Direct/API Integration

Credit & Debit Card Processing

v 2.5.03

Table of Contents

Introduction

Intended Audience

This document is technical in nature and should be used by your company's developers to integrate your systems into the payment gateway. It assumes that the reader has knowledge and understanding of internet protocols like HTTPs, SSL and XML/SOAP. With that being said, however, this introduction section will also provide valuable, non-technical information to any interested non-developer.

Simplifying the Integration Process

There are many complexities when dealing with card transactions. If you try and tackle them all at once the task of integrating will seem complicated. The best way to do the integration is to follow a simple step by step approach and break the process down into manageable sections, each adding functionality as you go along.

To assist you example code is available in the resource section in most of the common programming languages. Where possible please use these well documented examples as a starting point.

Adhering to good coding practices will also greatly simplify your task.

IMPORTANT – PLEASE READ CAREFULLY

Important Notes

Gateway URLs

In this document, payment gateway specific URLs have “paymentprocessor.net” as the domain (For example, https://gw1. paymentprocessor.net/). When using these URL’s in the integration, “paymentprocessor.net” must be replaced by with the name of the payment gateway provider.

Gateway Messages

The gateway accepts data in the form of SOAP (v1.1) XML messages over HTTPS.

Notation Explained

The message variables are primarily described using a hierarchical table – the hierarchy information is implied by the indentation amount of the first column. You can see the XML schema diagrams and example messages in the appendices. The table has 5 Columns:

1) Tag/Attribute Name – this contains the name of the tag (or the name of the attribute of a tag)
2) Data Type – this gives the valid data type that a tag or attribute can contain
3) Max Length – this gives the maximum length for the contents of a tag or attribute. If a “-“ is in this column, then the tag or attribute has no max length, or it is a special type (like a Boolean for example)
4) Mandatory or Always present – for input messages, this is whether or not the tag or attribute is required for a valid message & for output messages this is whether the tag or attribute will always be present in the message
5) Comment – this gives a brief description of the function of the tag or attribute along with anything else worth noting in relation to that tag or attribute

Rows in orange are tags that do not have any content (i.e. they can have attributes, but they don’t have any content apart from child tags).

Rows in white are either tags that have content, or are attributes (marked so) of the containing tag.

Tag/Attribute Name	Data Type	Max Length	Mandatory or Always Present	Comments
RootTag (no attributes or content, only child tags)				
ChildTag (has no content)				
AnAttribute (attribute)				
AnotherChildTag (has content)				

NOTE: The Mandatory/Always Present fields take into account their scope in the XML hierarchy – if a tag is labelled as Mandatory, then it is mandatory if its parent tag is present. The same applies to a tag's presence in the response message.

This simple table would represent the simple XML message (not including the SOAP envelope or body):

```
<RootTag>
    <ChildTag AnAttribute="SomeValue">
        <AnotherChildTag>SomeValue</AnotherChildTag>
    </ChildTag >
</RootTag>
```

The possible values for the data types are detailed in the table below

Data Type	Description
N	Numeric – only numbers allowed
A	Alpha – any printable character is allowed
B	Boolean – only TRUE or FALSE are allowed
-	Special types – these variables only allow a specific set of values. Details of the allowed values are given in the comments section

CardDetailsTransaction

Introduction

The CardDetailsTransaction message is the mainstay of the gateway. It is the one message that merchants must implement in order to process card payments (along with the ThreeDSecureAuthentication message if they wish to be able to take payments that are validated by the 3D Secure scheme)

Request Message

Below are the details for the request message to initiate a transaction where the card details are submitted.

Tag/Attribute Name	Data Type	Max Length	Mandatory	Comments
PaymentMessage			**Yes**	
MerchantAuthentication			**Yes**	
MerchantID (attribute)	A	15	Yes	The gateway account merchant ID issued (not to be confused with the MMS username)
Password (attribute)	A	15	Yes	The gateway account password
TransactionDetails			**Yes**	
Amount (attribute)	N	15	Yes	The transaction amount in minor currency – e.g. for £10.00, it must be submitted as 1000.
CurrencyCode (attribute)	N	3	Yes	ISO 4217 e.g. GBP: 826
OrderID	-	50	Yes	A merchant side ID for the order – primarily used to for determining duplicate transactions
OrderDescription	A	256	No (N/A)	A description for the order
AuthCode	A	-	No (See comment)	This provides an auth code for the transaction is one was obtained manually
ThreeDSecurePassthroughData			**No**	
EnrolmentStatus (attribute)	A	1	Yes	The status value from the VeRes (CH.enrolled) – can be either Y, N or U
AuthenticationStatus (attribute)	A	1	No	The status value from the PaRes (TX.status) – can be either Y, N, U or A
ECI (attribute)	A	2	No	The 2 digital electronic commerce indicator from the PaRes (TX.eci). Must be present if AuthenticationStatus is either

					Y or A
	AuthenticationValue	N	28	No	The authentication value from the PaRes. For Verified By Visa, this is known as CAVV (Cardholder Authentication Verification Value), for MasterCard SecureCode, it is known as UCAF (Universal Cardholder Authentication Field). It must be present if AuthenticationStatus is either Y or A
	TransactionIdentifier	A	28	No	The transaction identifier (xid) for the transaction
	MessageDetails			**Yes**	
	TransactionType (attribute)	-	-	Yes	Must be either SALE, REFUND or PREAUTH
	ThreeDSecureBrowserDetails			**No**	
	DeviceCategory (attribute)	N	-	No	Determines the category for the customer's browser – 0 for computer grade browser, 1 for a mobile device
	AcceptHeaders	A		No	The headers that the device's browser accepts
	UserAgent	A		No	The user agent string for the device's browser
	TransactionControl			**No**	
	EchoCardType	B	-	No (False)	Instructs the gateway to include the card type of the transaction in the message response
	EchoAVSCheckResult	B	-	No (False)	Instructs the gateway to include the AVS results for the transaction in the message response
	EchoCV2CheckResult	B	-	No (False)	Instructs the gateway to include the CV2 results for the transaction in the message response
	EchoAmountReceived	B	-	No (False)	Instructs the gateway to include the amount that was passed to it in the message response
	DuplicateDelay	N	3	No (60)	Sets the amount of time (in seconds) that any orders to the same gateway account with the same OrderID and CardNumber should be rejected
	AVSOverridePolicy	-	-	No	Sets an override AVS checking

			(As set in MMS)	policy for this transaction. (See Appendix 3 for details)
CV2OverridePolicy	-	-	No (As set in MMS)	Sets an override CV2 checking policy for this transaction. (See Appendix 3 for details)
ThreeDSecureOverridePolicy	B	-	No (As set in MMS)	Instructs the gateway to enable/disable 3D Secure checking for this transaction (where possible)
CardDetails			**Yes**	
CardName	A	100	Yes	The name on the customer's card
CardNumber	N	20	Yes	The customer's card number
ExpiryDate			**Yes**	
Month (attribute)	N	2	Yes	The month of the expiry date in 2 digit numeric format – e.g. for July, must be submitted as 07
Year (attribute)	N	2	Yes	The year of the expiry date in 2 digit numeric format – e.g. for 2007, must be submitted as 07
CV2	N	4	No	The security number (also called CVV or CVV2) printed on the customer's card – usually the last 3 or 4 digits printed on the signature strip
IssueNumber	N	2	No	The issue number printed on the customer's card
CustomerDetails			**No**	
BillingAddress			**No**	
Address1	A	100	No	Customer's billing address line 1
Address2	A	50	No	Customer's billing address line 2
Address3	A	50	No	Customer's billing address line 3
Address4	A	50	No	Customer's billing address line 4
City	A	50	No	Customer's billing address city
State	A	50	No	Customer's billing address state
PostCode	A	50	No	Customer's billing address post code
CountryCode	N	3	No	ISO 3166-1 e.g. United Kingdom: 826
EmailAddress	E	100	No	The email address of the customer – NOTE: anything passed in here is validated as an email address, so anything

					passed in must be a valid email address
	PhoneNumber	A	30	No	The customer’s phone number
	CustomerIPAddress	I	15	No	The IP address of the customer (NOT the IP address of the merchant’s website). This is used to determine the customer’s country of origin. The format is xxx.xxx.xxx.xxx
	DateOfBirth	**D**	**10**	**No**	**The date of birth of the customer. Must be in the format YYYY-MM-DD**
	PrimaryAccountDetails			**No**	
	Name	A	100	No	The name of the primary account holder (used for MCC 6012 accounts only)
	AccountNumber	A	50	No	The account number of the primary account holder (used for MCC 6012 accounts only)
	DateOfBirth	D	10	No	The date of birth of the primary account holder (used for MCC 6012 accounts only)
	AddressDetails			**No**	
	PostCode	A	50	No	The post code of the primary account holder (used for MCC 6012 accounts only)

Response

Below are the details for the response that will be received after sending a CardDetailsTransaction request.

Tag/Attribute Name	Data Type	Max Length	Always Present	Comments
CardDetailsTransactionResponse			**Yes**	
CardDetailsTransactionResult			**Yes**	
AuthorisationAttempted (attribute)	B	-	Yes	This indicates whether the transaction was actually sent to the acquirer for authorisation, or whether it failed before authorisation
StatusCode	N		Yes	This indicates the status of the transaction
Message	A		Yes	This gives a more detailed description of the status of the transaction
ErrorMessages			**No**	
MessageDetail			**Yes**	
Detail (multiple)	A	256	Yes	If there were multiple error messages (e.g. multiple input variable validation errors, then they will be detailed here)
PreviousTransactionResult			**No**	
StatusCode	N		Yes	If the transaction was deemed to be a duplicate transaction, this indicates the status of the previous transaction
Message	A		Yes	If the transaction was deemed to be a duplicate transaction, this gives a more detailed description of the status of the previous transaction
TransactionOutputData			**No**	
CrossReference (attribute)	A	25	Yes	This is the unique cross reference for this transaction. If the card has been determined as requiring 3D Secure authentication this must be used as the merchant reference. If the transaction required 3D Secure authentication, then this must be passed to the ACS as 'MD'. If the transaction was rejected as a duplicate transaction, this

				value will hold the cross reference of the previous transaction
AuthCode	A	15	No	If the transaction was successful, then the auth code is passed out here
AddressNumericCheckResult	-	-	No	If requested in the CardDetailsTransaction request message, this gives the results of the address numeric check – will be PASSED, FAILED, PARTIAL, NOT_CHECKED or UNKNOWN
PostCodeCheckResult	-	-	No	If requested in the CardDetailsTransaction request message, this gives the results of the post code check – will be PASSED, FAILED, PARTIAL, NOT_CHECKED or UNKNOWN
CV2CheckResult	-	-	No	If requested in the CardDetailsTransaction request message, this gives the results of the CV2check – will be PASSED, FAILED, NOT_CHECKED or UNKNOWN
CardTypeData			**No**	
CardType	A	-	Yes	If requested in the CardDetailsTransaction request message, this gives the card type for the transaction. (See Appendix 4 for details)
Issuer	A	100	No	The card issuer (if known)
AmountReceived	N	15	No	If requested in the CardDetailsTransaction request message, this gives the amount that was passed to the gateway via the request message
ThreeDSecureOutputData			**No**	
PaREQ	A	-	Yes	If the card has been determined as requiring 3D Secure authentication, this gives the base64 encoded payment request that must be passed to the ACS for authentication. This must be sent to the ACS as 'PaReq'
ACSURL	A	-	Yes	If the card has been determined as requiring 3D

				Secure authentication, this gives the URL of the ACS server that the PaREQ must be sent to
GatewayEntryPoints			**Yes**	
GatewayEntryPoint (multiple)			**Yes**	
EntryPointURL (attribute)	A	256	Yes	The URL of the active gateway entry point
Metric (attribute)	N	5	Yes	A metric value giving an indication of whether transactions should be sent to this gateway entry point

Things to Note

- If requested, the AmountReceived will always echo the amount passed to the gateway in the CardDetailsTransaction message, regardless of the outcome of the transaction (apart from if the message could not be validated due to content errors)
- If the CV2 is not submitted in the CardDetailsTransaction message, then the CV2CheckResult returned in the CardDetailsTransactionResponse will be deemed as UNKNOWN, rather than FAILED
- If the address or the post code information is not submitted in the CardDetailsTransaction message, then the AddressNumericCheckResult and/or the PostCodeCheckResult returned in the CardDetailsTransactionResponse will be deemed as UNKNOWN rather than FAILED
- If the transaction requires 3D Secure validation, then the CrossReference will be used as the variable “MD” which needs to be posted to the Access Control Server (ACSURL) along with the PaREQ

CrossReferenceTransaction

Introduction

Cross reference transactions are primarily used so that the merchant can run subsequent transactions against previous transactions without having to store the credit card details from the original transaction. These transactions may be subsequent sales (used for recurring billing), full or partial collection of funds (if the initial transaction was a pre-authorisation), or full or partial refunds (if the initial transaction was a sale or a collection)

Request

Below are the details for the request message to initiate a cross reference transaction.

