastDay</td>
<td>The associated email address has been rejected by one of ThreatMetrix’s customers and/or partners 10 or more times in the last day on the suspicion of fraud.</td>
</tr>
<tr>
<td>EmailRejectedByNetworkInLastWeek</td>
<td>A transaction using this email address has been rejected by one of ThreatMetrix’s customers and/or partners in the last week on the suspicion of fraud.</td>
</tr>
<tr>
<td>EmailUsernameTooLong</td>
<td>The name portion (i.e. to the left of &quot;@&quot;) of this email address exceeds 30 characters. Email addresses associated with suspicious user names are often used as part of attacks. Overly long user names are a common marker of such domains.</td>
</tr>
<tr>
<td>EmailUsernameWithNonLetters</td>
<td>The name portion (i.e. to the left of &quot;@&quot;) of this email address contains non-letter characters (e.g. numbers and special characters). Email addresses associated with suspicious usernames are often used as part of attacks. Usernames with non-letter characters are a common marker of such domains.</td>
</tr>
<tr>
<td>ExcessivePaymentsOnDeviceHour</td>
<td>An abnormally high number of transactions have been submitted from this device in the last hour. This is a common indicator of fraudulent payment.</td>
</tr>
<tr>
<td>ExcessivePaymentsOnDeviceDay</td>
<td>An abnormally high number of transactions have been submitted from this device in the last 24 hours. This is a common indicator of fraudulent payment.</td>
</tr>
<tr>
<td>ExcessivePaymentsOnFuzzyDeviceHour</td>
<td>An abnormally high number of transactions appear to have been submitted from this device in the last hour. This is a common indicator of fraudulent payment.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ExcessivePaymentsOnFuzzyDeviceDay</td>
<td>An abnormally high number of transactions appear to have been submitted from this device in the last 24 hours. This is a common indicator of fraudulent payment.</td>
</tr>
<tr>
<td>FlashBrowserLanguageMismatch</td>
<td>The language used by the web browser used to submit this transaction does not match the language used by the Flash plug-in. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashCook1esJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects, cookies, and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashCookiesDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects and cookies. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashImagesCookiesDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects, images, and cookies. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashImagesCookiesJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects, images, cookies, and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashImagesDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects and images. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashImagesJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects, images, and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>FlashJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled Flash objects and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FuzzyDeviceLocalAgeLessThanOneHour</td>
<td>The originating device appears to have been seen by Worldpay for the first time within the past hour. This may point to a fraudster simulating a new device through advanced techniques in an attempt to avoid detection.</td>
</tr>
<tr>
<td>FuzzyDeviceOnLocalBlacklist</td>
<td>The originating device appears to be on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>FuzzyDeviceOnLocalWhitelist</td>
<td>The originating device appears to be on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>FuzzyDeviceOnThreatMetrixGlobalBlklst</td>
<td>The originating device appears to be on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>FuzzyDeviceRejectedByNetworkInLastWeek</td>
<td>The originating device appears to have been rejected by one of ThreatMetrix's customers and/or partners in the last week on the suspicion of fraud.</td>
</tr>
<tr>
<td>GeolocationLanguageMismatch</td>
<td>The language detected from the originating web browser is not appropriate for the location. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>HiddenProxy</td>
<td>This transaction was submitted through a hidden web proxy, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>ImagesCookiesDisabled</td>
<td>The browser used to submit this transaction has disabled images and cookies. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>ImagesCookiesJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled images, cookies, and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>ImagesDisabled</td>
<td>The browser used to submit this transaction has disabled images. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>ImagesJavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled images and JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>IPHasNegativeReputation</td>
<td>The originating IP address is a potential threat based upon analysis of its activity across the ThreatMetrix network.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPOnLocalBlacklist</td>
<td>The originating IP address is on ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>IPOnLocalWhitelist</td>
<td>The originating IP address is on ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>IPOnThreatMetrixGlobalBlacklist</td>
<td>The originating IP address is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>IPRRejectedByNetwork10Times</td>
<td>The originating IP address has been rejected by one of ThreatMetrix’s customers and/or partners 10 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>IPRRejectedByNetwork25Times</td>
<td>The originating IP address has been rejected by one of ThreatMetrix’s customers and/or partners 25 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>IPRRejectedByNetwork5Times</td>
<td>The originating IP address has been rejected by one of ThreatMetrix’s customers and/or partners 5 or more times on the suspicion of fraud.</td>
</tr>
<tr>
<td>JavascriptDisabled</td>
<td>The browser used to submit this transaction has disabled JavaScript. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>KnownVPNISP</td>
<td>This transaction was submitted through a known Virtual Private Network (VPN), a method that is sometimes employed when trying to cloak one’s identity.</td>
</tr>
<tr>
<td>MalwareDetectedOnDevice</td>
<td>The originating device appears to have been infected with malware.</td>
</tr>
<tr>
<td>OpenProxy</td>
<td>This transaction was submitted through an open web proxy, a method that is sometimes employed when trying to cloak one’s identity.</td>
</tr>
<tr>
<td>PaymentCardBINShippingAddressGeolocationMismatch</td>
<td>The customer’s ship-to address country does not match that of the payment card’s issuing bank. This may be an indicator of a fraud attack.</td>
</tr>
<tr>
<td>PaymentCardBINTrueIPGeolocationMismatch</td>
<td>The geolocation of the True IP address does not match that of the payment card’s issuing bank. This may be an indicator of a fraud attack.</td>
</tr>
<tr>
<td>PaymentCardDistanceTraveled</td>
<td>This payment card has been associated with transactions originating from locations at least 1,000 miles apart in the last hour. This is a red flag and warrants caution.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PaymentCardOnThreatMetrixGlobalBlacklist</td>
<td>This payment card is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>PaymentCardRejectedByNetworkInLastWeek</td>
<td>A transaction using this payment card has been rejected by one of ThreatMetrix's customers and/or partners in the last week on the suspicion of fraud.</td>
</tr>
<tr>
<td>PhoneNumberOnLocalBlacklist</td>
<td>The customer telephone number for this transaction is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>PhoneNumberOnLocalWhitelist</td>
<td>The customer telephone number for this transaction is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>PossibleCookieWipingDay</td>
<td>The user appears to have cleared his or her browser's cookies 3 or more times in the last day. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>PossibleCookieWipingHour</td>
<td>The user appears to have cleared his or her browser's cookies 3 or more times in the last hour. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>PossibleCookieWipingWeek</td>
<td>The user appears to have cleared his or her browser's cookies 3 or more times in the last week. This is common to fraud attacks and may be an indicator of misuse.</td>
</tr>
<tr>
<td>PossibleVPNOrTunnel</td>
<td>This transaction may have been submitted through a Virtual Private Network (VPN), a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>PossibleVPNConnection</td>
<td>This transaction may have been submitted through a Virtual Private Network (VPN), a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>PotentialVirtualMachine</td>
<td>This transaction may have been submitted using a Virtual Machine, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>ProxyHasNegativeReputation</td>
<td>The originating IP proxy is a potential threat based upon an analysis of its activity across the ThreatMetrix network.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProxyIPHasNegativeReputation</td>
<td>The originating IP address is a potential threat based upon analysis of its activity across the ThreatMetrix network.</td>
</tr>
<tr>
<td>ProxyIPOnLocalBlacklist</td>
<td>The originating proxy IP address is on Worldpay's ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>ProxyIPOnLocalWhitelist</td>
<td>The originating proxy IP address is on Worldpay's ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>ProxyIPOnThreatMetrixGlobalBlacklist</td>
<td>The originating proxy IP address is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>SatelliteISP</td>
<td>This transaction was submitted through a Satellite Internet Service Provider, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>SatelliteProxyISP</td>
<td>This transaction was submitted through a Satellite Proxy Internet Service Provider, a method that is sometimes employed when trying to cloak one's identity.</td>
</tr>
<tr>
<td>SessionAnomaly</td>
<td>Characteristics of the originating device appear to have been modified during the course of the user's web session. This is atypical and may indicate misuse.</td>
</tr>
<tr>
<td>ShippingAddressTrueIPGeolocationMismatch</td>
<td>The shipping address country does not match that of the True IP address. This may be an indicator of a package redirection/interception/forwarding or re-shipping fraudster attack.</td>
</tr>
<tr>
<td>SuspectedSessionCloaking</td>
<td>The characteristics of the originating browser are consistent with common fraud attacks, and may be an indicator of a fraudster's deliberate attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>SuspectedTORNetwork</td>
<td>This transaction appears to have originated from a TOR network, a common source of fraud attacks.</td>
</tr>
<tr>
<td>SystemStateAnomaly</td>
<td>The system state of the originating device has changed two or more times within the past hour. This is atypical and may indicate misuse.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeZoneTrueGeolocationMismatch</td>
<td>The time zone setting on the originating device does not match to the true geolocation of the customer. This is atypical and may be an indicator of misuse.</td>
</tr>
<tr>
<td>TransparentProxy</td>
<td>This transaction was submitted through a transparent web proxy, a method that is most often used in corporate environments, though also employed when trying to cloak one’s identity.</td>
</tr>
<tr>
<td>TrueIPDNSGeolocationMismatch</td>
<td>The geolocation of the True IP address does not match that of the DNS provider. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>TrueIPHasNegativeReputation</td>
<td>The originating IP address is a potential threat based upon analysis of its activity across the ThreatMetrix network.</td>
</tr>
<tr>
<td>TrueIPOnLocalBlacklist</td>
<td>The originating true IP address is on Worldpay’s ThreatMetrix-hosted blacklist.</td>
</tr>
<tr>
<td>TrueIPOnLocalWhitelist</td>
<td>The originating true IP address is on Worldpay’s ThreatMetrix-hosted whitelist.</td>
</tr>
<tr>
<td>TrueIPOnThreatMetrixGlobalBlacklist</td>
<td>The originating true IP address is on the ThreatMetrix global blacklist.</td>
</tr>
<tr>
<td>TrueIPProxyIPCityMismatch</td>
<td>The city of the True IP address does not match that of the Proxy IP address. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>TrueIPProxyIPGeolocationMismatch</td>
<td>The geolocation of the True IP address does not match that of the Proxy IP address. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>TrueIPProxyIPISPMismatch</td>
<td>The Internet Service Provider of the True IP address does not match that of the Proxy IP address. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>TrueIPProxyIPOrganizationMismatch</td>
<td>The organization of the True IP address does not match that of the Proxy IP address. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
</tbody>
</table>
### TABLE A-6  Triggered Rules Definitions

<table>
<thead>
<tr>
<th>Triggered Rule Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TrueIPProxyIPRegionMismatch</td>
<td>The region of the True IP address does not match that of the Proxy IP address. This may be an indicator of a fraudster's attempt to cloak his or her identity.</td>
</tr>
<tr>
<td>TrueIPRejectedByNetwork10TimesInLastDay</td>
<td>The originating IP address has been rejected by one of ThreatMetrix's customers and/or partners 10 or more times in the last day on the suspicion of fraud.</td>
</tr>
<tr>
<td>TrueIPRejectedByNetworkInLastWeek</td>
<td>The originating true IP has been rejected by one of ThreatMetrix's customers and/or partners in the last week on the suspicion of fraud.</td>
</tr>
<tr>
<td>UnusualProxyAttributes</td>
<td>This web proxy used to submit the transaction has unusual attributes (e.g. dialup), which may indicate an attempt to cloak one's identity.</td>
</tr>
</tbody>
</table>
## A.7 XML Validation Error Messages

Table A-7 provides examples of XML Validation Error Messages. These messages are the value associated with the `message` attribute of either a `litleResponse` or `litleOnlineResponse`, when the `response` attribute is “1”.

**NOTE:** If `response` is “0”, the associated message is “Valid Format”.

For information about response values 2 through 5, please refer to Additional Response Header Error Messages on page 831.

### TABLE A-7 Response Header Error Message Examples

<table>
<thead>
<tr>
<th>Example Message (message attribute of litleOnlineResponse)</th>
<th>Description (line numbers will vary according to the location of the error)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error validating xml data against the schema on line 13. The length of the value is 3, but the required minimum is 4.</td>
<td>The value on line 13 does not meet the minimum length requirement for the element as specified in the schema. For example, if you specified 812 as a value for the <code>&lt;expDate&gt;</code> element, which has a minLength of 4, the system returns this error message.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 18. The length of the value is 6, but the required maximum is 4.</td>
<td>The value on line 18 exceeds the maximum length requirement for the element as specified in the schema. For example, if you specified 082012 as a value for the <code>&lt;expDate&gt;</code> element, which has a maxLength of 4, the system returns this error message.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 11. The value is not a member of the enumeration.</td>
<td>The value on line 11 is not a valid enumeration for the specified element. For example, The <code>&lt;type&gt;</code> element allows values of VI, MC, DI, AX, DC, JC, PP and BL. If you submitted a value of VISA, the system returns this error message.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 8. Content of element “amount” is incomplete.</td>
<td>The <code>&lt;amount&gt;</code> element does not contain a valid value. For example, if you submitted a captureGivenAuth request and included the amount element without specifying a value, the system returns this error message.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 6 tag name “echeckSale” is not allowed. Possible tag names are: <code>&lt;authorization&gt;</code>, <code>&lt;capture&gt;</code>, <code>&lt;credit&gt;</code>, <code>&lt;sale&gt;</code>, <code>&lt;void&gt;</code></td>
<td>The submitted transaction failed validation against the schema, because an element name was out of sequence or not allowed in the transaction. The error message specifies the invalid element (<code>&lt;echeckSale&gt;</code> in the example), as well as the possible valid elements.</td>
</tr>
<tr>
<td>Example Message (message attribute of littleOnlineResponse)</td>
<td>Description (line numbers will vary according to the location of the error)</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>System Error - Call Vantiv</td>
<td>Typically, the system returns this error if there was a problem with authentication due to an error in the submitted Merchant Id, user, and/or password. The problem may also be due to the use of single quotes around the attribute (merchantId) value.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 1. Probably namespace URI of tag &quot;littleOnlineRequest&quot; is wrong (correct one is &quot;<a href="http://www.litle.com/schema">http://www.litle.com/schema</a>&quot;)</td>
<td>The URI named in the xmlns= attribute is incorrect. The problem may also be due to the use of single quotes around the attribute value. <strong>Note:</strong> The URI may differ based upon the version of cnpAPI you are using.</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 12786. The entity name must immediately follow the '&amp;' in the entity reference.</td>
<td>The '&amp;' symbol is used in XML to designate certain special characters. The error indicates that the symbol was submitted without an entity name (for example, &quot; or &amp; ). Typically, the error occurs when the name or one of the address lines of the billToAddress element includes the symbol instead of the entity reference. For example, “John &amp; Mary Smith” should be sent as “John &amp; Mary Smith” or “John and Mary Smith”).</td>
</tr>
<tr>
<td>Error validating xml data against the schema on line 1. Content is not allowed in prolog.</td>
<td>This error is usually an indication of extraneous characters appearing in front of the first XML element. For example, the “?” before the “&gt;” symbol in the following line would cause this error to be returned: ?&lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-8&quot;&gt;</td>
</tr>
<tr>
<td>Duplicate Batch (Batch ID: 28292109643, session sequence: 1, unique ID:) not processed - 29 duplicate transactions (57 total) in a row found. Duplicate Batch (Batch ID: 23829210964, session sequence: 1, unique ID:) not processed - 96.49% of the transactions (57 total) in the batch are duplicates.</td>
<td><strong>(Batch only)</strong> The system has determined that the Batch is a duplicate and therefore not processed. The first part of the message provides the count of the greatest number of consecutive duplicate transaction in the batch (29 in the example). The second part of the message the overall percentage of duplicates in the batch (96.49% in the example). The limits are more than 10 consecutive duplicate transactions detected and/or more than 25% of all transaction in the batch detected as duplicates.</td>
</tr>
</tbody>
</table>
A.8 Additional Response Header Error Messages

When submitting transactions via Open Access, there are additional HTTP responses and validation errors that may occur. The table below provides information about these responses/error messages, as well as some additional errors you may encounter during Online transaction processing.

**NOTE:** The response value and message in the table represent the values for the `response` and `message` attributes of either a `litleResponse` or `litleOnlineResponse`.

If `Resubmit?` = No, you must debug and modify the submitted message prior to resubmission in order to have any chance of success. Resubmitting the same message will result in the same failure.

<table>
<thead>
<tr>
<th>response value</th>
<th>message</th>
<th>HTTP Status Code/Message</th>
<th>Description</th>
<th>Resubmit?</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Invalid XML. Contact ecommercesupport @worldpay.com.</td>
<td>200 OK</td>
<td>The submission is not valid XML containing the <code>user</code> and <code>password</code> elements.</td>
<td>No - debug or contact Worldpay Support</td>
</tr>
<tr>
<td>3</td>
<td>Invalid credentials. Contact ecommercesupport @worldpay.com.</td>
<td>200 OK</td>
<td>The submission contains empty or invalid credentials (<code>user</code> and <code>password</code>).</td>
<td>Maybe - verify credentials</td>
</tr>
<tr>
<td>4</td>
<td>Connection limit exceeded. Contact ecommercesupport @worldpay.com.</td>
<td>200 OK</td>
<td>The merchant has exceeded the maximum number of concurrent connections.</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Objectionable content detected. Contact ecommercesupport @worldpay.com.</td>
<td>200 OK</td>
<td>The system has determined that the submission may contain objectionable content or the message exceeds 20K maximum characters.</td>
<td>Reduce the size of the message below the 20K limit and/or contact Worldpay Support</td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>403 Method Not Allowed</td>
<td>The merchant has exceeded the maximum number of concurrent connections.</td>
<td>Yes</td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>405 Method Not Allowed</td>
<td>Only HTTP POST method is allowed.</td>
<td>No - debug or contact Worldpay Support</td>
</tr>
</tbody>
</table>
### TABLE A-8  HTTP Status Message and Validation Errors

<table>
<thead>
<tr>
<th>response value</th>
<th>message</th>
<th>HTTP Status Code/Message</th>
<th>Description</th>
<th>Resubmit?</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>404 Not Found</td>
<td>An invalid URI was used. Verify the URI you are using is correct and that you have not appended any parameters to the URI.</td>
<td>No - debug or contact Worldpay Support</td>
</tr>
<tr>
<td>N/A</td>
<td>N/A</td>
<td>417 Expectation Failed</td>
<td>An HTTP <strong>Expect</strong> header was included in the HTTP POST, which is not allowed.</td>
<td>No - debug or contact Worldpay Support</td>
</tr>
</tbody>
</table>
A.9 ACH Return Reason Codes

Table A-9 is a list of ACH Return Reason Codes, which can apply to either eCheck transactions, or Dynamic Payout funding instructions. These codes are not returned in the cnpAPI response messages, but are visible in iQ on the eCheck Returns Received report, as well as the Payment Detail screen and the Funding Instruction Detail screen.

NOTE: If an eCheck is returned for reason Code R01 or R09, it is eligible for redeposit.

---

Table A-9 eCheck Return Reason Codes

<table>
<thead>
<tr>
<th>ACH Return Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R01</td>
<td>Insufficient funds in account</td>
</tr>
<tr>
<td>R02</td>
<td>Account is closed</td>
</tr>
<tr>
<td>R03</td>
<td>No account on file</td>
</tr>
<tr>
<td>R04</td>
<td>Invalid account number</td>
</tr>
<tr>
<td>R05</td>
<td>Unauthorized debit to consumer account</td>
</tr>
<tr>
<td>R06</td>
<td>Returned at request of ODFI</td>
</tr>
<tr>
<td>R07</td>
<td>Authorization revoked by customer</td>
</tr>
<tr>
<td>R08</td>
<td>Payment stopped</td>
</tr>
<tr>
<td>R09</td>
<td>Insufficient collected funds in account being charged</td>
</tr>
<tr>
<td>R10</td>
<td>Customer advises not Authorized, notice not provided, improper source document, or amount of entry not accurately obtained from source document</td>
</tr>
<tr>
<td>R11</td>
<td>Check truncation return</td>
</tr>
<tr>
<td>R12</td>
<td>Account sold to another financial institution</td>
</tr>
<tr>
<td>R13</td>
<td>Invalid ACH routing number</td>
</tr>
<tr>
<td>R14</td>
<td>Representative payee is deceased or cannot continue in that capacity</td>
</tr>
<tr>
<td>R15</td>
<td>Beneficiary or account holder other than representative payee deceased</td>
</tr>
<tr>
<td>R16</td>
<td>Account funds have been frozen</td>
</tr>
<tr>
<td>R17</td>
<td>Item returned because of invalid data; refer to addenda for invalid information</td>
</tr>
<tr>
<td>R18</td>
<td>Improper effective date</td>
</tr>
</tbody>
</table>
### TABLE A-9  eCheck Return Reason Codes

<table>
<thead>
<tr>
<th>ACH Return Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R19</td>
<td>Amount error</td>
</tr>
<tr>
<td>R20</td>
<td>Account does not allow ACH transactions or limit for transactions has been exceeded</td>
</tr>
<tr>
<td>R21</td>
<td>Invalid company identification</td>
</tr>
<tr>
<td>R22</td>
<td>Invalid individual ID</td>
</tr>
<tr>
<td>R23</td>
<td>Credit entry refused by receiver</td>
</tr>
<tr>
<td>R24</td>
<td>Duplicate entry</td>
</tr>
<tr>
<td>R25</td>
<td>Addenda record error</td>
</tr>
<tr>
<td>R26</td>
<td>Mandatory field error</td>
</tr>
<tr>
<td>R27</td>
<td>Trace number error</td>
</tr>
<tr>
<td>R28</td>
<td>Routing/transit number check digit error</td>
</tr>
<tr>
<td>R29</td>
<td>Corporate customer advised not authorized</td>
</tr>
<tr>
<td>R30</td>
<td>RDFI not participant in check truncation program</td>
</tr>
<tr>
<td>R31</td>
<td>Permissible return entry</td>
</tr>
<tr>
<td>R32</td>
<td>RDFI non-settlement</td>
</tr>
<tr>
<td>R33</td>
<td>Return of item</td>
</tr>
<tr>
<td>R34</td>
<td>Limited participation ODFI</td>
</tr>
<tr>
<td>R35</td>
<td>Return of improper debit entry</td>
</tr>
<tr>
<td>R36</td>
<td>Return of improper credit entry</td>
</tr>
<tr>
<td>R37</td>
<td>Source document presented for payment</td>
</tr>
<tr>
<td>R38</td>
<td>Stop payment on source document</td>
</tr>
<tr>
<td>R39</td>
<td>Improper source document</td>
</tr>
<tr>
<td>R40</td>
<td>Return of item by government agency</td>
</tr>
<tr>
<td>R41</td>
<td>Invalid Transaction Code</td>
</tr>
<tr>
<td>R42</td>
<td>Routing/transit number check digit error</td>
</tr>
<tr>
<td>R43</td>
<td>Invalid account number</td>
</tr>
<tr>
<td>R44</td>
<td>Invalid individual ID</td>
</tr>
<tr>
<td>R45</td>
<td>Invalid individual name or company name</td>
</tr>
<tr>
<td>R46</td>
<td>Invalid representative payee indicator code</td>
</tr>
<tr>
<td>R47</td>
<td>Duplicate enrollment</td>
</tr>
</tbody>
</table>
## TABLE A-9  eCheck Return Reason Codes

<table>
<thead>
<tr>
<th>ACH Return Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R50</td>
<td>State law affecting RCK acceptance</td>
</tr>
<tr>
<td>R51</td>
<td>Item is ineligible, notice not provided, signature not genuine, or original item altered for adjustment entry</td>
</tr>
<tr>
<td>R52</td>
<td>Stop payment on item</td>
</tr>
<tr>
<td>R53</td>
<td>Item and ACH entry presented for payment</td>
</tr>
<tr>
<td>R61</td>
<td>Misrouted return - RDFI for original entry has placed incorrect routing/transit number in RDFI identification field</td>
</tr>
<tr>
<td>R67</td>
<td>Duplicate return</td>
</tr>
<tr>
<td>R68</td>
<td>Untimely return - return was not sent within the established time frame</td>
</tr>
<tr>
<td>R69</td>
<td>Field errors</td>
</tr>
<tr>
<td>R70</td>
<td>Permissible return entry not accepted</td>
</tr>
<tr>
<td>R71</td>
<td>Misrouted dishonored return -incorrect routing/transit number in RDFI identification field</td>
</tr>
<tr>
<td>R72</td>
<td>Untimely return - dishonored return was not sent within the established time frame</td>
</tr>
<tr>
<td>R73</td>
<td>Timely original return - RDFI certifies the original return entry was sent within established time frame for original returns</td>
</tr>
<tr>
<td>R74</td>
<td>Corrected return - RDFI is correcting a previous return entry that was dishonored because it contained incomplete or incorrect information</td>
</tr>
<tr>
<td>R75</td>
<td>Original return not a duplicate</td>
</tr>
<tr>
<td>R76</td>
<td>No errors found</td>
</tr>
<tr>
<td>R80</td>
<td>Cross-border payment coding error</td>
</tr>
<tr>
<td>R81</td>
<td>Non-participant in cross-border program</td>
</tr>
<tr>
<td>R82</td>
<td>Invalid foreign RDFI identification</td>
</tr>
<tr>
<td>R83</td>
<td>Foreign RDFI unable to settle</td>
</tr>
<tr>
<td>R84</td>
<td>Cross-border entry not processed by originating gateway operator</td>
</tr>
</tbody>
</table>
A.10 ACH NoC Change Codes

Table A-10 is a list of ACH NOC Change Codes. These codes are included in the daily NOC report made available to you via sFTP.