Tag/Attribute Name	Data Type	Max Length	Mandatory (Default)	Comments
PaymentMessage			**Yes**	
MerchantAuthentication			**Yes**	
MerchantID	A	15	Yes	The gateway account merchant ID issued (not to be confused with the MMS username)
Password	A	15	Yes	The gateway account password
TransactionDetails			**Yes**	
Amount (attribute)	N	15	No (False)	The transaction amount in minor currency – e.g. for £10.00, it must be submitted as 1000. Mandatory for all TransactionTypes except VOID
CurrencyCode (attribute)	N	3	No (False)	ISO 4217 e.g. GBP: 826. Mandatory for all TransactionTypes except VOID
OrderID	A	50	Yes	A merchant side ID for the order – primarily used to for determining duplicate transactions. Pulled forward from the previous transaction if not set & NewTransaction is false
OrderDescription	A	256	No (See comment)	A description for the order. Pulled forward from the previous transaction if not set & NewTransaction is false
MessageDetails				
TransactionType (attribute)	-	-	Yes	Must be either COLLECTION, REFUND, PREAUTH, SALE, VOID or RETRY
NewTransaction (attribute)	B	-	No (True)	Instructs the gateway to treat this transaction as a new

				transaction
CrossReference (attribute)	A	25	Yes	The cross reference for the previous transaction
TransactionControl			**No**	
EchoCardType	B	-	No (False)	Instructs the gateway to include the card type of the transaction in the message response
EchoAVSCheckResult	B	-	No (False)	Instructs the gateway to include the AVS results for the transaction in the message response
EchoCV2CheckResult	B	-	No (False)	Instructs the gateway to include the CV2 results for the transaction in the message response
EchoAmountReceived	B	-	No (False)	Instructs the gateway to include the amount that was passed to it in the message response
DuplicateDelay	N	3	No (60)	Sets the amount of time (in seconds) that any orders to the same gateway account with the same OrderID and CardNumber should be rejected
AVSOverridePolicy	-	4	No (As set in MMS)	Sets an override AVS checking policy for this transaction. (See Appendix 3 for details)
CV2OverridePolicy	-	2	No (As set in MMS)	Sets an override CV2 checking policy for this transaction. (See Appendix 3 for details)
ThreeDSecureOverridePolicy	B	-	No (False)	Instructs the gateway to enable/disable 3D Secure checking for this transaction (where possible)
OverrideCardDetails			**No**	
CardName	A	100	No (See comment)	The name on the customer's card. Only submit to override the value for the previous transaction (submit "blank" to not use the value from the previous transaction)
CardNumber	N	20	No (See comment)	The customer's card number. Only submit to override the value for the previous transaction
ExpiryDate			**No**	
Month	N	2	No	The month of the expiry date

			(See comment)	in 2 digit numeric format – e.g. for July, must be submitted as 07. Only submit to override the value for the previous transaction (submit -1 to not use the value from the previous transaction)
Year	N	2	No (See comment)	The year of the expiry date in 2 digit numeric format – e.g. for 2007, must be submitted as 07. Only submit to override the value for the previous transaction (submit -1 to not use the value from the previous transaction)
CV2	N	4	No (See comment)	The security number (also called CVV or CVV2) printed on the customer's card – usually the last 3 or 4 digits printed on the signature strip. Only submit to override the value for the previous transaction (submit -1 to not use the value from the previous transaction)
IssueNumber	N	2	No (See comment)	The issue number printed on the customer's card. Only submit to override the value for the previous transaction (submit -1 to not use the value from the previous transaction)
CustomerDetails			**No**	
BillingAddress			**No**	
Address1	A	100	No (See comment)	Customer's billing address line 1. Only pulled forward from previous transaction if NONE of the address fields have been set
Address2	A	50	No (See comment)	Customer's billing address line 2. Only pulled forward from previous transaction if NONE of the address fields have been set
Address3	A	50	No (See comment)	Customer's billing address line 3. Only pulled forward from previous transaction if NONE of the address fields have been set
Address4	A	50	No (See	Customer's billing address line 4. Only pulled forward

			comment)	from previous transaction if NONE of the address fields have been set
City	A	50	No (See comment)	Customer’s billing address city. Only pulled forward from previous transaction if NONE of the address fields have been set
State	A	50	No (See comment)	Customer’s billing address state. Only pulled forward from previous transaction if NONE of the address fields have been set
PostCode	A	50	No (See comment)	Customer’s billing address post code. Only pulled forward from previous transaction if NONE of the address fields have been set
CountryCode	N	3	No (See comment)	ISO 3166-1 e.g. United Kingdom: 826. Only pulled forward from previous transaction if NONE of the address fields have been set
EmailAddress	E	100	No (See comment)	The email address of the customer – NOTE: anything passed in here is validated as an email address, so anything passed in must be a valid email address. Pulled forward from previous transaction if not set
PhoneNumber	A	30	No (See comment)	The customer’s phone number. Pulled forward from previous transaction if not set
CustomerIPAddress	I	15	No	The IP address of the customer (NOT the IP address of the merchant’s website). This is used to determine the customer’s country of origin. The format is xxx.xxx.xxx.xxx
DateOfBirth	D	10	No	The date of birth of the customer. Must be in the format YYYY-MM-DD
PrimaryAccountDetails			**No**	
Name	A	100	No	The name of the primary account holder (used for MCC 6012 accounts only)
AccountNumber	A	50	No	The account number of the primary account holder (used for MCC 6012 accounts only)
DateOfBirth	D	10	No	The date of birth of the

					primary account holder (used for MCC 6012 accounts only)
	AddressDetails			**No**	
	PostCode	A	50	No	The post code of the primary account holder (used for MCC 6012 accounts only)

Response

Below are the details for the response that will be received after sending a CrossReferenceTransaction request.

Tag/Attribute Name	Data Type	Max Length	Always Present	Comments
CrossReferenceTransactionResponse			**Yes**	
CrossReferenceTransactionResult			**Yes**	
AuthorisationAttempted (attribute)	B	-	Yes	This indicates whether the transaction was actually sent to the acquirer for authorisation, or whether it failed before authorisation
StatusCode	N	-	Yes	This indicates the status of the transaction
Message	A	-	Yes	This gives a more detailed description of the status of the transaction
ErrorMessages			**No**	
MessageDetail			**Yes**	
Detail (multiple)	A	256	Yes	If there were multiple error messages (e.g. multiple input variable validation errors, then they will be detailed here)
PreviousTransactionResult			**No**	
StatusCode	N		Yes	If the transaction was deemed to be a duplicate transaction, this indicates the status of the previous transaction
Message	A		Yes	If the transaction was deemed to be a duplicate transaction, this gives a more detailed description of the status of the previous transaction
TransactionOutputData			**No**	
CrossReference (attribute)	A	25	Yes	This is the unique cross reference for this transaction. If the card has been determined as requiring 3D Secure authentication this must be used as the merchant reference. If the transaction was rejected as a duplicate transaction, this value will hold the cross reference of the previous transaction
AuthCode	A	15	No	If the transaction was

				successful, then the auth code is passed out here
AddressNumericCheckResult	-	-	No	If requested in the CrossReferenceTransaction request message, this gives the results of the address numeric check – will be PASSED, FAILED, PARTIAL, NOT_CHECKED or UNKNOWN
PostCodeCheckResult	-	-	No	If requested in the CrossReferenceTransaction request message, this gives the results of the post code check – will be PASSED, FAILED, PARTIAL, NOT_CHECKED or UNKNOWN
CV2CheckResult	-	-	No	If requested in the CrossReferenceTransaction request message, this gives the results of the CV2check – will be PASSED, FAILED, NOT_CHECKED or UNKNOWN
CardTypeData				
CardType	A	-	Yes	If requested in the CrossReferenceTransaction request message, this gives the card type for the transaction. (See Appendix 4 for details)
Issuer	A	100	No	The card issuer (if known)
AmountReceived	N	15	No	If requested in the CrossReferenceTransaction request message, this gives the amount that was passed to the gateway via the request message
ThreeDSecureOutputData			**No**	
PaREQ	A	-	Yes	If the card has been determined as requiring 3D Secure authentication, this gives the base64 encoded payment request that must be passed to the ACS for authentication. This must be sent to the ACS as ‘PaReq’
ACSURL	A	-	Yes	If the card has been determined as requiring 3D Secure authentication, this gives the URL of the ACS server that the PaREQ must be sent to

GatewayEntryPoints			**Yes**	
GatewayEntryPoint (multiple)			**Yes**	
EntryPointURL (attribute)	A	256	Yes	The URL of the active gateway entry point
Metric (attribute)	N	5	Yes	A metric value giving an indication of whether transactions should be sent to this gateway entry point

Things to Note

- We do not store the CV2 values of any transactions, so they are not available to be pulled forwards from the previous transaction. This means that the unless the CV2 is supplied as part of the OverrideCardDetails in the CrossReferenceTransaction message then the results returned will always be UNKNOWN
- If requested, the AmountReceived will always echo the amount passed to the gateway regardless of the outcome of the transaction (apart from if the message could not be validated due to content errors)
- If the address or the post code information is not submitted in the CrossReferenceTransaction message then the AddressNumericCheckResult and the PostCodeCheckResult will be deemed to be UNKNOWN rather than FAILED
- If this transaction is marked as not a new transaction in the CrossReferenceTransaction message, then the OrderID and OrderDescription will be pulled forward from the previous transaction unless they are present in this message
- If this transaction is marked as a new transaction in the CrossReferenceTransaction message, then the OrderID and OrderDescription will not be pulled forward from the previous transaction.

ThreeDSecureAuthentication

Introduction

The 3D Secure authentication request is used when the initial transaction has been returned as requiring the customer to validate their card details with their card issuer. This validation interrupts the payment process & effectively causes a single transaction to be handled in 2 distinct messages – the first message is the initial CardDetailsTransaction message, which completes with a “3D Secure validation required” result & the second message, which contains the 3D Secure validation response from the customer’s card issuer (collected by the customer themselves). The ThreeDSecureAuthentication is the second of the two messages described above.

Request

Below are the details for the request message to initiate a 3D Secure authentication transaction.

Tag/Attribute Name	Data Type	Max Length	Mandatory	Comments
ThreeDSecureMessage			**Yes**	
MerchantAuthentication			**Yes**	
MerchantID	A	15	Yes	The gateway account merchant ID issued (not to be confused with the MMS username)
Password	A	15	Yes	The gateway account password
ThreeDSecureInputData			**Yes**	
CrossReference (attribute)	A	25	Yes	The cross reference returned by the previous response that included the ThreeDSecureOutputData
PaRES	A	-	Yes	The base64 encoded PaRES string returned by the interaction with the ACS server

Response

Below are the details for the response that will be received after sending a ThreeDSecureAuthentication request.

Tag/Attribute Name	Data Type	Max Length	Always Present	Comments
ThreeDSecureAuthenticationResponse			**Yes**	
ThreeDSecureAuthenticationResult			**Yes**	
AuthorisationAttempted (attribute)	B	-	Yes	This indicates whether the transaction was actually sent to the acquirer for authorisation, or whether it failed before authorisation
StatusCode	N	-	Yes	This indicates the status of the transaction
Message	A	-	Yes	This gives a more detailed description of the status of the transaction
ErrorMessages			**No**	
MessageDetail			**Yes**	
Detail (multiple)	A	256	Yes	If there were multiple error messages (e.g. multiple input variable validation errors, then they will be detailed here)
PreviousTransactionResult			**No**	
StatusCode	N		Yes	If the transaction was deemed to be a duplicate transaction, this indicates the status of the previous transaction
Message	A		Yes	If the transaction was deemed to be a duplicate transaction, this gives a more detailed description of the status of the previous transaction
TransactionOutputData			**No**	
CrossReference (attribute)	A	25	Yes	This is the unique cross reference for this transaction. If the transaction was rejected as a duplicate transaction, this value will hold the cross reference of the previous transaction
AuthCode	A	15	No	If the transaction was successful, then the auth code is passed out here
AddressNumericCheckResult	-	-	No	If requested in the initial CardDetailsTransaction or

				CrossReferenceTransaction request message, this gives the results of the address numeric check – will be PASSED, FAILED, NOT_CHECKED or UNKNOWN
PostCodeCheckResult	-	-	No	If requested in the initial CardDetailsTransaction or CrossReferenceTransaction request message, this gives the results of the post code check – will be PASSED, FAILED, NOT_CHECKED or UNKNOWN
CV2CheckResult	-	-	No	If requested in the initial CardDetailsTransaction or CrossReferenceTransaction request message, this gives the results of the CV2check – will be PASSED, FAILED, NOT_CHECKED or UNKNOWN
ThreeDSecureAuthenticationCheckResult	-	-	No	This gives the results of the 3D Secure authentication check – will be PASSED, FAILED or UNKNOWN
CardTypeData			**No**	
CardType	A	-	Yes	If requested in the initial CardDetailsTransaction or CrossReferenceTransaction request message, this gives the card type for the transaction. (See Appendix 4 for details)
Issuer	A	100	No	The card issuer (if known)
AmountReceived	N	15	No	If requested in the initial CardDetailsTransaction or CrossReferenceTransaction request message, this gives the amount that was passed to the gateway via the request message
GatewayEntryPoints			**Yes**	
GatewayEntryPoint (multiple)			**Yes**	
EntryPointURL (attribute)	A	256	Yes	The URL of the active gateway entry point
Metric (attribute)	N	5	Yes	A metric value giving an indication of whether transactions should be sent to this gateway entry point

Things to Note

- The contents of the variable “MD” used in the 3D Secure validation process should be passed in as the CrossReference of the ThreeDSecureAuthentication message
- The value of the ThreeDSecureAuthentication results will give the results of the 3D Secure authentication – it will be either PASSED, FAILED or UNKNOWN. It is worth noting that in some cases, even if the authentication is UNKNOWN or FAILED, then the transaction can still be processed (albeit without the liability shift that happens with 3D Secure authentication)

GetCardType

Introduction

This message allows the merchant to determine the card type of the card in question.

Request

Below are the details for the request message to initiate a get card type transaction.

Tag/Attribute Name	Data Type	Max Length	Mandatory	Comments
Message			**Yes**	
MerchantAuthentication			**Yes**	
MerchantID	A	15	Yes	The gateway account merchant ID issued (not to be confused with the MMS username)
Password	A	15	Yes	The gateway account password
CardNumber	N	20	Yes	The customer’s card number

Response

Below are the details for the response that will be received after sending a GetCardType request.

Tag/Attribute Name	Data Type	Max Length	Always Present	Comments
GetCardTypeResponse			**Yes**	
GetCardTypeResult			**Yes**	
StatusCode	N	1	Yes	This indicates the status of the transaction
Message	A	-	No	This gives a more detailed description of the status of the transaction
ErrorMessages			**No**	
MessageDetail			**Yes**	
Detail (multiple)	A	256	Yes	If there were multiple error messages (e.g. multiple input variable validation errors, then they will be detailed here)
GetCardTypeOutputData			**No**	
CardTypeData				
CardType	A	-	Yes	Gives the card type (see appendix 5)
Issuer	A	100	No	The card issuer (if known)
LuhnCheckRequired	B	-	Yes	Gives a true or false stating whether the Luhn (mod10) check needs to be run against the card number to validate it
IssueNumberStatus	A	20	Yes	Give the status of the issue number. Will be one of the following: MUST_BE_SUBMITTED, DO_NOT_SUBMIT, SUBMIT_ONLY_IF_ON_CARD, IGNORED_IF_SUBMITTED
GatewayEntryPoints			**Yes**	
GatewayEntryPoint (multiple)			**Yes**	
EntryPointURL (attribute)	A	256	Yes	The URL of the active gateway entry point
Metric (attribute)	N	5	Yes	A metric value giving an indication of whether transactions should be sent to this gateway entry point

GetGatewayEntryPoints

Introduction

This message returns the details of all the gateway entry points

Request

Below are the details for the request message to initiate a get gateway entry points transaction.