**TABLE A-10** eCheck NOC Change Codes

<table>
<thead>
<tr>
<th>ACH NOC Change Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C01</td>
<td>Incorrect account number</td>
</tr>
<tr>
<td>C02</td>
<td>Incorrect routing/transit number</td>
</tr>
<tr>
<td>C03</td>
<td>Incorrect routing/transit number and incorrect account number</td>
</tr>
<tr>
<td>C04</td>
<td>Incorrect account name</td>
</tr>
<tr>
<td>C05</td>
<td>Incorrect transaction code</td>
</tr>
<tr>
<td>C06</td>
<td>Incorrect account number and transaction code</td>
</tr>
<tr>
<td>C07</td>
<td>Incorrect routing/transit number, account number and transaction code</td>
</tr>
<tr>
<td>C08</td>
<td>Incorrect foreign RDFI identification</td>
</tr>
<tr>
<td>C09</td>
<td>Incorrect individual ID</td>
</tr>
<tr>
<td>C13</td>
<td>Addenda format error</td>
</tr>
<tr>
<td>C61</td>
<td>Misrouted NOC</td>
</tr>
<tr>
<td>C62</td>
<td>Incorrect trace number</td>
</tr>
<tr>
<td>C63</td>
<td>Incorrect company ID</td>
</tr>
<tr>
<td>C64</td>
<td>Incorrect individual ID</td>
</tr>
<tr>
<td>C65</td>
<td>Incorrectly formatted correct data</td>
</tr>
<tr>
<td>C66</td>
<td>Incorrect discretionary data</td>
</tr>
<tr>
<td>C67</td>
<td>Routing/transit number not from original entry</td>
</tr>
<tr>
<td>C68</td>
<td>Account number not from original entry</td>
</tr>
<tr>
<td>C69</td>
<td>Incorrect transaction code</td>
</tr>
<tr>
<td>C96</td>
<td>Administrative return dishonor (dollar amount will be zero)</td>
</tr>
<tr>
<td>C99</td>
<td>Converted to MICR draft (check conversion items)</td>
</tr>
</tbody>
</table>
A.11 Canadian eCheck Return Codes

Table A-9 is a list of Canadian eCheck Return Reason Codes. These codes are not returned in the cnpAPI response messages, but are visible in iQ on the eCheck Returns Received report, as well as the Payment Detail screen and the Funding Instruction Detail screen.

NOTE: Return Reason Codes R901 or R908 are eligible for redeposit. Similarly, Code R900-xx are Edit rejects, which you can resubmit with corrected information.

### TABLE A-11 Canadian eCheck Return Reason Codes

<table>
<thead>
<tr>
<th>Canadian eCheck Return Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R900-07</td>
<td>Institution ID Invalid</td>
</tr>
<tr>
<td>R900-08</td>
<td>Account Number Invalid</td>
</tr>
<tr>
<td>R900-16</td>
<td>Institution ID for Return Invalid</td>
</tr>
<tr>
<td>R900-17</td>
<td>Account Number for Return Invalid</td>
</tr>
<tr>
<td>R900-D2</td>
<td>Destination Institute is Not Defined on FIF</td>
</tr>
<tr>
<td>R900-E1</td>
<td>Destination Account Number Invalid</td>
</tr>
<tr>
<td>R900-L2</td>
<td>Institution for Return Not Defined on FIF</td>
</tr>
<tr>
<td>R900-L3</td>
<td>Institution for Return Cross Reference Invalid</td>
</tr>
<tr>
<td>R900-M1</td>
<td>Account for Return Invalid</td>
</tr>
<tr>
<td>R901</td>
<td>NFS (Debit Only)</td>
</tr>
<tr>
<td>R902</td>
<td>Cannot Trace</td>
</tr>
<tr>
<td>R903</td>
<td>Payment Stopped/Recalled</td>
</tr>
<tr>
<td>R904</td>
<td>Post/Stale Dated</td>
</tr>
<tr>
<td>R905</td>
<td>Account Closed</td>
</tr>
<tr>
<td>R907</td>
<td>No Debit Allowed</td>
</tr>
<tr>
<td>R908</td>
<td>Funds Not Cleared</td>
</tr>
<tr>
<td>R909</td>
<td>Currency/Account Mismatch</td>
</tr>
<tr>
<td>R910</td>
<td>Payor/Payee Deceased</td>
</tr>
<tr>
<td>R911</td>
<td>Account Frozen</td>
</tr>
<tr>
<td>R912</td>
<td>Invalid/Incorrect Account Number</td>
</tr>
<tr>
<td>R914</td>
<td>Incorrect Payor/Payee Name</td>
</tr>
</tbody>
</table>
## TABLE A-11  Canadian eCheck Return Reason Codes

<table>
<thead>
<tr>
<th>Canadian eCheck Return Reason Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R915</td>
<td>PAD No Agreement Existed - Business/Personal</td>
</tr>
<tr>
<td>R916</td>
<td>PAD Not According to Agreement - Personal</td>
</tr>
<tr>
<td>R917</td>
<td>PAD Agreement Revoked - Personal</td>
</tr>
<tr>
<td>R918</td>
<td>PAD No Confirmation/Pre-Notification - Personal</td>
</tr>
<tr>
<td>R919</td>
<td>PAD Not According to Agreement - Business</td>
</tr>
<tr>
<td>R920</td>
<td>PAD Agreement Revoked - Business</td>
</tr>
<tr>
<td>R921</td>
<td>PAD No Confirmation/Pre-Notification - Business</td>
</tr>
<tr>
<td>R922</td>
<td>Customer Initiated Return - CREDIT only</td>
</tr>
<tr>
<td>R990</td>
<td>Institution in Default</td>
</tr>
</tbody>
</table>
Credit Card Number Formats

This appendix has two parts. The first provides basic information about card numbers, such as length, prefixes, and validation numbers. The second part provides information about the Luhn Mod-10 algorithm used to validate account numbers.

Credit Card Number Formats:

Table B-1 provides information on number formats for various credit card types.

**NOTE:** The data presented here is for informational proposes only and is subject to change by the Credit Card Associations/Companies. You should verify the information using additional sources prior to using it to create or alter any of your business systems, processes, or procedures.

**TABLE B-1  Card Number Formats**

<table>
<thead>
<tr>
<th>Card Type</th>
<th>Card Number Prefix/Range</th>
<th>Number Length</th>
<th>Card Validation Number Length</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Express</td>
<td>34 and 37</td>
<td>15 digits</td>
<td>4 digits</td>
<td></td>
</tr>
<tr>
<td>Diners Club International</td>
<td>36</td>
<td>14 digits</td>
<td>3 digits</td>
<td>Account Numbers starting with 36 should be submitted as Discover.</td>
</tr>
<tr>
<td>Diners Club (US and Canada)</td>
<td>54 and 55</td>
<td>16 digits</td>
<td>3 digits</td>
<td>These are processed through the MasterCard network and must be submitted as MasterCard.</td>
</tr>
</tbody>
</table>
### Credit Card Number Formats

**Luhn Mod-10 Algorithm for Card Number Validation:**

The Luhn Mod-10 algorithm was invented in 1954 by IBM scientist Hans Peter Luhn and is a relatively simple formula used in numerous applications to validate identification numbers, including credit cards. The algorithm detects all single digit errors in an account number, as well as most transpositions of adjacent numbers.

Use the following method to determine if an account number is Mod-10 compliant:

1. Working from the right, double every other number. If the result of any doubling is a 2-digit number, treat them as individual digits for step 2. For example, 2 * 9 = 18, should be treated as a 1 and an 8.
2. Add all the numbers together, including those you did not double. Remember to treat any 2-digit numbers as individual numbers.
3. If the result of step 2 is a multiple of 10, the account number is Mod-10 compliant.

<table>
<thead>
<tr>
<th>Card Type</th>
<th>Card Number Prefix/Range</th>
<th>Number Length</th>
<th>Card Validation Number Length</th>
<th>Comments</th>
</tr>
</thead>
</table>
| Discover  | 30000000-30599999  
            | 30950000-30959999  
            | 35280000-35899999  
            | 36  
            | 38  
            | 39  
            | 64  
            | 65  
            | 60110000-60110999  
            | 60112000-60114999  
            | 60117400-60117499  
            | 60117700-60117999  
            | 60118600-60119999  
            | 62212600-62292599  
            | 62400000-62699999  
            | 62820000-62889999  
            | 14 digits or 16 digits | 3 digits | Includes ranges for China UnionPay, JCB, and Diners Club International supported by Discover in the US. |
| MasterCard | 51-55  
             | 222100 - 272099 | 16 digits or 19 digits | 3 digits | The 222100 - 272099 BIN range will be active as of October 2016. Merchants must support this BIN range no later than January 1, 2017. |
| Visa      | 4 | 16 digits or 19 digits | 3 digits | |

**Card Type**

- **Discover**
  - 30000000-30599999
  - 30950000-30959999
  - 35280000-35899999
  - 36
  - 38
  - 39
  - 64
  - 65
  - 60110000-60110999
  - 60112000-60114999
  - 60117400-60117499
  - 60117700-60117999
  - 60118600-60119999
  - 62212600-62292599
  - 62400000-62699999
  - 62820000-62889999
- **MasterCard**
  - 51-55
  - 222100 - 272099
- **Visa**
  - 4
**Example:** Mod-10 Algorithm

For the account number 4005550000081019, the computations are shown in the table below.

<table>
<thead>
<tr>
<th></th>
<th>4</th>
<th>0</th>
<th>0</th>
<th>5</th>
<th>5</th>
<th>5</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>8</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>x2</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>10</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>8+</td>
<td>0+</td>
<td>0+</td>
<td>5+</td>
<td>1+0+</td>
<td>5+</td>
<td>0+</td>
<td>0+</td>
<td>0+</td>
<td>0+</td>
<td>8+</td>
<td>2+</td>
<td>0+</td>
<td>2+</td>
<td>9</td>
</tr>
</tbody>
</table>

The result is 40, which is a multiple of 10 and therefore compliant.
The following table provides a list of credit card numbers you can use in our Certification environment to construct your own transactions beyond what is required for certification. These account numbers are extracted from the Certification tests of Chapter 2 and the transaction examples of Chapter 3. Never use these account numbers in the live, production environment.

**IMPORTANT:** Per PCI DSS Requirements and Security Assessment Procedure, Section 6.4.3, "Production data (live PANs) are not used for testing or development."