Tag/Attribute Name	Data Type	Max Length	Mandatory	Comments
Message			**Yes**	
MerchantAuthentication			**Yes**	
MerchantID	A	15	Yes	The gateway account merchant ID issued (not to be confused with the MMS username)
Password	A	15	Yes	The gateway account password

Response

Below are the details for the response that will be received after sending a GetGatewayEntryPoints request.

Tag/Attribute Name	Data Type	Max Length	Always Present	Comments
GetGatewayEntryPointsResponse			**Yes**	
GetGatewayEntryPointsResult			**Yes**	
StatusCode	N	1	Yes	This indicates the status of the transaction
Message	A	-	No	This gives a more detailed description of the status of the transaction
ErrorMessages			**No**	
MessageDetail			**Yes**	
Detail (multiple)	A	256	Yes	If there were multiple error messages (e.g. multiple input variable validation errors, then they will be detailed here)
GetGatewayEntryPointsOutputData			**No**	
GatewayEntryPoints			**Yes**	
GatewayEntryPoint (multiple)			**Yes**	
EntryPointURL (attribute)	A	256	Yes	The URL of the active gateway entry point
Metric (attribute)	N	5	Yes	A metric value giving an indication of whether transactions should be sent to this gateway entry point

Payment Gateway High Availability

Introduction

The payment gateway has been designed to address the modern business electronic payment and credit card processing needs. Our gateway system is housed in a series of world-class data centres, providing an ultra-high availability system to meet the most mission critical processing needs.

Whilst we provide an ultra-high availability gateway, to utilise this correctly can be an intricate operation. Below are some details you will need to factor in when implementing a system which makes efficient use of our ultra-high availability gateway.

Gateway Entry Points

As you may be aware, the gateway has multiple entry points. That is you can communicate with the gateway through more than one URL to process transactions. The key part of any well designed system is to do it in the most efficient manner. These are primarily two ways of communicating with these entry points to ensure transactions are processed. These methods are explained in more detail below.

- Blind Processing
- Gateway State Awareness

Gateway Entry Point Metric

The gateway entry point metric is a vital piece of information received from the gateway. Using one of the two methods below, will help you use this metric properly.

What Is It?

The Gateway Entry Point Metric is a numerical value which is used within the transaction processing on the merchant's system. This numerical value determines which order to use the gateway entry points specified.

How It Works

The gateway entry point metric is used by the merchants system whilst it is determining which entry point to use first. The lower the entry point metric value, the higher the priority it has over other entry points. A value of "-1" means that the entry point will be skipped, might is useful if there is an outage on one of the entry points, which makes firing transaction at that entry point irrelevant.

Gateway Entry Point Selection Methods

Blind Processing

Description

This method is what you may find in the integration sample pack code. The nature of this method is that the merchant's system is not aware of the 'state' of the gateway entry points (be that up and live or perhaps down with an outage of some sort).

How It Works

The way this method utilises the multiple gateway entry points is by 'blindly' firing all transactions to entry point 1 first. If this entry is up, the transaction will be processed and the response message returned as expected. If that entry point is down it will timeout and then the merchant's system will fire the same transaction to the next entry point. Again, if this entry point is live, the transaction will be processed, if it is not, it will timeout. The merchant's system will keep 'blindly' firing transactions to each of the gateway entry points in succession until the transaction is processed by one of them, or, all entry points have been tried.

Pros

The main advantage of this method is that it is incredibly simple to implement. The integration sample code can be pretty much used as is direct out of the sample files in a live environment.

Cons

If the first entry point is down, then every transaction will be fired at it regardless, meaning that it will then need to time-out this attempt before it attempts the next gateway entry point, so transaction processing will be longer in times of entry point outage, i.e. each transaction will need to independently realise that an entry point is down, but then never pass on this "knowledge" to the next transaction.

Entry Point State Awareness

Description

This method differs from the 'Blind Processing' method greatly. The nature of this method is to not fire transactions at each gateway entry point in order, but to record the entry point of each successful transaction, which then in turn becomes the first entry point to try in the next transaction.

How It Works

When the merchant's system is in a new or "reset" mode, the transactions will use the above "Blind Processing" method. Once this transaction is successful (either transaction authorised or rejected response), the transaction result passed back to the merchant's system contains information as to the gateway entry point which processed the transaction. This entry point should be persistently stored on the merchant's system, usually in a database table record of some sort. When the next transaction takes place, it looks at that database table and uses the latest successful gateway entry point as it first entry point to attempt. Like the "Blind Processing" method, if this transaction fails, it will then failover to the next entry point. Again, it will do this until all the entry points have been tried and failed, or the transaction is processed successfully and once again, the successful entry point used is persistently stored ready for the next transaction.

Pros

The "Entry Point State Awareness" method is more sophisticated than the "Blind Processing" method due to its awareness as to the entry points state. This allows more efficient transaction processing during times of an entry point outage.

Cons

This can be an intricate piece of work to develop and implement, even in its simplest form described in the "How It Works" section. The reason we say this is because you can make it even more sophisticated still, as described in the "Additional Ideas" section below.

Additional Ideas

The “Entry Point State Awareness” method is sophisticated in the sense that it isn’t just blindly firing transactions at each entry point in succession. However, the simple version explained in the “How It Works” section still has its flaws but was deliberately kept minimal to get the idea across in its simplest form.

One weakness of the simple “Entry Point State Awareness” method described above is the potential for “stale data”. What we mean by this is that if a transaction is processed and the successful entry point stored, but the next transaction isn’t processed for an hour for example, the “state” of the last successful entry point could easily be invalid. In the world of I.T and the internet, the “state” of systems can change very quickly. You may have a perfectly up and running entry point one minute, something could happen and bring it down and it’s then not able to process transactions. Based on this “stale data”, it would be worth implementing a timeout threshold on the merchant’s system so that if a transaction begins to be processed, but the previous transaction time is greater than the threshold allows, the your “Entry Point State Awareness” is “reset” and you begin the process again using the “Blind Processing” method. This will ensure not only that you’re using the most successful entry point for each new transaction, but if there is a large enough gap between the previous and the next transaction, that you re-test the each entry point in succession if your “state awareness” becomes stale.

There is one additional piece of development that merchants can implement on their system to add yet another level of sophistication. This utilises the function “GetGatewayEntryPoints” found earlier in this document. What this function does is fire a very basic message at a specified gateway entry point and if the gateway entry point is up, it will return a complete list of all available gateway entry points and a metric value for each which it deems appropriate. This also means that if any of the gateway entry points are down, the entry point metric value of “-1” is returned for that particular entry point and the others have the metric value adjusted accordingly. The intricate part of this is that you will still need to have multiple gateway entry points to fire this basic message at. This is because, if you only have one entry point to fire this message at, and that entry point is down, then you will not yield a response from the gateway. This function will also return any **NEW** gateway entry points added that the merchants system may not yet be aware of which will prove to be very useful at increasing the merchants system ultra-high availability status. In order to implement this properly, you will need to have all available entry points persistently stored, in a database being the most favourable option. Then periodically (as you see appropriate) as part of a **SCHEDULED** server maintenance job of some description firing the “GetGatewayEntryPoints” message to the gateway using the “Gateway Entry Point State Awareness” method mentioned above. When you get a successful response from the gateway, you insert any **NEW** gateway entry points that aren’t yet in the database table, and updating all of their respective entry point metric values. The reason you never do this as part of an actual transaction is to not add any delay times to transactions whilst waiting for the “GetGatewayEntryPoints” message to yield its response from the gateway.

Appendix 1: Gateway Response StatusCodes

Below are the status codes likely to be received when integrating with the gateway.

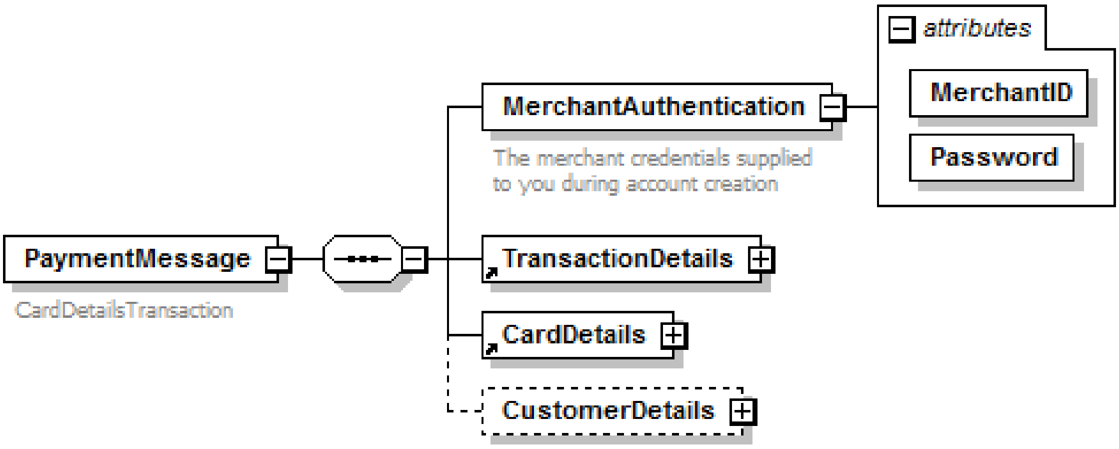
Status Code	Transaction Result	Description
0	Successful	**Transaction Authorised:** The transaction was successful and you will be given an Authorisation Code as part of the message returned by the gateway.
3	Incomplete	**Transaction Awaiting 3D Secure Authentication:** Transaction is now awaiting 3D Secure Authentication from the customer. This status has a 2 hour expiry time set by the card scheme, at which point, the transaction will fail (Issuer Authentication Expired).
4	Referred	**Transaction Referred:** The card issuer has parked the transaction awaiting contact with the customer before proceeding to authorise or decline the transaction.
5	Declined	**Transaction Failed:** The transaction was declined by the card issuer or acquiring bank. In the event of the Address or CV2 verification failure, this will also be noted on the message from the gateway (Example, “Card declined: AVS policy + CV2 policy”). If the message given by the gateway only says “Card declined” with no other information, then no other information was given to us from the card issuer or acquiring bank as to the underlying reason why. The only person who can find out why the transaction was declined is the customer by contacting their bank directly.
20	Duplicate Transaction	The transaction which was processed was a duplicate. If this is the case, then the original transaction information is also passed back from the gateway so you can determine the result of the original transaction. Please refer to your respective integration method documentation form more information.
30	Failed (Error(s) Occurred)	**Transaction Failed:** This is usually an indicator that the integration to the gateway is incomplete and/or not working correctly. There will also be additional error information feedback from the gateway for merchants to determine what the error is specifically. Please refer to your respective integration methods documentation for more information.

Appendix 2: Message Schema Diagrams

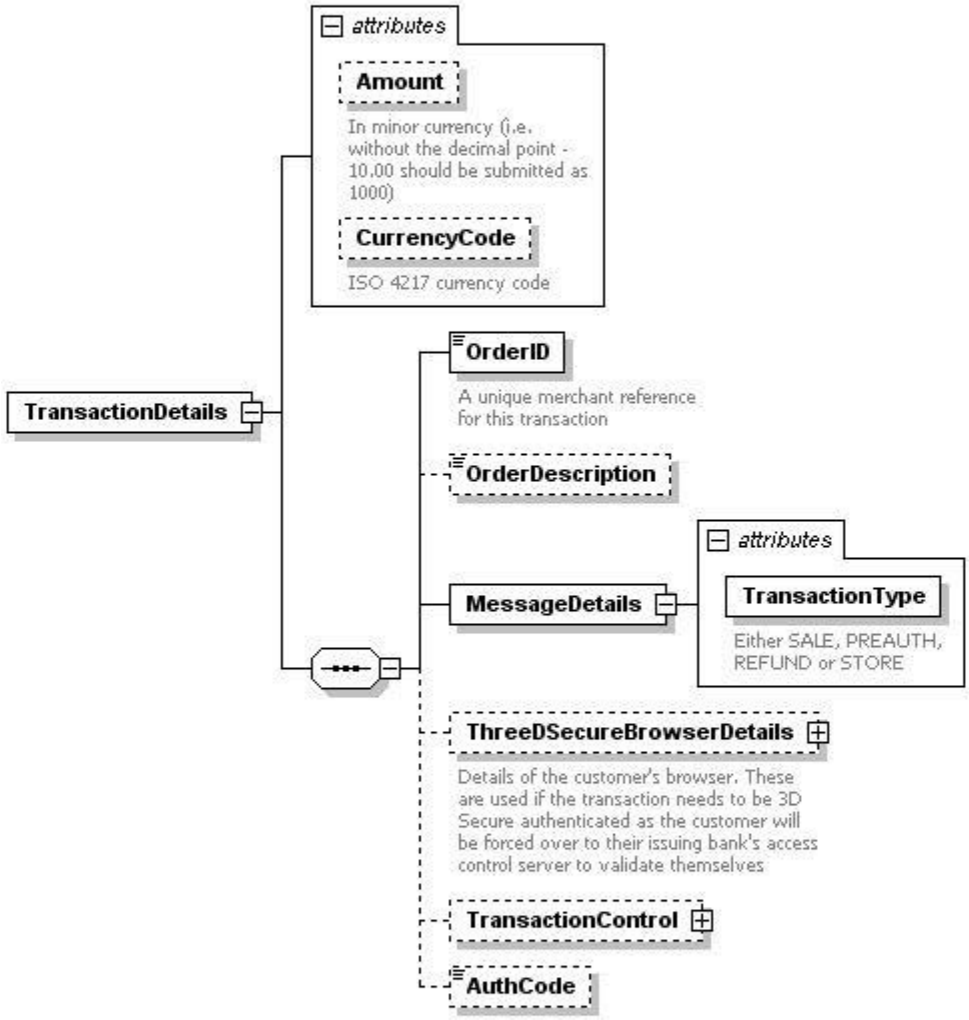
CardDetailsTransaction

Request – PaymentMessage

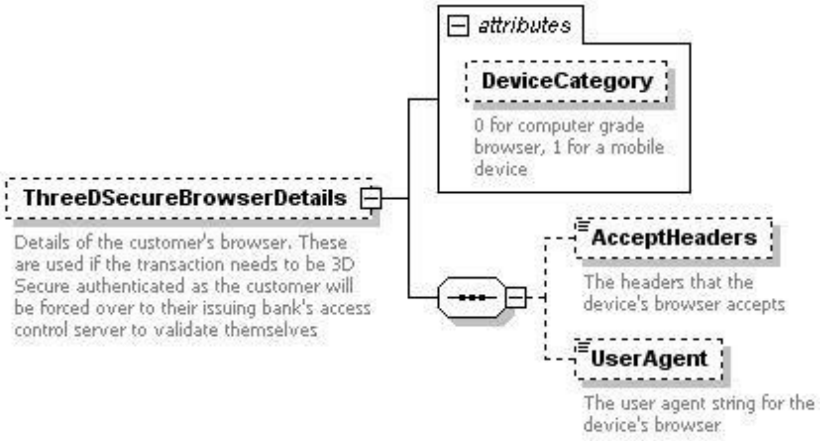
This describes the root PaymentMessage node of the CardDetailsTransaction request message