### TABLE C-1 Test Card Numbers

<table>
<thead>
<tr>
<th>Account Number</th>
<th>Card Type</th>
<th>CVV2/CID</th>
</tr>
</thead>
<tbody>
<tr>
<td>4457010000000009</td>
<td>Visa</td>
<td>349</td>
</tr>
<tr>
<td>4457010100000008</td>
<td>Visa</td>
<td>992</td>
</tr>
<tr>
<td>4457010140000141</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457010200000247</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4100200300011001</td>
<td>Visa</td>
<td>463</td>
</tr>
<tr>
<td>4100200300012009</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4100200300013007</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4100200310000002</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4024720001231239</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457012400000001</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457013200000001</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457119922390123</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000300000007</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000100000009</td>
<td>Visa</td>
<td>N/A</td>
</tr>
</tbody>
</table>
## TABLE C-1  Test Card Numbers

<table>
<thead>
<tr>
<th>Account Number</th>
<th>Card Type</th>
<th>CVV2/CID</th>
</tr>
</thead>
<tbody>
<tr>
<td>4457003100000003</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000400000006</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000200000008</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000800000002</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457000900000001</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4457001000000008</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>40055000000081019</td>
<td>Visa</td>
<td>N/A</td>
</tr>
<tr>
<td>4000000000000002</td>
<td>Visa</td>
<td>555</td>
</tr>
<tr>
<td>2223000148400010</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2223000048400011</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2223280062080010</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222630061560019</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222470061880012</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222400061240016</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222400041240011</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2223520063560019</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2223520043560014</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222420040560011</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2222410040360017</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>2223020040760014</td>
<td>MasterCard</td>
<td>001</td>
</tr>
<tr>
<td>5112000100000003</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112002100000009</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112002200000008</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>511200200000002</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112000300000001</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112000400000000</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112010400000009</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112000600000008</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112010000000003</td>
<td>MasterCard</td>
<td>261</td>
</tr>
<tr>
<td>5112010100000002</td>
<td>MasterCard</td>
<td>251</td>
</tr>
<tr>
<td>Account Number</td>
<td>Card Type</td>
<td>CVV2/CID</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
<td>----------</td>
</tr>
<tr>
<td>5112010140000004</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5154605300000121</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5167001020236549</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5500000254444445</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5592106621450897</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5590409551104142</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5587755665222179</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5445840176552850</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5390016478904678</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112010201000109</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112010202000108</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5194560012341234</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5435101234510196</td>
<td>MasterCard</td>
<td>987</td>
</tr>
<tr>
<td>5407102010000018</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>5112000900000005</td>
<td>MasterCard</td>
<td>N/A</td>
</tr>
<tr>
<td>6011010000000003</td>
<td>Discover</td>
<td>758</td>
</tr>
<tr>
<td>6011010100000002</td>
<td>Discover</td>
<td>184</td>
</tr>
<tr>
<td>6011010140000004</td>
<td>Discover</td>
<td>N/A</td>
</tr>
<tr>
<td>375000026600004</td>
<td>American Express</td>
<td>N/A</td>
</tr>
<tr>
<td>3750010000000005</td>
<td>American Express</td>
<td>N/A</td>
</tr>
<tr>
<td>375001010000003</td>
<td>American Express</td>
<td>0421</td>
</tr>
<tr>
<td>375001014000009</td>
<td>American Express</td>
<td>N/A</td>
</tr>
<tr>
<td>341234567890127</td>
<td>American Express</td>
<td>N/A</td>
</tr>
</tbody>
</table>
PayFac Dynamic Payout

This appendix discusses the Worldpay Instruction-Based Dynamic Payout option provided for use by Payment Facilitators.

Topics covered in this document include:

- Advantages of Using Dynamic Payout
- Overview of Dynamic Payout
- Example of Funding Instructions
- Funding Instruction Certification Testing
- SSR Reports
- Tax ID Validation Process

NOTE: Please also refer to the PayFac Dynamic Payout FAQ document, which contains answers to numerous topics including payout timing, split platform processing, report availability, and general process items.
D.1 Advantages of Using Dynamic Payout

Dynamic Payout is a solution platform that controls the distribution of funds using flexible, customized instructions defined by the Payment Facilitator. The solution provides a closed-loop transaction life cycle from payment to payout. With one connection for payments and payouts, our solution reduces a Payment Facilitator’s dependency on other vendors, minimizing cost associated with PCI and reducing scope.

Dynamic Payout is dependent upon the Payment Facilitator submitting instructions each time a payout is required. A Payment Facilitator can fund merchants on a fixed schedule, such as daily, weekly or monthly, or irregular schedule. Payment Facilitators may even choose to delay funding based on contractual or risk related issues.

The Dynamic Payout funding solution has the following capabilities and benefits:

Capabilities:

• Execute payouts for all card brands, including American Express, as early as the next day.
• Determine when to payout.
• Calculate the fees to charge sub-merchants for rendering service. You can use a complex formula or a tiered billing structure.
• Charge fees at a transaction level, not just MID level.
• Maintain a reserve on your sub-merchants.
• Fund a sub-merchant that needs funds split across multiple bank accounts. You can fund multiple bank accounts per sub-merchant.
D.2 Overview of Dynamic Payout

Dynamic Payout is a method of distributing funds to your sub-merchants, physical check (writing) services, other third party vendors (in the payment flow), and yourself via the ACH network. With Dynamic Payout, you board sub-merchants and submit transactions normally. Funds from settled transactions accumulate in a FBO (For Benefit Of) account. You distribute the funds to your sub-merchants by submitting a Batch file containing Funds Transfer Instructions (see Example of Funding Instructions), or Online by submitting individual instruction. You submit these instructions based upon your payout agreements with your sub-merchants (daily, weekly, monthly, etc.). Worldpay processes the instructions and moves the funds from the holding account to the sub-merchants. Funding can take place as early as the next day after settlement or on the same day if you utilize Same Day Funding.

FIGURE D-1 Overview of Instruction-Based Funding

D.2.1 Timing of Transactions, Reports, and Money Movement

The diagram on the next page illustrates the typical timing and flow of transactions, reports, and money movement. In the diagram, the process begins with the delivery of transactions on Monday morning from the Sub-merchants to the Payment Facilitator and completes on Wednesday morning with the final money movement.

NOTE: The diagram below does not reflect the Same Day Funding or FastAccess Funding™ options.
D.2.2 Money Movement and Accounts

Figure D-3 below illustrates the various possible accounts, as well as the funding instructions associated with moving the funds between the accounts.

**NOTE:** The money movement into the PayFac Settlement account from card and eCheck transactions is the Net Settled Sales (i.e., Deposits - Refunds). The funds debited from the PayFac Operating account is the total of Interchange + Chargebacks + Assessments + Worldpay Fees for Payment Facilitators processing on the eComm platform or just Worldpay fees, if processing on the Worldpay Core platform.

**IMPORTANT:** Worldpay performs a check on the net (deposits - refunds) money movement between accounts for each Batch of funding instructions. If there are insufficient funds in any account impacted by the funding instructions, Worldpay rejects the entire Batch without processing any instructions. The returned error message provides information about which account lacks the necessary funds.
The cnpAPI file shown in Example of Funding Instructions on page 855 provides examples of the transactions used for each type of money movement. The following transaction type are available for your use:

- **Funding Instruction PayFac Credit (FIPC)** - used to move funds from the PayFac Settlement account to the PayFac Operating account.
- **Funding Instruction PayFac Debit (FIPD)** - used to move funds from the PayFac Operating account to the PayFac Settlement account.
- **Funding Instruction Reserve Credit (FIRC)** - used to move funds from the PayFac Settlement account to the PayFac Reserve account.
- **Funding Instruction Reserve Debit (FIRD)** - used to move funds from the PayFac Reserve account to the PayFac Settlement account.
- **Funding Instruction Sub-merchant Credit (FISC)** - used to move funds from the PayFac Settlement account to the sub-merchant Operating account.
• **Funding Instruction Sub-merchant Debit (FISD)** - used to move funds from the sub-merchant Operating account to the PayFac Settlement account.

• **Funding Instruction Vendor Credit (FIVC)** - used to move funds from the PayFac Settlement account to the Vendor account.

• **Funding Instruction Vendor Debit (FIVD)** - used to move funds from the Vendor account to the PayFac Settlement account.

• **Funding Instruction Physical Check Credit (FICC)** - used to move funds from the PayFac Settlement account to the Physical Check account.

• **Funding Instruction Physical Check Debit (FICD)** - used to move funds from the Physical Check account to the PayFac Settlement.

• **Funding Instruction Void** - used to void a submitted, but unsettled funding instruction. (See Funding Instruction Void Transactions on page 858.)

• **FastAccess Funding™** - used to transfer funds to certain eligible MasterCard or Visa debit cards. Transfer of funds takes place within 30 minutes. (See FastAccess Funding on page 852.)

### D.2.2.1 Same Day Funding

Starting with V11.1, Worldpay provides an additional funding instruction submission window that adds the capability of same day funding. This means you can submit a funding instruction files (Batch or Online) before 11:00 AM ET and the funds move the same day. Submissions must also meet the following conditions for same day processing.

- All transactions, Batch or Online must be for less than $25,000 (single transaction limit).

**NOTE:** If you include a transaction over $25,000, we mark the transactions as same day, but processing takes place next day. Because you set the sameDayFunding flag to true, we assess fees for same day funding. To avoid excess fees, do not attempt to use same day funding for these transactions.

- We only process same day funding batch submissions on non-holiday weekdays (i.e., no weekends or holidays).
- Batch or Online instructions submitted outside the allowed window are processed as next day funding instructions.
- You must set the sameDayFunding attribute (of the batchRequest or litleOnlineRequest) to true.

If you miss the submission window with a batch or online file marked for same day funding, we process it as a normal, next day funded batch and apply only the normal next day funding fee.

### D.2.2.2 FastAccess Funding

The FastAccess Funding feature allows you to fund a sub-merchant within 30 minutes of submitting the fastAccessFunding transaction by pushing the funds to certain MasterCard or Visa cards held by your sub-merchant.

The example below uses the token structure to designate the card to which you are pushing the funds. This transaction type also allows the use of the card structure or the paypage structure in place of the token structure.
Example:  FastAccess Funding Request (using a token)

```xml
<cnponlineRequest version="12.0" xmlns="http://www.vantivcnp.com/schema"
  merchantId="100">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <fastAccessFunding id="A123456" reportGroup="FastPayment">
    <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
    <submerchantName>Some Merchant Inc.</submerchantName>
    <fundsTransferId>123e4567e89b12d3</fundsTransferId>
    <amount>3000</amount>
    <token>
      <cnpToken>1111000101039449</cnpToken>
      <expDate>1112</expDate>
      <cardValidationNum>987</cardValidationNum>
      <type>VI</type>
    </token>
  </fastAccessFunding>
</cnponlineRequest>
```

Example:  FastAccess Funding Response

```xml
<litleOnlineResponse version="11.3" xmlns="http://www.litle.com/schema"
  response="0" message="Valid Format">
  <fastAccessFundingResponse id="A123456" reportGroup="FastPayment">
    <litleTxnId>82823972759879805</litleTxnId>
    <fundsTransferId>123e4567e89b12d3</fundsTransferId>
    <response>000</response>
    <responseTime>2017-09-09T20:28:32</responseTime>
    <postDate>2017-09-09</postDate>
    <message>Approved</message>
  </fastAccessFundingResponse>
</litleOnlineResponse>
```

D.2.2.3  Account Balance Verifications

When the Batch or Online transaction arrives, Worldpay performs a front-end check on each Funding Instruction within a Batch, as well as Online instructions. We verify sufficient account balances to cover the money movement from each account. If system detects insufficient funds in any account impacted by a funding instruction, Worldpay rejects the individual Online instruction or the entire Batch. The returned error message provides information about the account lacking funds. We also perform a back-end balance check on Batch files ready for delivery upstream. In this case, if the money movement of any Batch results in an insufficient balance in any account, we reject all Batches.
NOTE: To avoid possible account balance verification issues, Worldpay recommends you submit debit transactions first, in a separate Session file from the credit transactions.