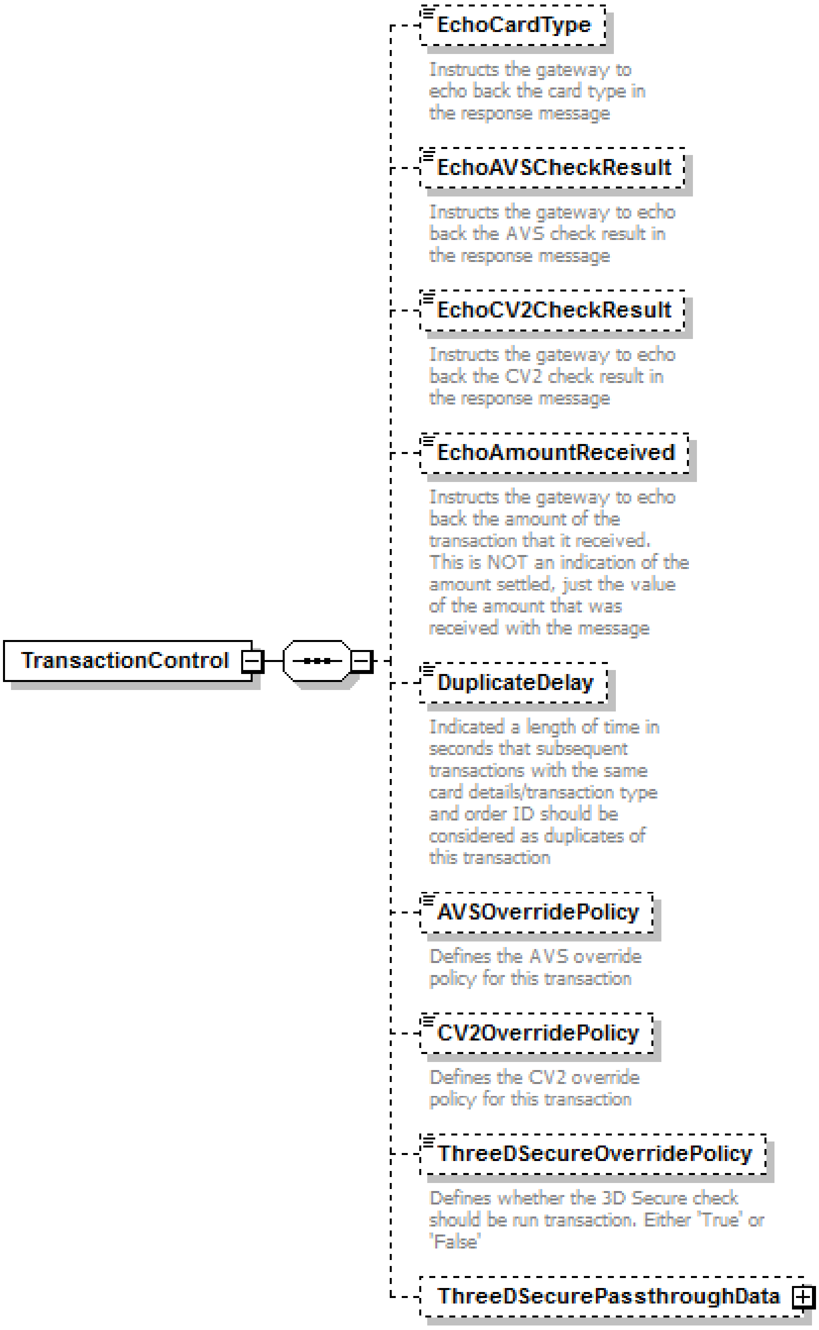
This describes the TransactionDetails child node of the root PaymentMessage node


attributes
Amount
In minor currency (i.e. without the decimal point - 10.00 should be submitted as 1000)
CurrencyCode
ISO 4217 currency code
TransactionDetails
OrderID
A unique merchant reference for this transaction
OrderDescription
attributes
MessageDetails
TransactionType
Either SALE, PREAUTH, REFUND or STORE
ThreeDSecureBrowserDetails
Details of the customer's browser. These are used if the transaction needs to be 3D Secure authenticated as the customer will be forced over to their issuing bank's access control server to validate themselves
TransactionControl
AuthCode

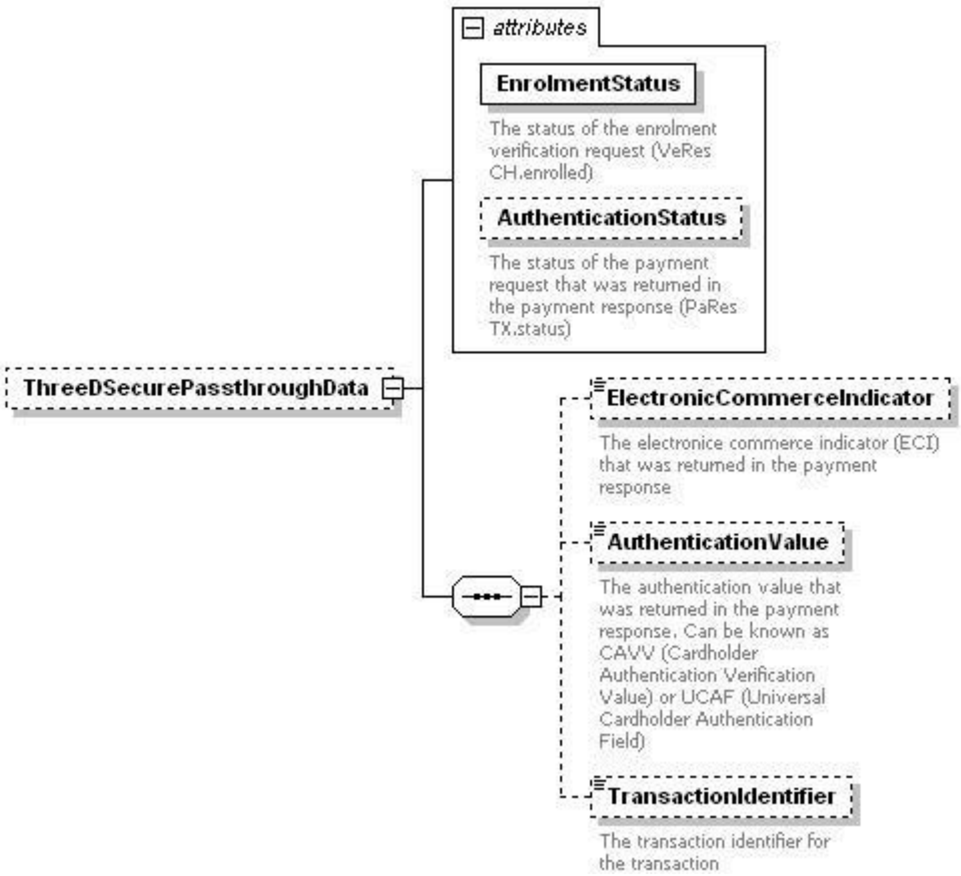
This describes the ThreeDSecureBrowserDetails child node of the TransactionDetails node


attributes
DeviceCategory
0 for computer grade browser, 1 for a mobile device
ThreeDSecureBrowserDetails
Details of the customer's browser. These are used if the transaction needs to be 3D Secure authenticated as the customer will be forced over to their issuing bank's access control server to validate themselves
AcceptHeaders
The headers that the device's browser accepts
UserAgent
The user agent string for the device's browser

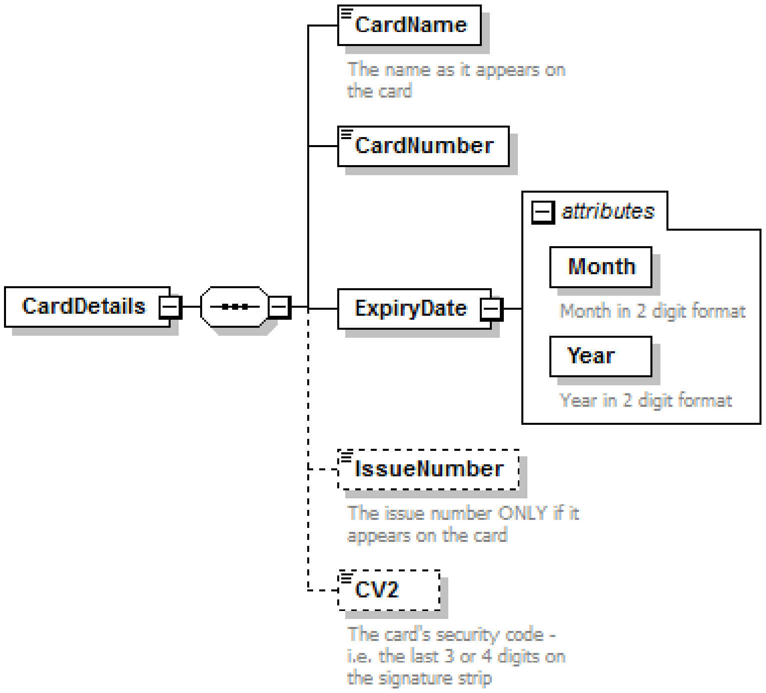
This describes the TransactionControl child node of the TransactionDetails node


EchoCardType
Instructs the gateway to echo back the card type in the response message
EchoAVSCheckResult
Instructs the gateway to echo back the AVS check result in the response message
EchoCV2CheckResult
Instructs the gateway to echo back the CV2 check result in the response message
EchoAmountReceived
Instructs the gateway to echo back the amount of the transaction that it received. This is NOT an indication of the amount settled, just the value of the amount that was received with the message
TransactionControl
DuplicateDelay
Indicated a length of time in seconds that subsequent transactions with the same card details/transaction type and order ID should be considered as duplicates of this transaction
AVSOverridePolicy
Defines the AVS override policy for this transaction
CV2OverridePolicy
Defines the CV2 override policy for this transaction
ThreeDSecureOverridePolicy
Defines whether the 3D Secure check should be run transaction. Either 'True' or 'False'
ThreeDSecurePassthroughData

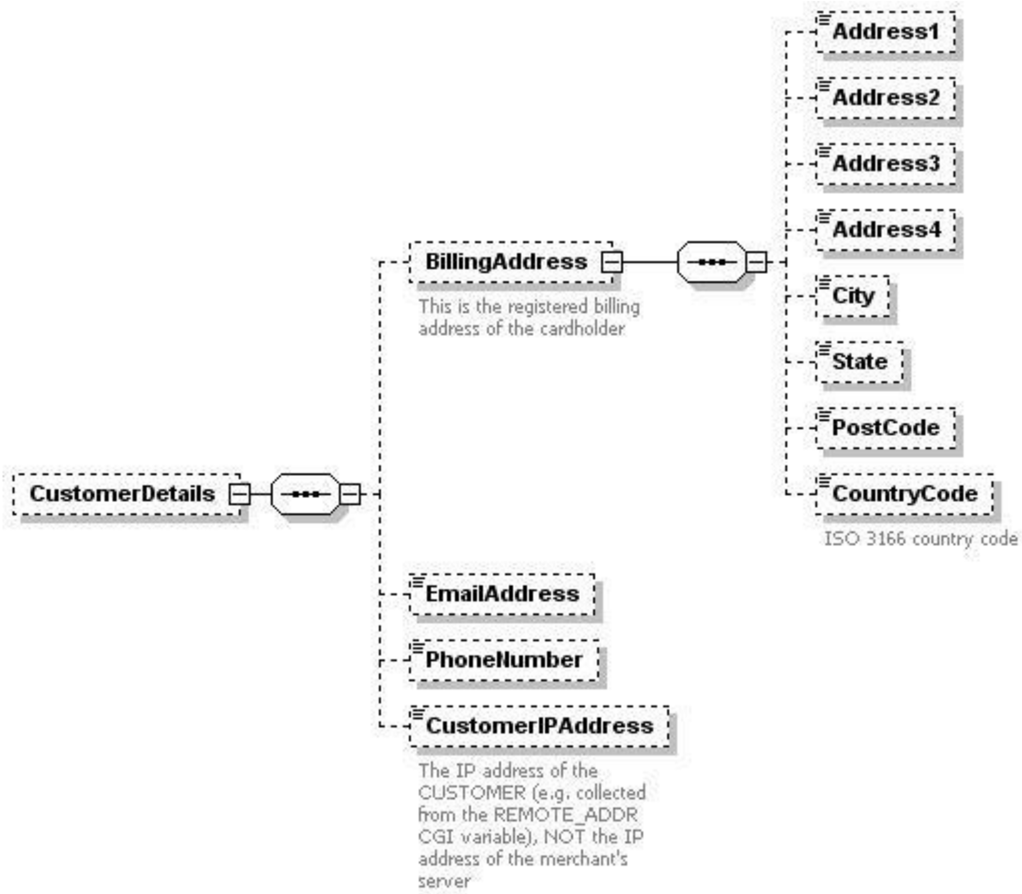
This describes the ThreeDSecurePassthroughData child node of the TransactionControl node


attributes
EnrolmentStatus
The status of the enrolment verification request (VeRes CH.enrolled)
AuthenticationStatus
The status of the payment request that was returned in the payment response (PaRes TX.status)
ThreeDSecurePassthroughData
ElectronicCommerceIndicator
The electronice commerce indicator (ECI) that was returned in the payment response
AuthenticationValue
The authentication value that was returned in the payment response. Can be known as CAVV (Cardholder Authentication Verification Value) or UCAF (Universal Cardholder Authentication Field)
TransactionIdentifier
The transaction identifier for the transaction

This describes the CardDetails child node of the root PaymentMessage node

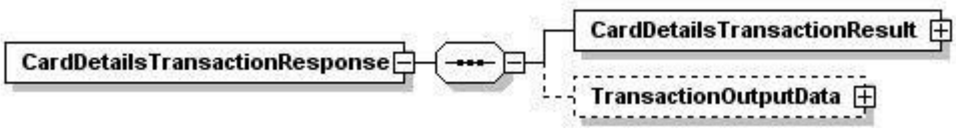

CardName
The name as it appears on the card
CardNumber
attributes
Month
Month in 2 digit format
Year
Year in 2 digit format
CardDetails
ExpiryDate
IssueNumber
The issue number ONLY if it appears on the card
CV2
The card's security code - i.e. the last 3 or 4 digits on the signature strip

This describes the CustomerDetails child node of the root PaymentMessage node

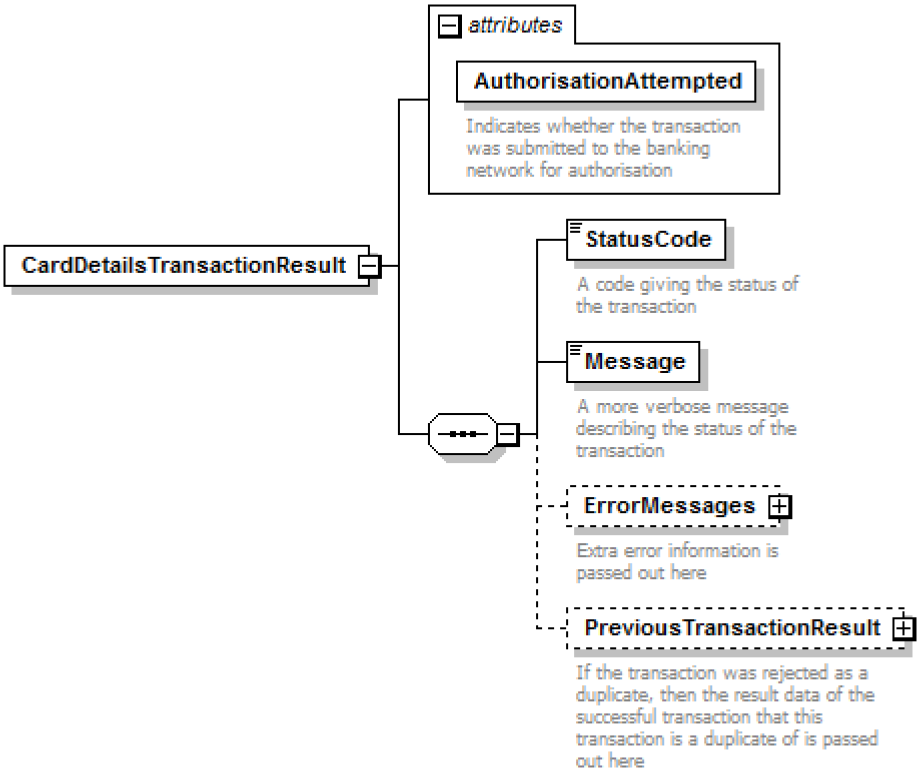

CustomerDetails
BillingAddress
This is the registered billing address of the cardholder
Address1
Address2
Address3
Address4
City
State
PostCode
CountryCode
ISO 3166 country code
EmailAddress
PhoneNumber
CustomerIPAddress
The IP address of the CUSTOMER (e.g. collected from the REMOTE_ADDR CGI variable), NOT the IP address of the merchant's server

Response – CardDetailsTransactionResponse

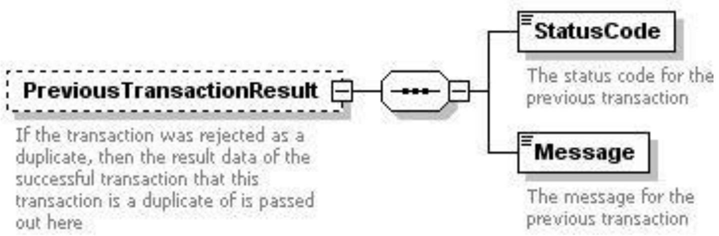
This describes the root CardDetailsTransactionResponse node of the CardDetailsTransaction response message



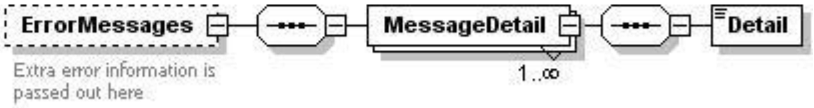
This describes the CardDetailsTransactionResult child node of the root CardDetailsTransactionResponse node


attributes
AuthorisationAttempted
Indicates whether the transaction was submitted to the banking network for authorisation
CardDetailsTransactionResult
StatusCode
A code giving the status of the transaction
Message
A more verbose message describing the status of the transaction
ErrorMessages
Extra error information is passed out here
PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here

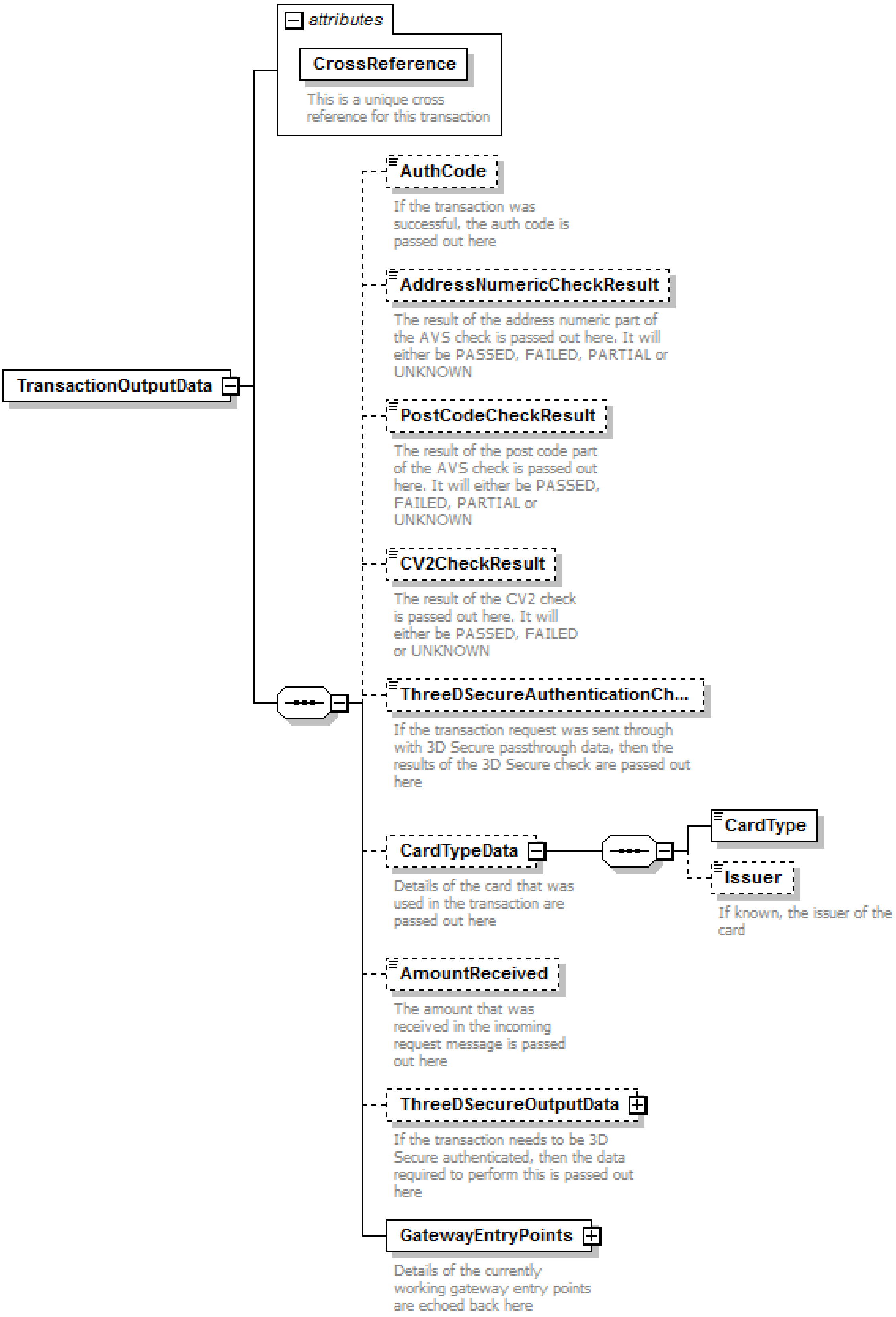
This describes the PreviousTransactionResult child node of the CardDetailsTransactionResult node


PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here
StatusCode
The status code for the previous transaction
Message
The message for the previous transaction

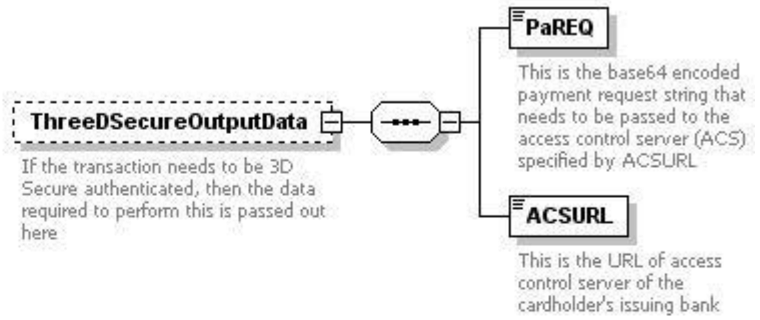
This describes the ErrorMessages child node of the CardDetailsTransactionResult node



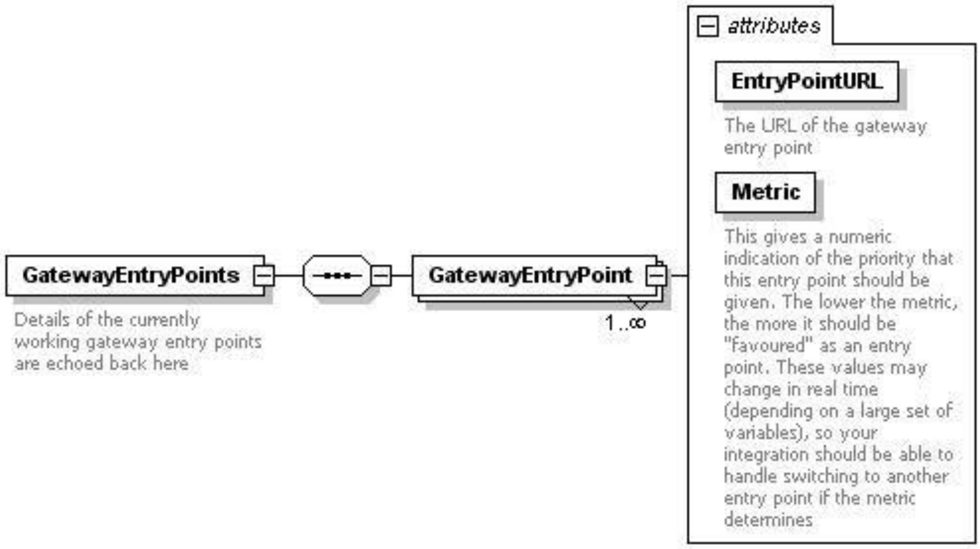
This describes the TransactionOutputData child node of the root CardDetailsTransactionResponse node


attributes
CrossReference
This is a unique cross reference for this transaction
TransactionOutputData
AuthCode
If the transaction was successful, the auth code is passed out here
AddressNumericCheckResult
The result of the address numeric part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
PostCodeCheckResult
The result of the post code part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
CV2CheckResult
The result of the CV2 check is passed out here. It will either be PASSED, FAILED or UNKNOWN
ThreeDSecureAuthenticationCh...
If the transaction request was sent through with 3D Secure passthrough data, then the results of the 3D Secure check are passed out here
CardTypeData
Details of the card that was used in the transaction are passed out here
CardType
Issuer
If known, the issuer of the card
AmountReceived
The amount that was received in the incoming request message is passed out here
ThreeDSecureOutputData
If the transaction needs to be 3D Secure authenticated, then the data required to perform this is passed out here
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here

This describes the ThreeDSecureOutputData child node of the TransactionOutputData node


ThreeDSecureOutputData
If the transaction needs to be 3D Secure authenticated, then the data required to perform this is passed out here
PaREQ
This is the base64 encoded payment request string that needs to be passed to the access control server (ACS) specified by ACSURL
ACSURL
This is the URL of access control server of the cardholder's issuing bank

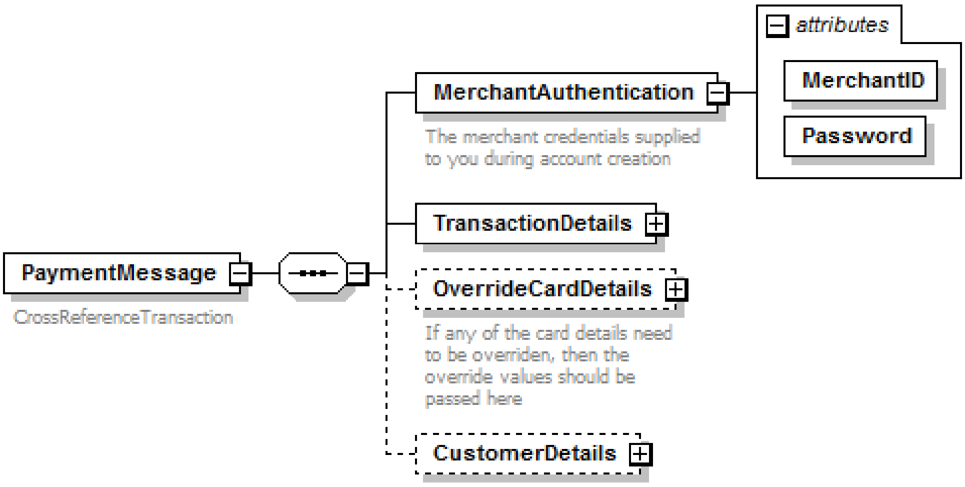
This describes the GatewayEntryPoints child node of the root TransactionOutputData node


attributes
EntryPointURL
The URL of the gateway entry point
Metric
This gives a numeric indication of the priority that this entry point should be given. The lower the metric, the more it should be "favoured" as an entry point. These values may change in real time (depending on a large set of variables), so your integration should be able to handle switching to another entry point if the metric determines
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here
GatewayEntryPoint
1..∞