Also, keep in mind that the system may not handle transactions within a Batch sequentially. This is likewise true for Batches within a Session, but we do handle Session files in the order sent. To guarantee sequential handling of Batch files, you must submit the Batch individually in Session files.

For example, you submit a Batch of funding instructions that include a number of reserveCredit and reserveDebit transactions, such that the net funds movement (credits - debits) results in $200,000 being moved from the PayFac Reserve Account to the PayFac Settlement Account. If the current balance in the Reserve account is less than $200,000, The front-end checks detect this situation and reject the entire Batch with a reject message similar to:

```xml
<cnpResponse version="12.2" xmlns="http://www.vantivcnp.com/schema" id="691" response="1" message="Over Balance (Cnp ID: 819812345678357001, session sequence: 2, unique ID: null) not processed - The specified Funding Instructions would result in a negative balance in your Reserve Account. Current settlement balance:20000000, current reserve balance:15000000, current physical check balance:990000" cnpSessionId="810123456789357102"></cnpResponse>
```

NOTE: The balance amounts shown in the example above are in cents. For example, the reserve account balance of 15000000 is $150000.00.
D.3 Example of Funding Instructions

To use Instruction-Based Funding, you must code to cnpAPI V9.0 or above for Batch, or V11.3 or above for Online. The example below shows a Batch containing the various funding instruction you can use. Do not mix other transaction types in a Batch containing funding instructions.

Example: cnpAPI Funding Instructions

```xml
<litleRequest version="11.3" xmlns="http://www.litle.com/schema"
numBatchRequests="1">
  <authentication>
    <user>username</user>
    <password>password</password>
  </authentication>
  <batchRequest merchantId="01601" numPayFacCredit="1" payFacCreditAmount="1000"
numPayFacDebit="1" payFacDebitAmount="2000" numSubmerchantCredit="1"
submerchantCreditAmount="3000" numSubmerchantDebit="1"
submerchantDebitAmount="4000" numReserveCredit="1" reserveCreditAmount="5000"
numReserveDebit="1" reserveDebitAmount="6000" numVendorCredit="1"
vendorCreditAmount="7000" numVendorDebit="1" vendorDebitAmount="8000"
numPhysicalCheckCredit="1" physicalCheckCreditAmount="9000"
numPhysicalCheckDebit="1" physicalCheckDebitAmount="10000"
sameDayFunding="true">
    <!-- Example of PayFac funding themselves. Funds move from the PayFac Settlement Account to the PayFac Operating Account. -->
    <payFacCredit id="abc12345" reportGroup="CollectedFees">
      <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
      <fundsTransferId>123e4567-e89b-12d3-a456-426655440000</fundsTransferId>
      <amount>1000</amount>
    </payFacCredit>
    <!-- Example of PayFac returning money to the settlement account. Funds move from the PayFac Operating Account to the PayFac Settlement Account. -->
    <payFacDebit id="abc12346" reportGroup="MiscRefunds">
      <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
      <fundsTransferId>123e4567-e89b-12d3-a456-426655440001</fundsTransferId>
      <amount>2000</amount>
    </payFacDebit>
    <!-- Example of PayFac funding the Submerchant. Funds move from the PayFac Settlement Account to the Submerchant Account. -->
    <submerchantCredit id="abc12347" reportGroup="SubMerchantPayment">
      <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
      <fundsTransferId>123e4567-e89b-12d3-a456-426655440002</fundsTransferId>
      <amount>3000</amount>
    </submerchantCredit>
  </batchRequest>
</litleRequest>
```
<accType>Checking</accType>
<accNum>123456789012</accNum>
<routingNum>114567895</routingNum>
</accountInfo>
<customIdentifier>Co_Descriptor</customIdentifier>
</submerchantCredit>

<!-- Example of PayFac debiting the Submerchant. Funds move from the Submerchant Account to the PayFac Settlement Account. -->
<submerchantDebit id="abc12348" reportGroup="SubMerchantRefund">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <submerchantName>Some Merchant Inc.</submerchantName>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440003</fundsTransferId>
  <amount>4000</amount>
  <accountInfo>
    <accType>Checking</accType>
    <accNum>123456789012</accNum>
    <routingNum>114567895</routingNum>
  </accountInfo>
  <customIdentifier>Co_Descriptor</customIdentifier>
</submerchantDebit>

<!-- Example of PayFac adding money into reserves. Funds move from the PayFac Settlement Account to the Reserve Account. -->
<reserveCredit id="abc12349" reportGroup="Reserve">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440004</fundsTransferId>
  <amount>5000</amount>
</reserveCredit>

<!-- Example of PayFac getting money from Reserves. Funds move from the Reserve Account to the PayFac Settlement Account. -->
<reserveDebit id="abc12350" reportGroup="SubMerchantRefund">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440005</fundsTransferId>
  <amount>6000</amount>
</reserveDebit>

<!-- Example of PayFac funding the vendor. Funds move from the PayFac Settlement Account to the Vendor Account. -->
<vendorCredit id="abc12351" reportGroup="vendorPayment">
  <fundingSubmerchantId>SomeVendor</fundingSubmerchantId>
  <vendorName>Some Vendor Inc.</vendorName>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440006</fundsTransferId>
  <amount>7000</amount>
  <accountInfo>
    <accType>Checking</accType>
    <accNum>123456789012</accNum>
    <routingNum>114567895</routingNum>
  </accountInfo>
<!-- Example of PayFac debiting the vendor account. Funds move from the Vendor Account to the Payfac Settlement Account. -->

```
/vendorDebit id="abc12352" reportGroup="vendorReturn">
  <fundingSubmerchantId>SomeVendor</fundingSubmerchantId>
  <vendorName>Some Vendor Inc.</vendorName>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440007</fundsTransferId>
  <amount>8000</amount>
  <accountInfo>
    <accType>Checking</accType>
    <accNum>123456789014</accNum>
    <routingNum>114567895</routingNum>
  </accountInfo>
</vendorDebit>
```

<!-- Example of PayFac funding the Physical Check Account. Funds move from the PayFac Settlement Account to the Physical Check Account -->

```
/physicalCheckCredit id="abc12353" reportGroup="physicalCheck">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440008</fundsTransferId>
  <amount>9000</amount>
<//physicalCheckCredit>
```

<!-- Example of PayFac debiting the Physical Check account. Funds move from the Physical Check Account to the PayFac Settlement Account-->

```
/physicalCheckDebit id="abc12354" reportGroup="physicalCheckDebit">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440009</fundsTransferId>
  <amount>10000</amount>
<//physicalCheckDebit>
```

Online Funding Instructions have the same structure as Batch transactions, except the parent element is litleOnlineRequest as shown in the example below.

**Example: Online Sub-merchant Credit Request**

```
<cnpOnlineRequest version="12.0" xmlns="http://www.vantivcnp.com/schema"
  merchantId="100" sameDayFunding="true">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
</cnpOnlineRequest>
```
<!-- Example of PayFac funding the Submerchant. Funds move from the PayFac Settlement Account to the Submerchant Account. -->

<submerchantCredit id="A123456" reportGroup="SubMerchantPayment">
  <fundingSubmerchantId>SomeSubMerchant</fundingSubmerchantId>
  <submerchantName>Some Merchant Inc.</submerchantName>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440002</fundsTransferId>
  <amount>3000</amount>
  <accountInfo>
    <accType>Checking</accType>
    <accNum>123456789012</accNum>
    <routingNum>114567895</routingNum>
  </accountInfo>
  <customIdentifier>Co_Descriptor</customIdentifier>
</submerchantCredit>
</cnpOnlineRequest>

All Funding Instruction responses, except the Void response, have the identical structure other than the parent element. The example shown below, for a payfacCreditResponse illustrates all response structures.

**Example: Funding Instruction Response**

<payFacCreditResponse id="asdf12345" reportGroup="CollectedFees">
  <litleTnxId>82823972759879805</litleTnxId>
  <fundsTransferId>123e4567-e89b-12d3-a456-426655440000</fundsTransferId>
  <response>000</response>
  <message>Approved</message>
</payFacCreditResponse>

**D.3.1 Funding Instruction Void Transactions**

You can use the Funding Instruction Void transaction type to void/remove a designated instruction from a submitted batch of instructions provided the following conditions are met:

- You must be coded to XML V10.1 or above.
- Submit the Funding Instruction Void transaction prior to your cutoff time.
- You must submit the funding instruction void the same day as the funding instruction you wish to void, or in the case of same day funding, before the intra-day cutoff times. This rule also applies to weekends.

**NOTE:** When you issue a Funding Instruction Void, allow 30 minutes for the system to add the amount of the void back to your Settlement Account.
**Example:** Online Funding Instruction Void Request

```
<litleOnlineRequest version="11.3" xmlns="http://www.litle.com/schema"
merchantId="100" sameDayFunding="false">
  <authentication>
    <user>User Name</user>
    <password>Password</password>
  </authentication>
  <!-- Example of Funding Instruction Void. -->
  <fundingInstructionVoid id="A123456" reportGroup="void" id="1">
    <litleTxnId>82828656962104521</litleTxnId>
  </fundingInstructionVoid>
</litleOnlineRequest>
```

**Example:** Online Funding Instruction Void Response

```
<litleOnlineResponse version="11.3" xmlns="http://www.litle.com/schema"
response="0" message="Valid Format">
  <fundingInstructionVoidResponse id="1" reportGroup="void">
    <litleTxnId>82828656962109465</litleTxnId>
    <response>000</response>
    <responseTime>2017-09-02T15:51:54</responseTime>
    <message>Approved</message>
  </fundingInstructionVoidResponse>
</litleOnlineResponse>
```

### D.4 Funding Instruction Certification Testing

In order to validate your cnpAPI structure for Instruction-Based transaction types, submit two Batches, one containing credit transaction and one containing debit transactions, using the data supplied in Table D-1.

**NOTE:** All test transactions result in approved responses. They are intended only to allow you to verify that your XML messages will pass validation.

Although the tests separate debit and credit transactions, there are no system requirements to do so when constructing your Batches.

For required field without supplied data, submit any value valid for the element.
### TABLE D-1  Funding Instruction Test Data

<table>
<thead>
<tr>
<th>Transaction Type</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td><strong>First Batch Containing Credit Transactions:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>payFacCredit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>submerchantCredit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reserveCredit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00005</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vendorCredit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00007</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>physicalCheckCredit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00009</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Second Batch Containing Debit Transactions:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>payFacDebit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00002</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>submerchantDebit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00004</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>reserveDebit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00006</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vendorDebit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00008</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TABLE D-1 Funding Instruction Test Data

<table>
<thead>
<tr>
<th>Transaction Type</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>physicalCheckDebit</td>
<td>&lt;fundsTransferId&gt;</td>
<td>00010</td>
</tr>
<tr>
<td></td>
<td>&lt;amount&gt;</td>
<td>10000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### TABLE D-2 FastAccess Funding Instruction Test Data

<table>
<thead>
<tr>
<th>Transaction Type</th>
<th>Supplied Data Elements</th>
<th>Key Response Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Element</td>
<td>Value</td>
</tr>
<tr>
<td>fastAccessFunding</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4005554444444403</td>
</tr>
<tr>
<td>fastAccessFunding</td>
<td>&lt;type&gt;</td>
<td>VI</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>4005554444444403</td>
</tr>
<tr>
<td>fastAccessFunding</td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5413330089020011</td>
</tr>
<tr>
<td>fastAccessFunding</td>
<td>&lt;type&gt;</td>
<td>MC</td>
</tr>
<tr>
<td></td>
<td>&lt;number&gt;</td>
<td>5413330089020011</td>
</tr>
</tbody>
</table>
D.5 SSR Reports

You are required to receive the following Scheduled Secure Reports when using the Dynamic Payout feature:

- **Funding Reject Report by ACH Return Date** - contains data about failures to transfers funds to Sub-merchants accounts. The report is produced daily.

- **NoC Report by ACH Return Date** - contains NoC data detailing changes in Sub-merchants accounts discovered during funds transfer operations. The report is produced daily.

- **Account Balance Report** - contains data about balances in various accounts used by this solution. The report is produced daily, each hour between 1:00PM ET and 9:00PM ET.