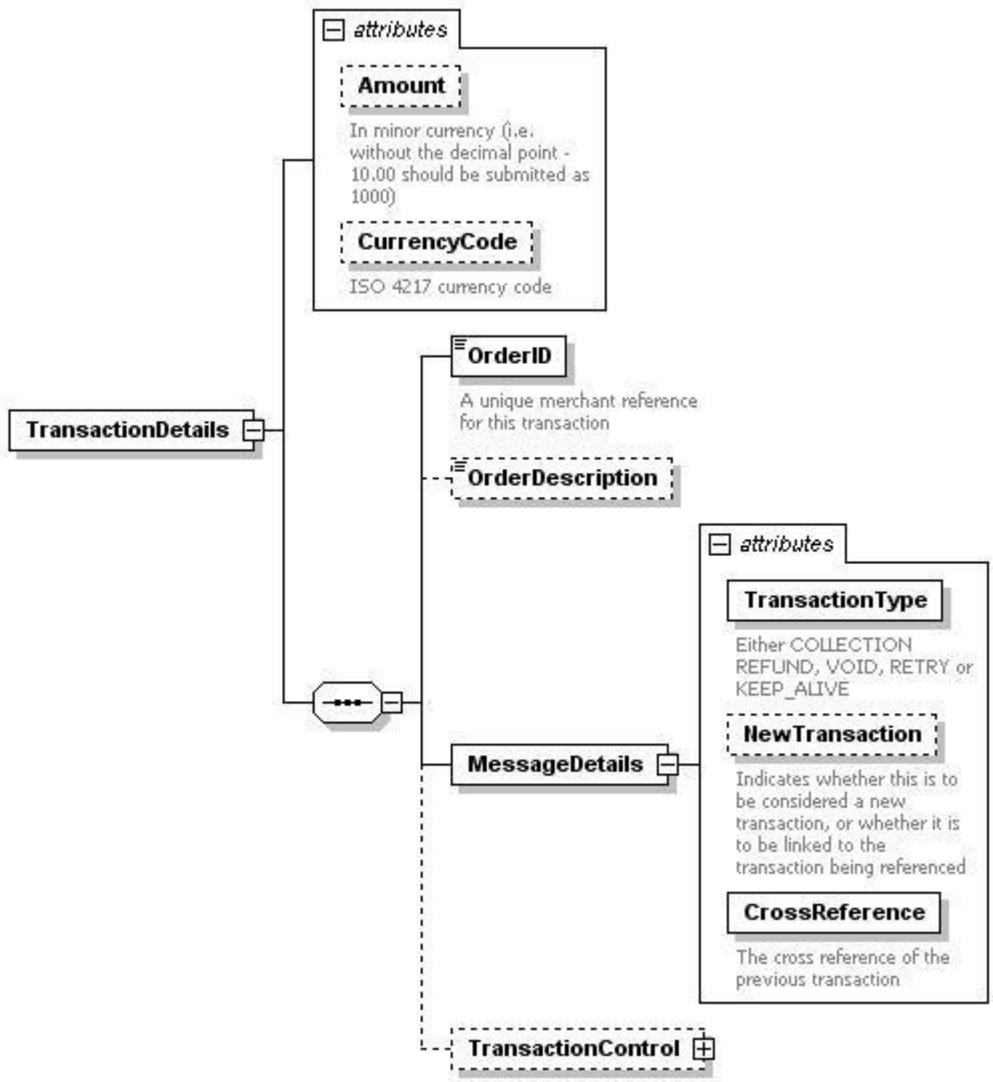
CrossReferenceTransaction

Request – PaymentMessage

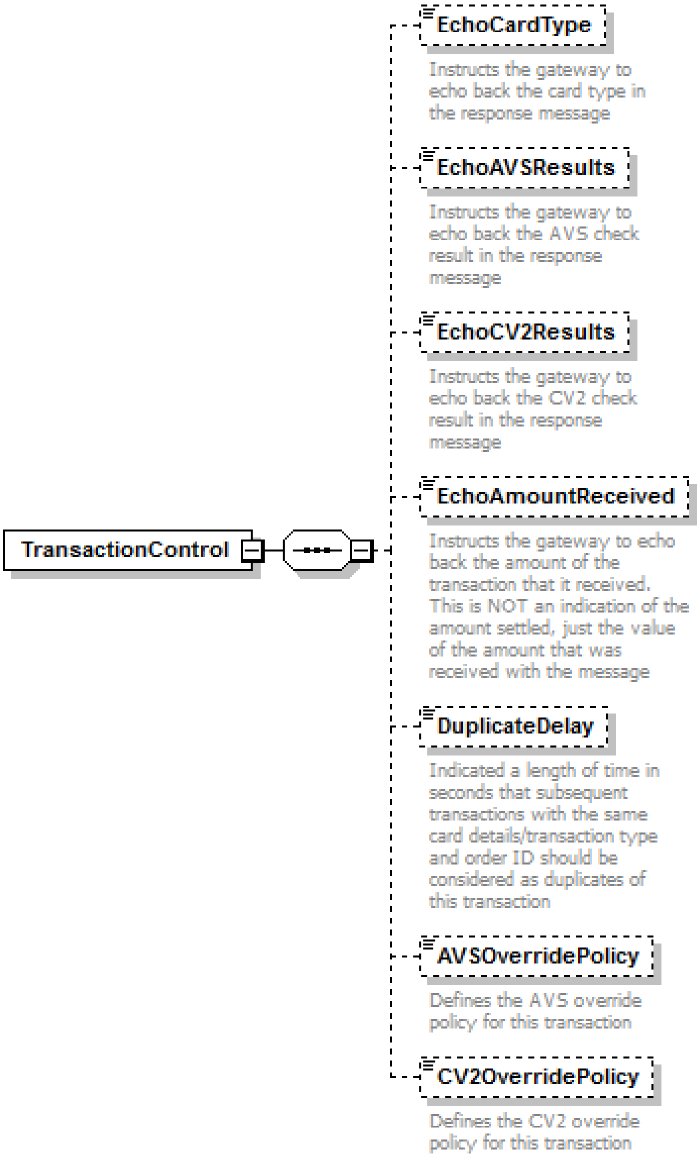
This describes the root PaymentMessage node of the CrossReferenceTransaction request message


attributes
MerchantID
Password
MerchantAuthentication
The merchant credentials supplied to you during account creation
TransactionDetails
PaymentMessage
CrossReferenceTransaction
OverrideCardDetails
If any of the card details need to be overriden, then the override values should be passed here
CustomerDetails

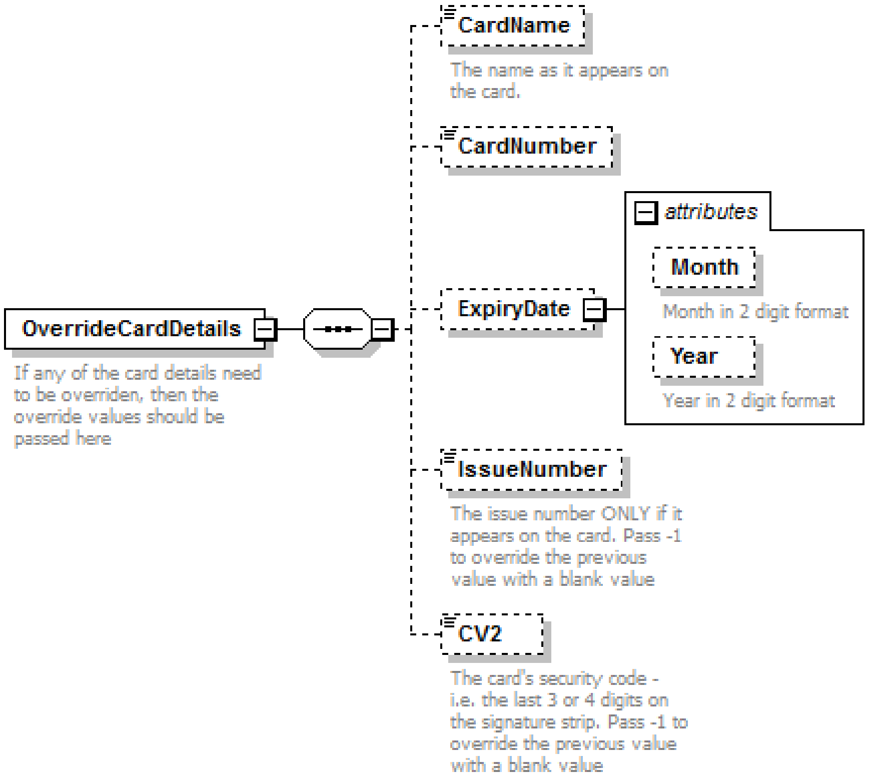
This describes the TransactionDetails child node of the root PaymentMessage node


attributes
Amount
In minor currency (i.e. without the decimal point - 10.00 should be submitted as 1000)
CurrencyCode
ISO 4217 currency code
TransactionDetails
OrderID
A unique merchant reference for this transaction
OrderDescription
attributes
TransactionType
Either COLLECTION REFUND, VOID, RETRY or KEEP_ALIVE
NewTransaction
Indicates whether this is to be considered a new transaction, or whether it is to be linked to the transaction being referenced
CrossReference
The cross reference of the previous transaction
MessageDetails
TransactionControl

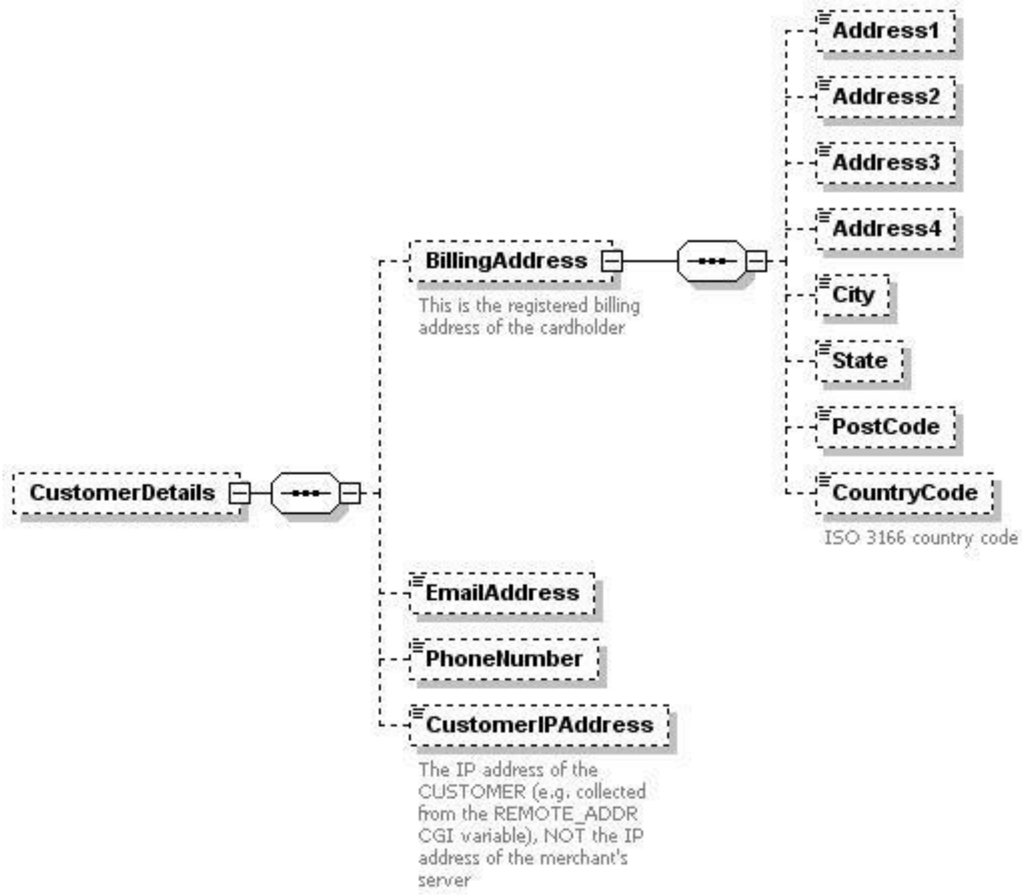
This describes the TransactionControl child node of the TransactionDetails node


TransactionControl
EchoCardType
Instructs the gateway to echo back the card type in the response message
EchoAVSResults
Instructs the gateway to echo back the AVS check result in the response message
EchoCV2Results
Instructs the gateway to echo back the CV2 check result in the response message
EchoAmountReceived
Instructs the gateway to echo back the amount of the transaction that it received. This is NOT an indication of the amount settled, just the value of the amount that was received with the message
DuplicateDelay
Indicated a length of time in seconds that subsequent transactions with the same card details/transaction type and order ID should be considered as duplicates of this transaction
AVSOverridePolicy
Defines the AVS override policy for this transaction
CV2OverridePolicy
Defines the CV2 override policy for this transaction

This describes the OverrideCardDetails child node of the root PaymentMessage node

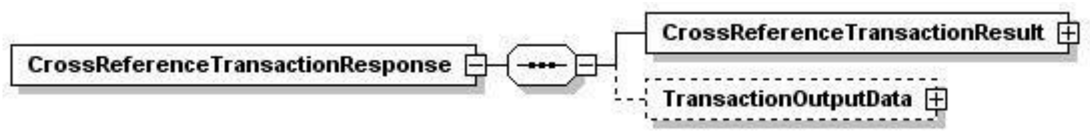

CardName
The name as it appears on the card.
CardNumber
attributes
Month
Month in 2 digit format
OverrideCardDetails
If any of the card details need to be overriden, then the override values should be passed here
ExpiryDate
Year
Year in 2 digit format
IssueNumber
The issue number ONLY if it appears on the card. Pass -1 to override the previous value with a blank value
CV2
The card's security code - i.e. the last 3 or 4 digits on the signature strip. Pass -1 to override the previous value with a blank value

This describes the CustomerDetails child node of the root PaymentMessage node

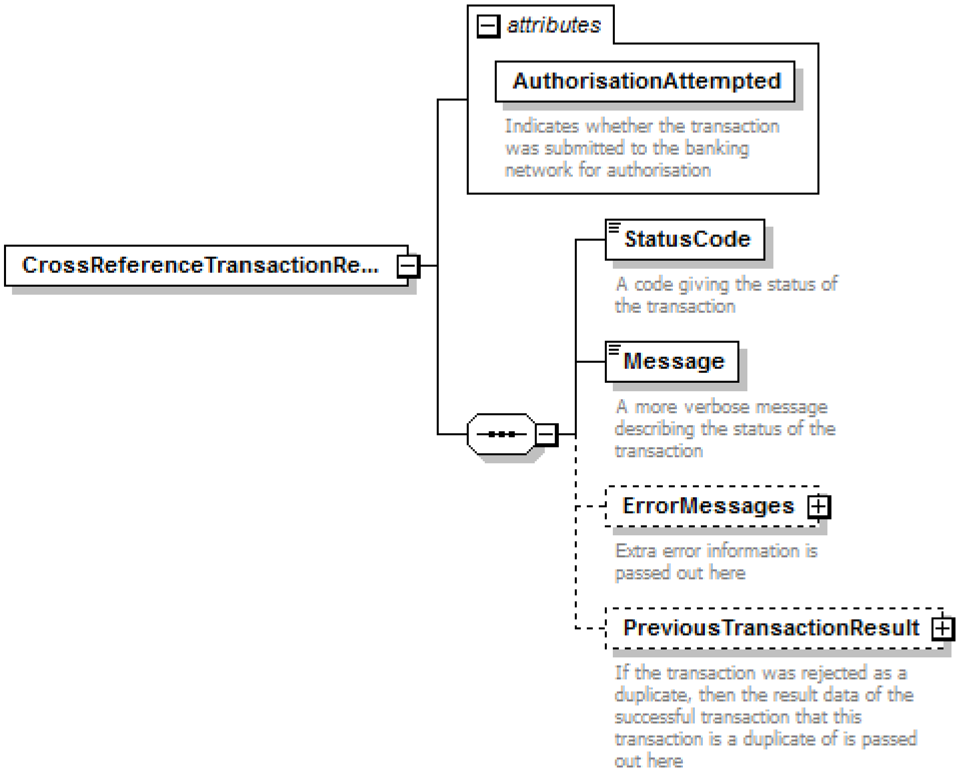

CustomerDetails
BillingAddress
This is the registered billing address of the cardholder
Address1
Address2
Address3
Address4
City
State
PostCode
CountryCode
ISO 3166 country code
EmailAddress
PhoneNumber
CustomerIPAddress
The IP address of the CUSTOMER (e.g. collected from the REMOTE_ADDR CGI variable), NOT the IP address of the merchant's server

Response - CrossReferenceTransaction

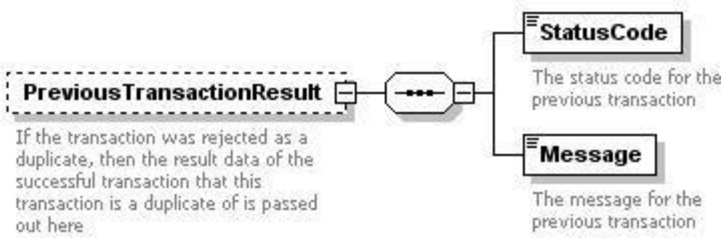
This describes the root CrossReferenceTransactionResponse node of the CrossReferenceTransaction response message


CrossReferenceTransactionResponse
CrossReferenceTransactionResult
TransactionOutputData

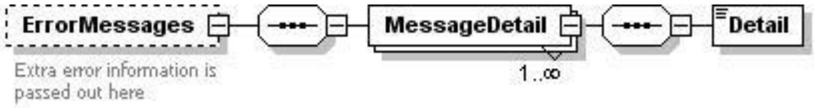
This describes the CrossReferenceTransactionResult child node of the root CrossReferenceTransactionResponse node


attributes
AuthorisationAttempted
Indicates whether the transaction was submitted to the banking network for authorisation
CrossReferenceTransactionRe...
StatusCode
A code giving the status of the transaction
Message
A more verbose message describing the status of the transaction
ErrorMessages
Extra error information is passed out here
PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here

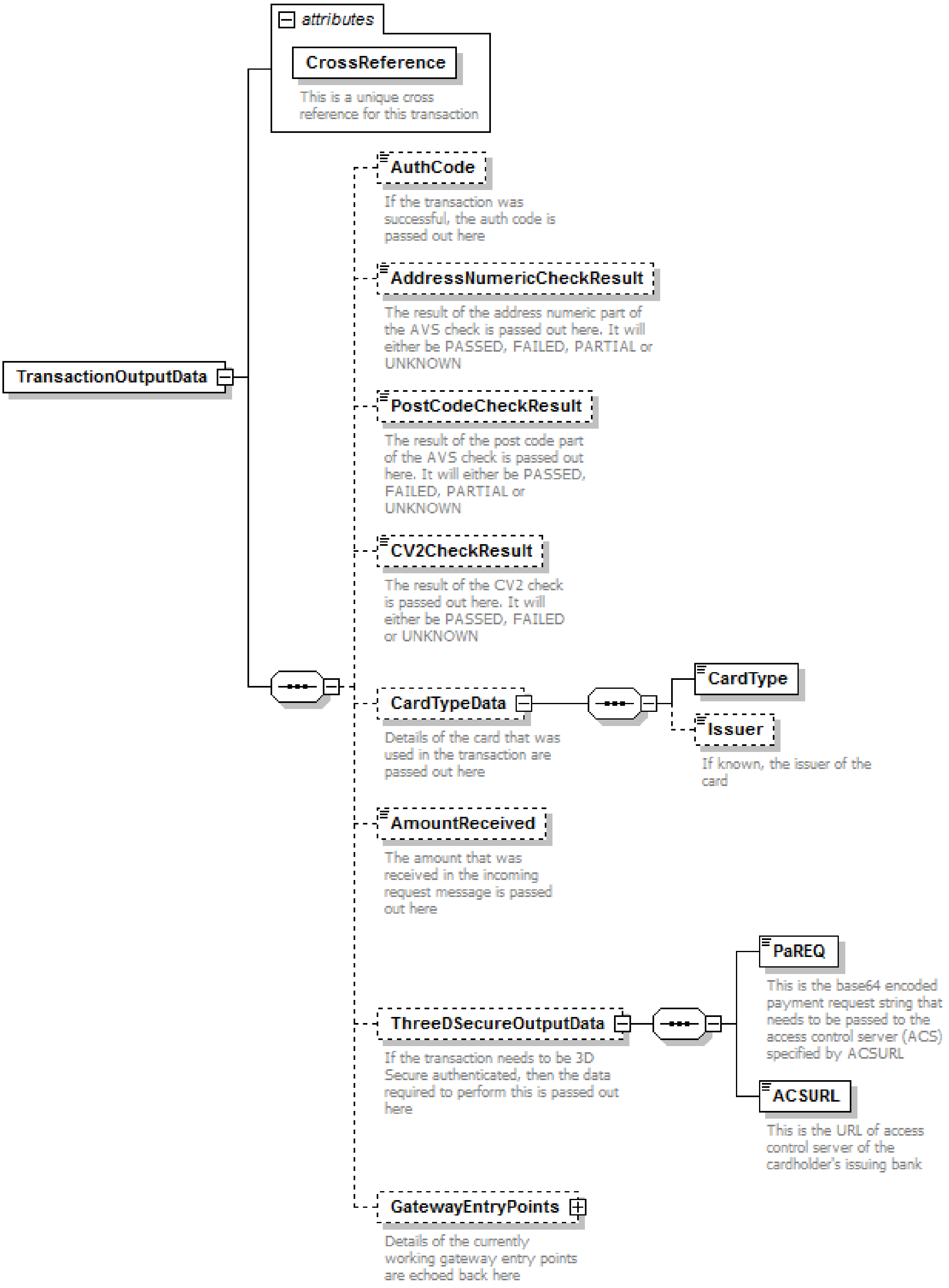
This describes the PreviousTransactionResult child node of the CardDetailsTransactionResult node


PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here
StatusCode
The status code for the previous transaction
Message
The message for the previous transaction

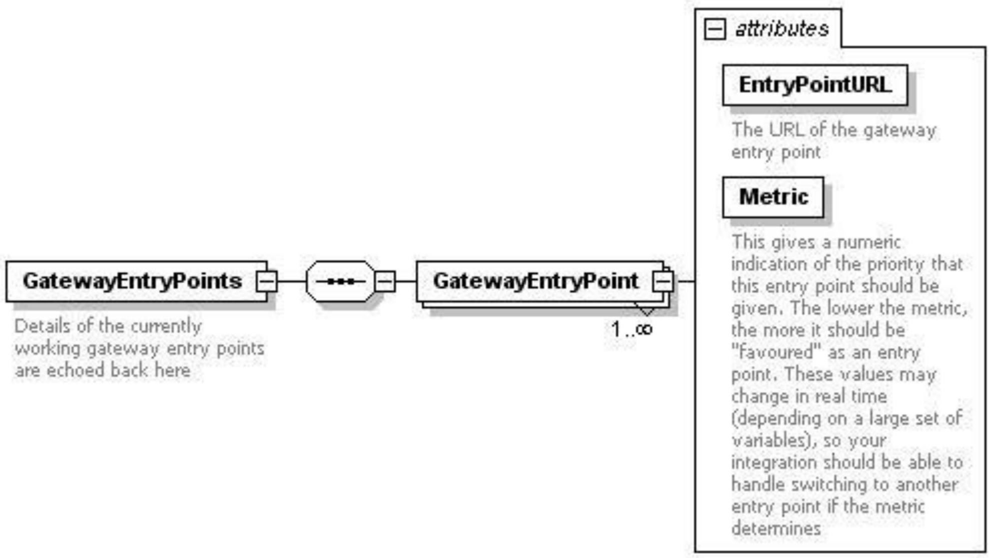
This describes the ErrorMessages child node of the CrossReferenceTransactionResult node



This describes the TransactionOutputData child node of the root CrossReferenceTransactionResponse node


attributes
CrossReference
This is a unique cross reference for this transaction
TransactionOutputData
AuthCode
If the transaction was successful, the auth code is passed out here
AddressNumericCheckResult
The result of the address numeric part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
PostCodeCheckResult
The result of the post code part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
CV2CheckResult
The result of the CV2 check is passed out here. It will either be PASSED, FAILED or UNKNOWN
CardTypeData
Details of the card that was used in the transaction are passed out here
CardType
Issuer
If known, the issuer of the card
AmountReceived
The amount that was received in the incoming request message is passed out here
ThreeDSecureOutputData
If the transaction needs to be 3D Secure authenticated, then the data required to perform this is passed out here
PaREQ
This is the base64 encoded payment request string that needs to be passed to the access control server (ACS) specified by ACSURL
ACSURL
This is the URL of access control server of the cardholder's issuing bank
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here

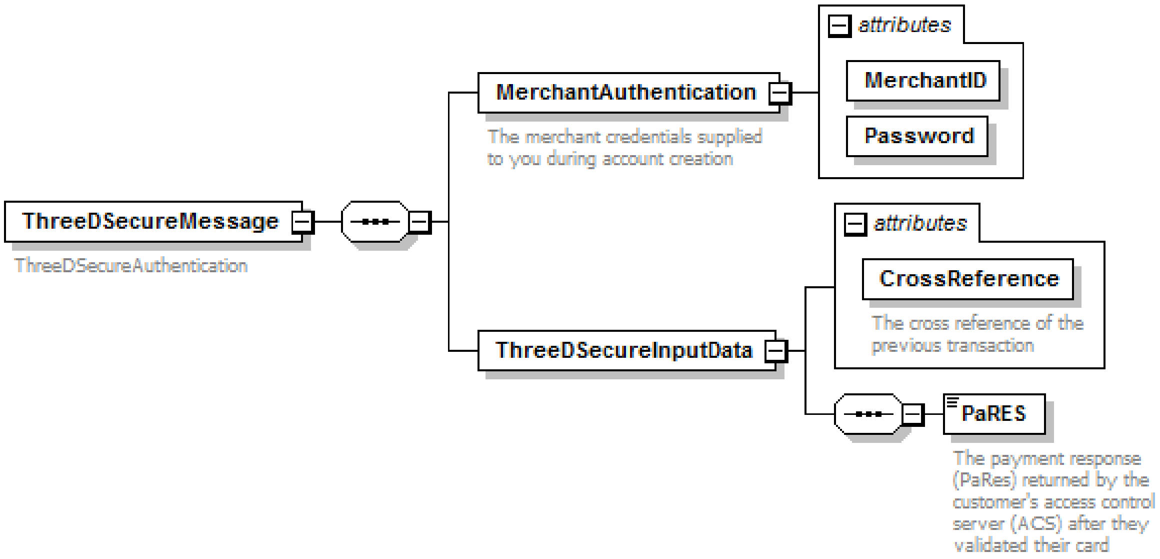
This describes the GatewayEntryPoints child node of the root TransactionOutputData node


attributes
EntryPointURL
The URL of the gateway entry point
Metric
This gives a numeric indication of the priority that this entry point should be given. The lower the metric, the more it should be "favoured" as an entry point. These values may change in real time (depending on a large set of variables), so your integration should be able to handle switching to another entry point if the metric determines
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here
GatewayEntryPoint
1..∞

ThreeDSecureAuthentication

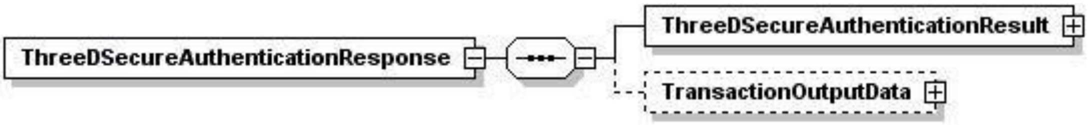
Request – ThreeDSecureMessage

This describes the root ThreeDSecureMessage node of the ThreeDSecureAuthentication request message