- **Tax Id Mismatch Report** (same as above) - contains data about Legal Entity Tax Identification Numbers validation failures. The report is produced daily.

- **Funding Instruction Confirmation Report** - provides data about all settled funding instructions from the previous day.

- **Balance Summary Report** - provides a summary of the balance in the PayFac account for the previous day.

In addition to the required reports listed above, you can use the SSR to track transactional data, chargebacks, and to assist in reconciliation operations. Some of these reports are:

- **Net Settled Sales by Transaction Report** - includes all settled and conveyed transactions (deposits and refunds), including echeck transactions. The report can be scheduled based upon either Activity (post) or Settlement (funds transfer) day.

- **Session Report** - includes all transactions for a particular activity post day and allows reconciliation against submitted transactions.

- **Transaction Summary Report** - includes summarized deposits and refunds (both settled and conveyed) submitted by the merchant and broken down by purchase currency, reporting groups, and payment type for a particular activity post day.

- **Activity Report** - includes summarized financial data for transactions (deposits and refunds) based upon activity post date and broken down by Reporting Group and payment type.

- **Settlement Report** - includes summarized financial data for settled transactions (deposits and refunds) based upon settlement (funds transfer) date and broken down by Reporting Group and payment type.

- **Chargeback Financial Report** - includes detailed information about financial impacting chargeback activities for a given activity (post) or fund transfer (settlement) date.

- **Chargeback Status Report** - provides details of all chargeback activities for a designated activity (post day) date or date range in the case of a monthly report. This report is run daily or monthly.

- **Fee Report** - provides a detailed breakdown of all Worldpay and Passthrough (Interchange) fees associated with transactions for a given activity (post) or fund transfer (settlement) date.

- **Visa Fixed Acquirer Network Fee Report** - a PayFac report that provides details of Visa FANF fees assessed (by Legal Entity). Worldpay produces the report on the 8th of each month. Each report contains data for the prior month.

- **MasterCard per Location Fee Report** - a PayFac report that provides details of location fees assessed by Legal Entity and Sub-merchant. Worldpay produces the report on the 15th of each month. Each report contains data for the prior month.
NOTE: For additional information about these and other reports, please refer to the latest version of the Worldpay eComm Scheduled Secure Reports Reference Guide.
D.6 Tax ID Validation Process

The Internal Revenue Service requires that “a payment settlement entity (PSE) must file Form 1099-K, Payment Card and Third Party Network Transactions, for payments made in settlement of reportable payment transactions for each calendar year.”

Worldpay issues 1099-Ks to e-commerce merchants, Payment Facilitators, and sub-merchants who are enabled for Dynamic Payout. Prior to issuing 1099-Ks, it is the responsibility of the Payment Facilitator to provide and validate sub-merchant Tax ID information to ensure that the validation process performed by Worldpay through IRS.gov is successful.

If the Tax ID validation fails, it is indicated in the Legal Entity Background Check Results panel of the PayFac Portal. You can also view the PayFac Tax Id Validation Report, available through the Scheduled Secure Report (SSR) service, to determine if a Legal Entity Tax ID Validation has failed. This report only applies to legal entities that have a sub-merchant funded through our platform, and is produced daily.

You can re-submit the sub-merchant request when certain Tax ID validation errors occur using the Edit button at the top of the View Sub-merchant page of the PayFac Portal to edit and re-submit Tax ID information.
## Index

### Numerics

<table>
<thead>
<tr>
<th>Numerics</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DS response certification testing,</td>
<td>133</td>
</tr>
</tbody>
</table>

### A

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAVS Response Codes</td>
<td>809</td>
</tr>
<tr>
<td>acNum</td>
<td>312</td>
</tr>
<tr>
<td>Account Updater</td>
<td>14</td>
</tr>
<tr>
<td>accountInfo</td>
<td>313</td>
</tr>
<tr>
<td>accountInformation</td>
<td>314</td>
</tr>
<tr>
<td>accountNumber</td>
<td>315</td>
</tr>
<tr>
<td>accountNumberLength</td>
<td>316</td>
</tr>
<tr>
<td>accountUpdate</td>
<td>317</td>
</tr>
<tr>
<td>accountUpdateFileRequestData</td>
<td>318</td>
</tr>
<tr>
<td>accountUpdater</td>
<td>319</td>
</tr>
<tr>
<td>accountUpdateResponse</td>
<td>323</td>
</tr>
<tr>
<td>accType</td>
<td>324</td>
</tr>
<tr>
<td>actionReason</td>
<td>325</td>
</tr>
<tr>
<td>activate</td>
<td>326</td>
</tr>
<tr>
<td>activateResponse</td>
<td>327</td>
</tr>
<tr>
<td>activateReversal</td>
<td>328</td>
</tr>
<tr>
<td>activateReversalResponse</td>
<td>329</td>
</tr>
<tr>
<td>active</td>
<td>330</td>
</tr>
<tr>
<td>addOnCode</td>
<td>331</td>
</tr>
<tr>
<td>address response certification testing</td>
<td>127</td>
</tr>
<tr>
<td>addressIndicator</td>
<td>332</td>
</tr>
<tr>
<td>addressLine1</td>
<td>333</td>
</tr>
<tr>
<td>Advanced AVS Response Codes</td>
<td>809</td>
</tr>
<tr>
<td>advancedAVSResult</td>
<td>334</td>
</tr>
<tr>
<td>advancedFraudChecks</td>
<td>335</td>
</tr>
<tr>
<td>advancedFraudResults</td>
<td>336</td>
</tr>
<tr>
<td>affiliate</td>
<td>337</td>
</tr>
<tr>
<td>affluence</td>
<td>338</td>
</tr>
<tr>
<td>Affluence Indicator Feature</td>
<td>25</td>
</tr>
<tr>
<td>allowPartialAuth</td>
<td>339</td>
</tr>
<tr>
<td>American Express</td>
<td></td>
</tr>
<tr>
<td>Authorization lifespan</td>
<td>192</td>
</tr>
<tr>
<td>amexAggregatorData</td>
<td>340</td>
</tr>
<tr>
<td>amount</td>
<td>341</td>
</tr>
<tr>
<td>androidpayResponse</td>
<td>342</td>
</tr>
<tr>
<td>applepay</td>
<td>343</td>
</tr>
<tr>
<td>applepayResponse</td>
<td>344</td>
</tr>
<tr>
<td>applicationData</td>
<td>345</td>
</tr>
<tr>
<td>applicationExpirationDate</td>
<td>346</td>
</tr>
</tbody>
</table>

### Basic Fraud Tools

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Express</td>
<td></td>
</tr>
<tr>
<td>Authorization lifespan</td>
<td>192</td>
</tr>
<tr>
<td>avsResult</td>
<td>364</td>
</tr>
<tr>
<td>availableBalance</td>
<td>363</td>
</tr>
<tr>
<td>AVS Filtering</td>
<td>31</td>
</tr>
<tr>
<td>authAmount</td>
<td>348</td>
</tr>
<tr>
<td>authCode</td>
<td>349</td>
</tr>
<tr>
<td>authDate</td>
<td>351</td>
</tr>
<tr>
<td>authenticatedByMerchant</td>
<td>352</td>
</tr>
<tr>
<td>authentication</td>
<td>353</td>
</tr>
<tr>
<td>authenticationResult</td>
<td>354</td>
</tr>
<tr>
<td>authenticationTransactionId</td>
<td>355</td>
</tr>
<tr>
<td>authenticationValue</td>
<td>356</td>
</tr>
<tr>
<td>authorization</td>
<td>358</td>
</tr>
<tr>
<td>Authorization Reversal transactions certification testing</td>
<td>104</td>
</tr>
<tr>
<td>authorizationResponse</td>
<td>359</td>
</tr>
<tr>
<td>authorizationSourcePlatform</td>
<td>360</td>
</tr>
<tr>
<td>authReversal</td>
<td>361</td>
</tr>
<tr>
<td>authReversalResponse</td>
<td>362</td>
</tr>
<tr>
<td>balanceInquiry</td>
<td>365</td>
</tr>
<tr>
<td>balanceInquiryResponse</td>
<td>366</td>
</tr>
<tr>
<td>Basic Fraud Tools</td>
<td></td>
</tr>
<tr>
<td>AVS Filtering</td>
<td>31</td>
</tr>
<tr>
<td>Prepaid Card Filtering</td>
<td>29</td>
</tr>
<tr>
<td>Prior Chargeback Filtering</td>
<td>30</td>
</tr>
<tr>
<td>Prior Fraud Filtering</td>
<td>31</td>
</tr>
<tr>
<td>Security Code No-match Filtering</td>
<td>30</td>
</tr>
<tr>
<td>Basic Fraud tools</td>
<td></td>
</tr>
<tr>
<td>International Card Filtering</td>
<td>29</td>
</tr>
<tr>
<td>batchRequest</td>
<td>367</td>
</tr>
<tr>
<td>batchResponse</td>
<td>375</td>
</tr>
</tbody>
</table>
beginningBalance, 376
billingDate, 377
billMeLaterRequest, 378
billMeLaterResponseData, 380
billToAddress, 381
bin, 383
bmlMerchantId, 384
bmlProductType, 385
bypassVelocityCheck, 386

C

campaign, 387
cancelSubscription, 388
cancelSubscriptionResponse, 388, 389
capability, 390
capture, 391
captureGivenAuth, 393
captureGivenAuthResponse, 394
captureResponse, 395
card, 396
cardAcceptorTaxId, 398
Cardholder Type Indicator Feature, 26
cardholderAuthentication, 399
cardholderId, 400
cardholderName, 401
cardOrToken, 402
cardProductType, 403
cardSuffix, 404
cardValidationNum, 405
cardValidationResult, 406
cashBackAmount, 407
catLevel, 408
cccdPaymentInformation, 409
certification testing
  3DS responses, 133
  address responses, 127
  Authorization Reversal transactions, 104
  AVS response codes, 126
  response codes and messages, 129
  volume testing, 183
chargeback, 410
checkNum, 411
checkoutId, 412
city, 413
clinicOtherAmount, 414
code, 415
commodityCode, 416
companyName, 417
country, 418
createAddOn, 419
createDiscount, 420
createPlan, 421
createPlanResponse, 422
credit, 423
creditLine, 425
creditLittleTxnId, 426
creditResponse, 427
cryptogram, 428
currencyCode, 429
customAttribute1, 430
customBilling, 431
customerInfo, 434
customerIpAddress, 435
customerReference, 436
customerRegistrationDate, 437
customerType, 438
customerWorkTelephone, 439
customIdentifier, 433

d

data, 440
deactivate, 441
deactivateResponse, 442
deactivateReversal, 443
deactivateReversalResponse, 444
debtRepayment, 445
Decline Notice
eCheck Verification, 46
deleteAddOn, 446
deleteDiscount, 447
deliveryType, 448
dentalAmount, 449
depositReversal, 450
depositReversalResponse, 451
description, 452
descriptor, 453
destinationCountryCode, 454
destinationPostalCode, 455
detailTax, 456
deviceManufacturerIdentifier, 457
deviceReputationScore, 458
deviceReviewStatus, 459
discountAmount, 460
discountCode, 461
Discover
   Authorization lifespan, 192
dob, 462
duplicate Transaction Detection Feature, 8
   dutyAmount, 463
Dynamic Payout
   Account Balance Verification, 853
   Certification Testing, 859
   FastAccess Funding, 852
   Funding Instruction Void, 858
   Funding Instructions, 855
Money Movement and Accounts, 850
   Overview, 849
   Same Day Funding, 852
   Tax Id Validation, 864

E

echeck, 464
eCheck Auto Redeposit Feature, 48
eCheck NoC Update Feature, 47
eCheck Processing Feature, 46
eCheck Validation Feature, 46
eCheck Verification
   Decline Notice, 46
eCheck Verification Feature, 46
eCheckAccountSuffix, 465
echeckCredit, 466
echeckCreditResponse, 467
echeckForToken, 468
echeckOrEcheckToken, 469
echeckPreNoteCredit, 470
echeckPreNoteCreditResponse, 471
echeckPreNoteSale, 472
echeckPreNoteSaleResponse, 473
echeckRedeposit, 474
echeckRedepositResponse, 475
echeckSale, 476
echeckSalesResponse, 477
echeckToken, 478
echeckVerification, 479
echeckVerificationResponse, 480
echeckVoid, 481
echeckVoidResponse, 482
eclIndicator, 483
eemail, 484
eemployerName, 485
eencryptedTrack, 486
   endDate, 487
   endingBalance, 488
   endpoint, 489
   enhancedAuthResponse, 490
   enhancedData, 492
   entryMode, 496
ephemeralPublicKey, 497
   error messages
      response Header, 831
      XML validation, 829
   expDate, 498
   expMonth, 499
   expYear, 500
   extendedCardResponse, 501

F

FastAccess Funding, 852
fastAccessFunding, 502
fastAccessFundingResponse, 503
Features
   Account Updater, 14
   Affluence Indicator, 25
   Cardholder Type Indicator, 26
   Duplicate Transaction Detection, 8
   eCheck Auto Redeposit, 48
   eCheck NoC Update, 47
   eCheck Processing, 46
   eCheck Validation, 46
   eCheck Verification, 46
   Issuer Country Indicator, 25
   Prepaid Indicator, 24
   Recovery, 12
   Recurring Engine, 18
   Recycling Engine, 12
   Report Groups, 10
   Tokenization, 37
   fieldValue, 504
   filtering, 505
   Filtering Rules, 32
   finalPayment, 506
   firstName, 507
   forceCapture, 508
   forceCaptureResponse, 509
   formatId, 510
   Fraud Check transaction, 36
   Fraud Filtering
      Filtering Rules, 32
<table>
<thead>
<tr>
<th><strong>Index</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fraud Toolkit</strong>, 28</td>
</tr>
<tr>
<td><strong>Fraud Velocity Filtering</strong>, 30</td>
</tr>
<tr>
<td><strong>fraudCheck</strong>, 511, 686</td>
</tr>
<tr>
<td><strong>fraudCheckResponse</strong>, 512</td>
</tr>
<tr>
<td><strong>fraudFilterOverride</strong>, 513</td>
</tr>
<tr>
<td><strong>fraudResult</strong>, 514</td>
</tr>
<tr>
<td><strong>Funding Instruction Void</strong>, 858</td>
</tr>
<tr>
<td><strong>Funding Instructions</strong>, 855</td>
</tr>
<tr>
<td><strong>fundingInstructionVoid</strong>, 515</td>
</tr>
<tr>
<td><strong>fundingInstructionVoidResponse</strong>, 516</td>
</tr>
<tr>
<td><strong>fundingSource</strong>, 517</td>
</tr>
<tr>
<td><strong>fundingSubmerchantId</strong>, 518</td>
</tr>
<tr>
<td><strong>fundsTransferId</strong>, 519</td>
</tr>
<tr>
<td><strong>G</strong></td>
</tr>
<tr>
<td><strong>giftCardBin</strong>, 522</td>
</tr>
<tr>
<td><strong>giftCardResponse</strong>, 527</td>
</tr>
<tr>
<td><strong>H</strong></td>
</tr>
<tr>
<td><strong>header</strong>, 530</td>
</tr>
<tr>
<td><strong>healthcareIIAS</strong>, 532</td>
</tr>
<tr>
<td><strong>I</strong></td>
</tr>
<tr>
<td><strong>iban</strong>, 533</td>
</tr>
<tr>
<td><strong>idle time timeout</strong>, 90</td>
</tr>
<tr>
<td><strong>IIASFlag</strong>, 536</td>
</tr>
<tr>
<td><strong>incomeAmount</strong>, 537</td>
</tr>
<tr>
<td><strong>incomeCurrency</strong>, 538</td>
</tr>
<tr>
<td><strong>Instruction-Based Dynamic Payout</strong>, 78</td>
</tr>
<tr>
<td><strong>international</strong>, 540</td>
</tr>
<tr>
<td><strong>International Card Filtering</strong>, 29</td>
</tr>
<tr>
<td><strong>intervalType</strong>, 541</td>
</tr>
<tr>
<td><strong>invoiceReferenceNumber</strong>, 542</td>
</tr>
<tr>
<td><strong>Issuer Country Indicator Feature</strong>, 25</td>
</tr>
<tr>
<td><strong>Issuer Insights</strong>, 24</td>
</tr>
<tr>
<td><strong>issuerCountry</strong>, 543</td>
</tr>
<tr>
<td><strong>itemCategoryCode</strong>, 544</td>
</tr>
<tr>
<td><strong>itemDescription</strong>, 545</td>
</tr>
<tr>
<td><strong>itemDiscountAmount</strong>, 546</td>
</tr>
<tr>
<td><strong>itemSequenceNumber</strong>, 547</td>
</tr>
<tr>
<td><strong>K</strong></td>
</tr>
<tr>
<td><strong>ksn</strong>, 548</td>
</tr>
<tr>
<td><strong>L</strong></td>
</tr>
<tr>
<td><strong>lastName</strong>, 549</td>
</tr>
<tr>
<td><strong>lineItemData</strong>, 550</td>
</tr>
<tr>
<td><strong>lineItemTotal</strong>, 552</td>
</tr>
<tr>
<td><strong>lineItemTotalWithTax</strong>, 553</td>
</tr>
<tr>
<td><strong>littleInternalRecurringRequest</strong>, 554</td>
</tr>
<tr>
<td><strong>littleOnlineRequest</strong>, 555</td>
</tr>
<tr>
<td><strong>littleOnlineResponse</strong>, 556</td>
</tr>
<tr>
<td><strong>littleRequest</strong>, 558</td>
</tr>
<tr>
<td><strong>littleResponse</strong>, 559</td>
</tr>
<tr>
<td><strong>littleSessionId</strong>, 560</td>
</tr>
<tr>
<td><strong>littleToken</strong>, 562</td>
</tr>
<tr>
<td><strong>littleTxnId</strong>, 563</td>
</tr>
<tr>
<td><strong>load</strong>, 564</td>
</tr>
<tr>
<td><strong>loadResponse</strong>, 565</td>
</tr>
<tr>
<td><strong>loadReversal</strong>, 566</td>
</tr>
<tr>
<td><strong>loadReversalResponse</strong>, 567</td>
</tr>
<tr>
<td><strong>M</strong></td>
</tr>
<tr>
<td><strong>mandateProvider</strong>, 568</td>
</tr>
<tr>
<td><strong>mandateReference</strong>, 569</td>
</tr>
<tr>
<td><strong>mandateSignatureDate</strong>, 570</td>
</tr>
<tr>
<td><strong>mandateUrl</strong>, 571</td>
</tr>
<tr>
<td><strong>MasterCard</strong></td>
</tr>
<tr>
<td><strong>Authorization lifespan</strong>, 192</td>
</tr>
<tr>
<td><strong>merchantData</strong>, 573</td>
</tr>
<tr>
<td><strong>merchantGroupingId</strong>, 574</td>
</tr>
<tr>
<td><strong>merchantId</strong>, 575</td>
</tr>
<tr>
<td><strong>message</strong>, 576</td>
</tr>
<tr>
<td><strong>middleInitial</strong>, 577</td>
</tr>
<tr>
<td><strong>mpos</strong>, 578</td>
</tr>
<tr>
<td><strong>N</strong></td>
</tr>
<tr>
<td><strong>name</strong>, 579</td>
</tr>
<tr>
<td><strong>networkField</strong>, 580</td>
</tr>
<tr>
<td><strong>networkResponse</strong>, 582</td>
</tr>
<tr>
<td><strong>networkResponse</strong>, 583</td>
</tr>
<tr>
<td><strong>networkTransactionId</strong>, 584</td>
</tr>
<tr>
<td><strong>newAccountInfo</strong>, 585</td>
</tr>
<tr>
<td><strong>newCardInfo</strong>, 586</td>
</tr>
<tr>
<td><strong>newCardTokenInfo</strong>, 587</td>
</tr>
<tr>
<td><strong>newTokenInfo</strong>, 588</td>
</tr>
<tr>
<td><strong>nextRecycleTime</strong>, 589</td>
</tr>
<tr>
<td><strong>number</strong>, 590</td>
</tr>
<tr>
<td><strong>numberOfPayments</strong>, 591</td>
</tr>
</tbody>
</table>
Index

O

onlinePaymentCryptogram, 592
optional certification tests
  3DS responses, 133
  address responses, 127
  AVS response codes, 126
  response codes and messages, 129
  volume testing, 183
orderDate, 593
orderId, 594
orderSource, 595
origAccountNumber, 598
origActionType, 599
origId, 601
originalAccountInfo, 597
originalCard, 603
originalCardInfo, 604
originalCardTokenInfo, 605
originalNetworkTransactionId, 606
originalToken, 610
originalTokenInfo, 611
originalTransactionAmount, 612
origLitleTxnId, 614
origOrderId, 615

P

password, 616
payerId, 617
payFacCredit, 618
payFacCreditResponse, 619
payFacDebit, 620
payFacDebitResponse, 621
paymentDataType, 622
paypage, 624
paypageRegistrationId, 625
PayPal
  Authorization lifespan, 192
paypal, 626
payPalNotes, 627
payPalOrderComplete, 628
persistent connection timeout, 90
phone, 629
physicalCheckCredit, 630
physicalCheckCreditResponse, 631
physicalCheckDebit, 632
physicalCheckDebitResponse, 633
pin, 634
planCode, 635
pos, 636
postDate, 637
postDay, 638
preapprovalNumber, 639
preferredLanguage, 640
prepaid, 641
Prepaid Card Filtering, 29
Prepaid Indicator Feature, 24
prepaidCardType, 642
Prior Chargeback Filtering, 30
Prior Fraud Filtering, 31
processingInstructions, 643
processingType, 644
productCode, 645
publicKeyHash, 646

Q

quantity, 647
Query Transaction, 76, 280
queryTransaction, 648
queryTransactionResponse, 649
queryTransactionUnavailableResponse, 650

R

Recovery, 12
Recurring Engine, 18
recurringRequest, 651
recurringResponse, 651, 652
recurringTxnId, 653
recycleAdvice, 654
recycleAdviceEnd, 655
recycleBy, 656
recycleEngineActive, 657
recycleId, 658
recycling, 659
Recycling Engine, 12
Recycling Engine and Auth Reversal, 70, 77
recyclingRequest, 661
redirectToken, 662
redirectUrl, 663
refundReversal, 665
refundReversalResponse, 666
registerTokenRequest, 667
registerTokenResponse, 668
reloadable, 669
Report Group Feature, 10
reserveCredit, 670
reserveCreditResponse, 671
reserveDebit, 672
reserveDebitResponse, 673
residenceStatus, 674
response, 675
response codes and messages testing, 129
Response Header error messages, 831
responseCode, 676
responseMessage, 677
responseTime, 678
results_Max10, 679
RFRRequest, 682
RFRResponse, 683
routingNum, 684
RxAmount, 685

S
sale, 686
saleResponse, 687
salesTax, 688
Same Day Funding, 852
secondaryAmount, 689
Security Code No-match Filtering, 30
sellerId, 690
sellerMerchantCategoryCode, 691
sepaDirectDebit, 692
sepaDirectDebitResponse, 693
sequenceType, 695
shipFromPostalCode, 696
shippingAmount, 697
shipToAddress, 698
signature, 699
ssn, 702
startDate, 703
state, 704
Status Query Transaction, 76, 280
stopping auto-recycling, 70, 77
submerchantCredit, 705
submerchantCreditResponse, 706
submerchantDebit, 707
submerchantDebitResponse, 708
submerchantName, 709
subscription, 710
subscriptionId, 712
surchargeAmount, 713

T
Tagging Transactions, 11
taxAmount, 715
taxExempt, 716
taxIncludedInTotal, 717
taxRate, 718
taxType, 719
taxTypeIdentifier, 720
terminaldId, 721
termsAndConditions, 722
testing
Authorization Reversal transactions, 104
optional certification tests
3DS responses, 133
address responses, 127
AVS response codes, 126
response codes and messages, 129
volume testing, 183
threatMetrixSessionId, 723
timeout
persistent connection, 90
Timeout settings, 89
token, 724
Tokenization Feature, 37
tokenMessage, 726
tokenResponse, 727
tokenResponseCode, 728
totalHealthcareAmount, 729
track, 730
track1Status, 731
track2Status, 732
transactionAmount, 733
transactionId, 734
trialIntervalType, 735
trialNumberOfIntervals, 736
triggeredRule, 737
type, 739

U
unitCost, 741
unitOfMeasure, 742
unload, 743
unloadResponse, 744
unloadReversal, 745
unloadReversalResponse, 746
updateAddOn, 747
updateCardValidationNumOnToken, 749
updateCardValidationNumOnTokenResponse, 750
updatedCard, 748
updateDiscount, 751
updatedToken, 759
updatePlan, 752
updatePlanResponse, 753
updateSubscription, 754
updateSubscriptionResponse, 758
url, 760
user, 761

V
Value Added Services
   Fraud Toolkit, 28
      Issuer Insights, 24
vendorCredit, 762
vendorCreditResponse, 763
vendorDebit, 764
vendorDebitResponse, 765
vendorName, 766
verificationCode, 767
verify, 768
version, 769
virtualAccountNumber, 770
virtualAuthenticationKeyData, 771
virtualAuthenticationKeyPresenceIndicator, 772
virtualGiftCard, 773
virtualGiftCardBin, 774
virtualGiftCardResponse, 775
Visa
   Authorization lifespan, 192
      visionAmount, 776
void, 777
voidResponse, 778
volume certification testing, 183

W
wallet, 779
walletSourceType, 780
walletSourceTypeld, 781

X
XML elements
   accNum, 312
accountInfo, 313
accountInformation, 314
accountNumber, 315
accountNumberLength, 316
accountUpdate, 317
accountUpdateFileRequestData, 318
accountUpdater, 319
accountUpdateResponse, 323
accType, 324
actionReason, 325
activate, 326
activateResponse, 327
activateReversal, 328
activateReversalResponse, 329
active, 330
addOnCode, 331
addressIndicator, 332
addressLine1, 333
advancedAVSResult, 334
advancedFraudChecks, 335
advancedFraudResults, 336
affiliate, 337
affluence, 338
allowPartialAuth, 339
amexAggregatorData, 340
amount, 341
androidpayResponse, 342
applepay, 343
applepayResponse, 344
applicationData, 345
applicationExpirationDate, 346
applicationPrimaryAccountNumber, 347
approvedAmount, 348
authAmount, 349
authCode, 350
authDate, 351
authenticatedByMerchant, 352
authentication, 353
authenticationResult, 354
authenticationTransactionId, 355
authenticationValue, 356
authInformation, 357
authorization, 358
authorizationResponse, 359
<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>authorizationSourcePlatform</td>
<td>360</td>
</tr>
<tr>
<td>authReversal</td>
<td>361</td>
</tr>
<tr>
<td>authReversalResponse</td>
<td>362</td>
</tr>
<tr>
<td>availableBalance</td>
<td>363</td>
</tr>
<tr>
<td>avsResult</td>
<td>364</td>
</tr>
<tr>
<td>balanceInquiry</td>
<td>365</td>
</tr>
<tr>
<td>balanceInquiryResponse</td>
<td>366</td>
</tr>
<tr>
<td>batchRequest</td>
<td>367</td>
</tr>
<tr>
<td>batchResponse</td>
<td>375</td>
</tr>
<tr>
<td>beginningBalance</td>
<td>376</td>
</tr>
<tr>
<td>billingDate</td>
<td>377</td>
</tr>
<tr>
<td>billMeLaterRequest</td>
<td>378</td>
</tr>
<tr>
<td>billMeLaterResponseData</td>
<td>380</td>
</tr>
<tr>
<td>billToAddress</td>
<td>381</td>
</tr>
<tr>
<td>bin</td>
<td>383</td>
</tr>
<tr>
<td>bmlMerchantId</td>
<td>384</td>
</tr>
<tr>
<td>bmlProductType</td>
<td>385</td>
</tr>
<tr>
<td>bypassVelocityCheck</td>
<td>386</td>
</tr>
<tr>
<td>campaign</td>
<td>387</td>
</tr>
<tr>
<td>cancelSubscription</td>
<td>388</td>
</tr>
<tr>
<td>cancelSubscriptionResponse</td>
<td>388</td>
</tr>
<tr>
<td>capability</td>
<td>389</td>
</tr>
<tr>
<td>capture</td>
<td>390</td>
</tr>
<tr>
<td>captureGivenAuth</td>
<td>391</td>
</tr>
<tr>
<td>captureGivenAuthResponse</td>
<td>393</td>
</tr>
<tr>
<td>captureResponse</td>
<td>394</td>
</tr>
<tr>
<td>card</td>
<td>395</td>
</tr>
<tr>
<td>cardAcceptorTaxId</td>
<td>396</td>
</tr>
<tr>
<td>cardholderAuthentication</td>
<td>398</td>
</tr>
<tr>
<td>cardholderId</td>
<td>399</td>
</tr>
<tr>
<td>cardholderName</td>
<td>400</td>
</tr>
<tr>
<td>cardOrToken</td>
<td>401</td>
</tr>
<tr>
<td>cardProductType</td>
<td>402</td>
</tr>
<tr>
<td>cardSuffix</td>
<td>403</td>
</tr>
<tr>
<td>cardValidationNum</td>
<td>404</td>
</tr>
<tr>
<td>cardValidationResult</td>
<td>405</td>
</tr>
<tr>
<td>cashBackAmount</td>
<td>406</td>
</tr>
<tr>
<td>catLevel</td>
<td>407</td>
</tr>
<tr>
<td>ccdPaymentInformation</td>
<td>408</td>
</tr>
<tr>
<td>chargeback</td>
<td>409</td>
</tr>
<tr>
<td>checkNum</td>
<td>410</td>
</tr>
<tr>
<td>checkoutId</td>
<td>411</td>
</tr>
<tr>
<td>city</td>
<td>412</td>
</tr>
<tr>
<td>clinicOtherAmount</td>
<td>413</td>
</tr>
<tr>
<td>code</td>
<td>414</td>
</tr>
<tr>
<td>commodityCode</td>
<td>415</td>
</tr>
<tr>
<td>companyName</td>
<td>416</td>
</tr>
<tr>
<td>country</td>
<td>417</td>
</tr>
<tr>
<td>createAddOn</td>
<td>419</td>
</tr>
<tr>
<td>createDiscount</td>
<td>420</td>
</tr>
<tr>
<td>createPlan</td>
<td>421</td>
</tr>
<tr>
<td>createPlanResponse</td>
<td>422</td>
</tr>
<tr>
<td>credit</td>
<td>423</td>
</tr>
<tr>
<td>creditLine</td>
<td>424</td>
</tr>
<tr>
<td>creditLitleTxnId</td>
<td>425</td>
</tr>
<tr>
<td>creditResponse</td>
<td>426</td>
</tr>
<tr>
<td>cryptogram</td>
<td>427</td>
</tr>
<tr>
<td>currencyCode</td>
<td>428</td>
</tr>
<tr>
<td>customAttribute1</td>
<td>429</td>
</tr>
<tr>
<td>customBilling</td>
<td>430</td>
</tr>
<tr>
<td>customerInfo</td>
<td>431</td>
</tr>
<tr>
<td>customerIpAddress</td>
<td>432</td>
</tr>
<tr>
<td>customerReference</td>
<td>433</td>
</tr>
<tr>
<td>customerRegistrationDate</td>
<td>434</td>
</tr>
<tr>
<td>customerType</td>
<td>436</td>
</tr>
<tr>
<td>customerWorkTelephone</td>
<td>437</td>
</tr>
<tr>
<td>customIdentifier</td>
<td>438</td>
</tr>
<tr>
<td>data</td>
<td>439</td>
</tr>
<tr>
<td>deactivate</td>
<td>440</td>
</tr>
<tr>
<td>deactivateResponse</td>
<td>441</td>
</tr>
<tr>
<td>deactivateReversal</td>
<td>442</td>
</tr>
<tr>
<td>deactivateReversalResponse</td>
<td>443</td>
</tr>
<tr>
<td>debtRepayment</td>
<td>444</td>
</tr>
<tr>
<td>deleteAddOn</td>
<td>445</td>
</tr>
<tr>
<td>deleteDiscount</td>
<td>446</td>
</tr>
<tr>
<td>deliveryType</td>
<td>447</td>
</tr>
<tr>
<td>dentalAmount</td>
<td>448</td>
</tr>
<tr>
<td>destinationCountryCode</td>
<td>449</td>
</tr>
<tr>
<td>destinationPostalCode</td>
<td>450</td>
</tr>
<tr>
<td>detailTax</td>
<td>451</td>
</tr>
<tr>
<td>deviceManufacturerIdentifier</td>
<td>452</td>
</tr>
<tr>
<td>deviceReputationScore</td>
<td>453</td>
</tr>
<tr>
<td>deviceReviewStatus</td>
<td>454</td>
</tr>
<tr>
<td>discountAmount</td>
<td>455</td>
</tr>
<tr>
<td>discountCode</td>
<td>456</td>
</tr>
<tr>
<td>dob</td>
<td>457</td>
</tr>
<tr>
<td>dutyAmount</td>
<td>458</td>
</tr>
<tr>
<td>echeck</td>
<td>459</td>
</tr>
<tr>
<td>eCheckAccountSuffix</td>
<td>460</td>
</tr>
<tr>
<td>echeckCredit</td>
<td>461</td>
</tr>
<tr>
<td>echeckCreditResponse</td>
<td>462</td>
</tr>
<tr>
<td>echeckForToken</td>
<td>463</td>
</tr>
</tbody>
</table>
newCardInfo, 586
newCardTokenInfo, 587
newTokenInfo, 588
nextRecycleTime, 589
number, 590
numberOfPayments, 591
onlinePaymentCryptogram, 592
orderDate, 593
orderId, 594
orderSource, 595
origAccountNumber, 598
origActionType, 599
origId, 601
originalAccountInfo, 597
originalCard, 603
originalCardInfo, 604
originalCardTokenInfo, 605
originalNetworkTransactionId, 606
originalToken, 610
originalTokenInfo, 611
originalTransactionAmount, 612
origLitleTxnId, 614
origOrderId, 615
password, 616
payerId, 617
payFacCredit, 618
payFacCreditResponse, 619
payFacDebit, 620
payFacDebitResponse, 621
paymentDataType, 622
paypage, 624
paypageRegistrationId, 625
paypal, 626
payPalNotes, 627
payPalOrderComplete, 628
phone, 629
physicalCheckCredit, 630
physicalCheckCreditResponse, 631
physicalCheckDebit, 632
physicalCheckDebitResponse, 633
pin, 634
planCode, 635
pos, 636
postDate, 637
postDay, 638
preapprovalNumber, 639
preferredLanguage, 640
prepaid, 641
prepaidCardType, 642
processingInstructions, 643
processingType, 644
productCode, 645
publicKeyHash, 646
quantity, 647
queryTransaction, 648
queryTransactionResponse, 649
queryTransactionUnavailableResponse, 650
recurringRequest, 651
recurringResponse, 651, 652
recurringTxnId, 653
recycleAdvice, 654
recycleAdviceEnd, 655
recycleBy, 656
recycleEngineActive, 657
recycleId, 658
recycling, 659
recyclingRequest, 661
redirectToken, 662
redirectUrl, 663
refundReversal, 665
refundReversalResponse, 666
registerTokenRequest, 667
registerTokenResponse, 668
reloadable, 669
reserveCredit, 670
reserveCreditResponse, 671
reserveDebit, 672
reserveDebitResponse, 673
residenceStatus, 674
response, 675
responseCode, 676
responseMessage, 677
responseTime, 678
results_Max10, 679
RFRRequest, 682
RFRResponse, 683
routingNum, 684
RxAmount, 685
sale, 686
saleResponse, 687
salesTax, 688
secondaryAmount, 689
sellerId, 690
sellerMerchantCategoryCode, 691
sepaDirectDebit, 692
sepaDirectDebitResponse, 693