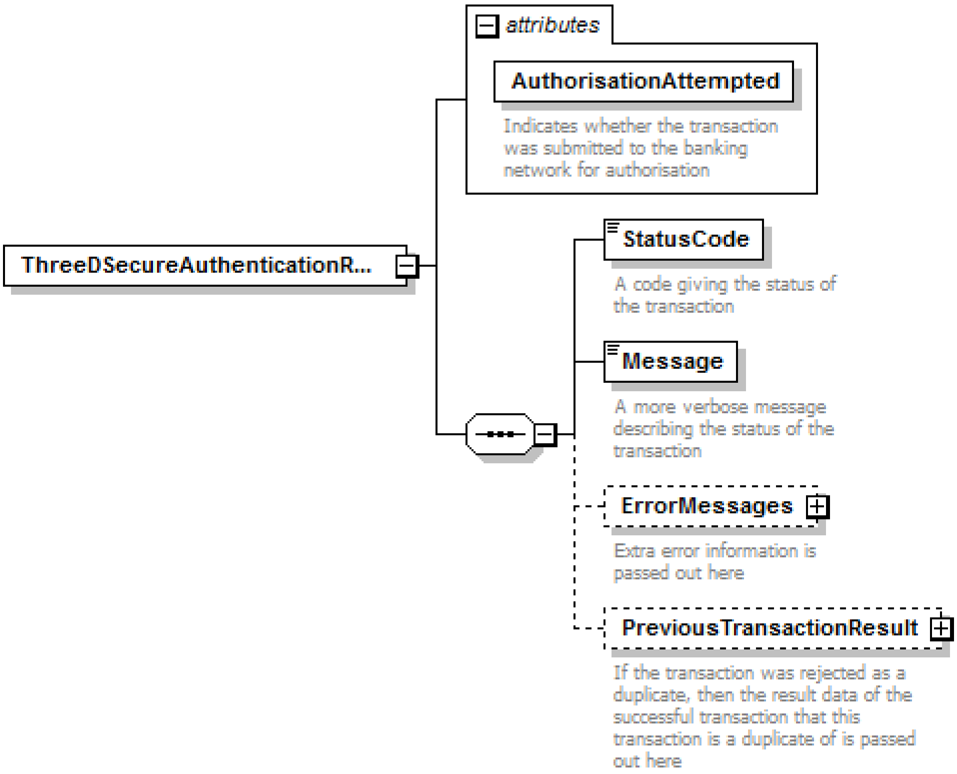

ThreeDSecureMessage
ThreeDSecureAuthentication
MerchantAuthentication
The merchant credentials supplied to you during account creation
attributes
MerchantID
Password
ThreeDSecureInputData
attributes
CrossReference
The cross reference of the previous transaction
PaRES
The payment response (PaRes) returned by the customer's access control server (ACS) after they validated their card

Response – ThreeDSecureAuthentication

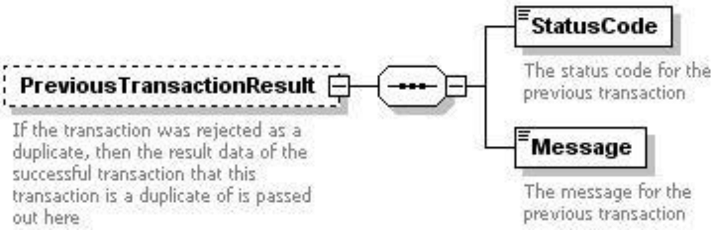
This describes the root ThreeDSecureAuthenticationResponse node of the ThreeDSecureAuthentication response message


ThreeDSecureAuthenticationResponse
ThreeDSecureAuthenticationResult
TransactionOutputData

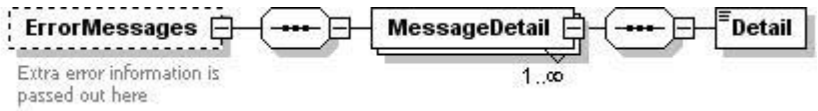
This describes the ThreeDSecureAuthenticationResult child node of the root ThreeDSecureAuthenticationResponse node


attributes
AuthorisationAttempted
Indicates whether the transaction was submitted to the banking network for authorisation
ThreeDSecureAuthenticationR...
StatusCode
A code giving the status of the transaction
Message
A more verbose message describing the status of the transaction
ErrorMessages
Extra error information is passed out here
PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here

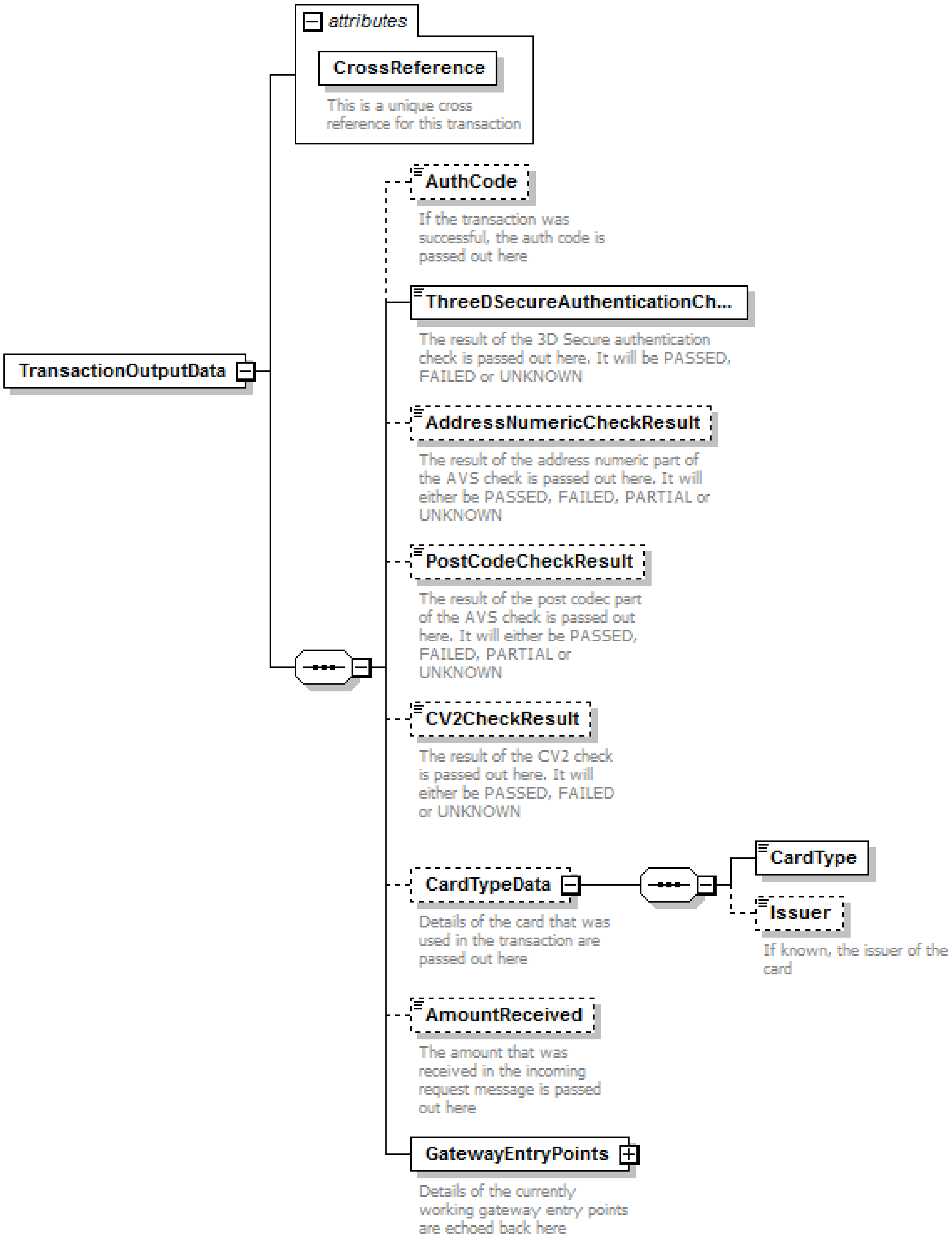
This describes the PreviousTransactionResult child node of the CardDetailsTransactionResult node


PreviousTransactionResult
If the transaction was rejected as a duplicate, then the result data of the successful transaction that this transaction is a duplicate of is passed out here
StatusCode
The status code for the previous transaction
Message
The message for the previous transaction

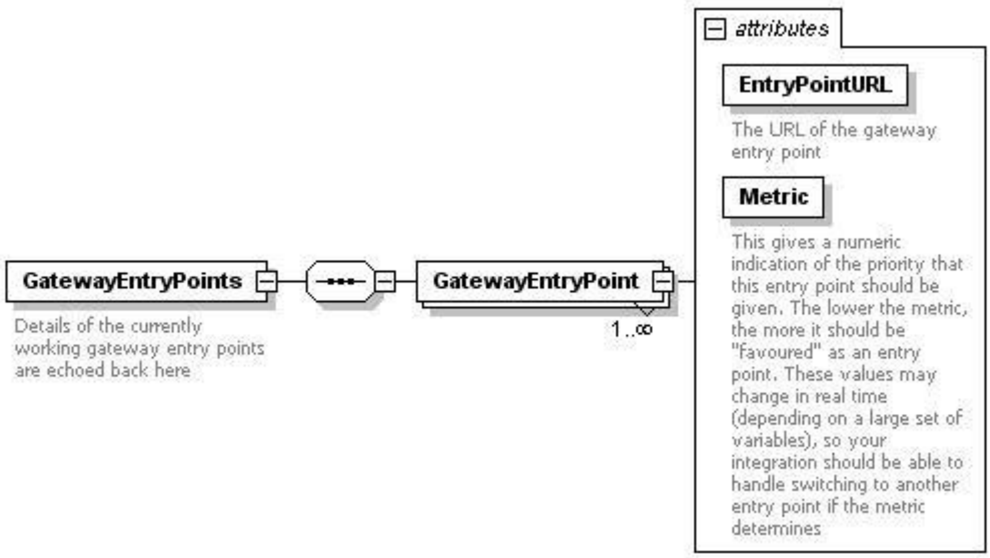
This describes the ErrorMessages child node of the ThreeDSecureAuthenticationResult node



This describes the TransactionOutputData child node of the root ThreeDSecureAuthenticationResponse node


attributes
CrossReference
This is a unique cross reference for this transaction
AuthCode
If the transaction was successful, the auth code is passed out here
ThreeDSecureAuthenticationCh...
The result of the 3D Secure authentication check is passed out here. It will be PASSED, FAILED or UNKNOWN
TransactionOutputData
AddressNumericCheckResult
The result of the address numeric part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
PostCodeCheckResult
The result of the post codec part of the AVS check is passed out here. It will either be PASSED, FAILED, PARTIAL or UNKNOWN
CV2CheckResult
The result of the CV2 check is passed out here. It will either be PASSED, FAILED or UNKNOWN
CardTypeData
Details of the card that was used in the transaction are passed out here
CardType
Issuer
If known, the issuer of the card
AmountReceived
The amount that was received in the incoming request message is passed out here
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here

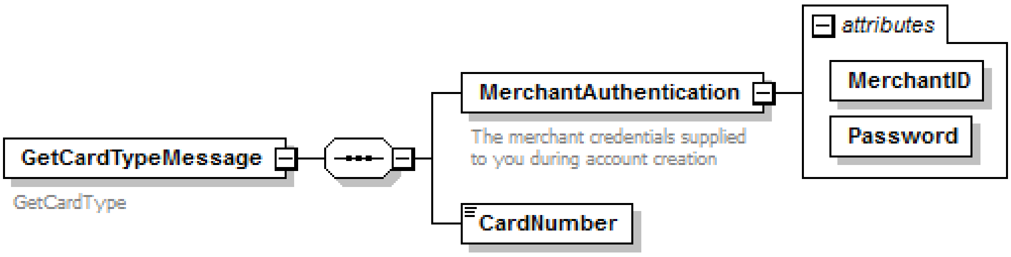
This describes the GatewayEntryPoints child node of the root TransactionOutputData node


attributes
EntryPointURL
The URL of the gateway entry point
Metric
This gives a numeric indication of the priority that this entry point should be given. The lower the metric, the more it should be "favoured" as an entry point. These values may change in real time (depending on a large set of variables), so your integration should be able to handle switching to another entry point if the metric determines
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here
GatewayEntryPoint
1..∞

GetCardType

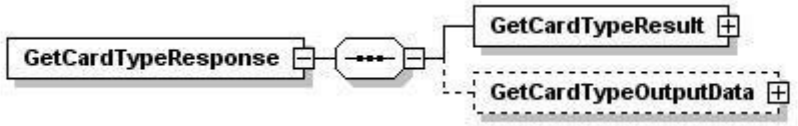
Request – GetCardTypeMessage

This describes the root GetCardTypeMessage node of the GetCardType request message

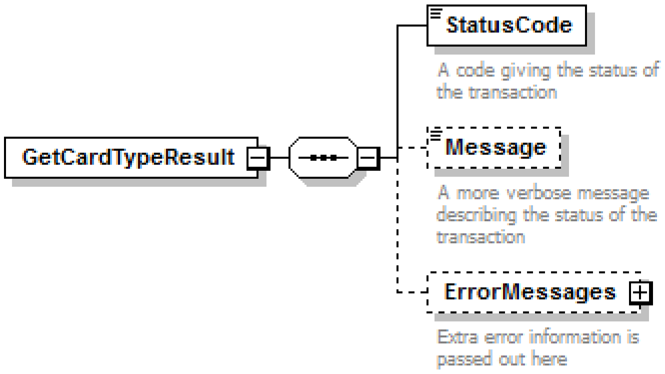


Response – GetCardTypeResponse

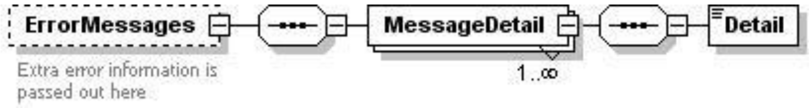
This describes the root GetCardTypeResponse node of the GetCardType response message



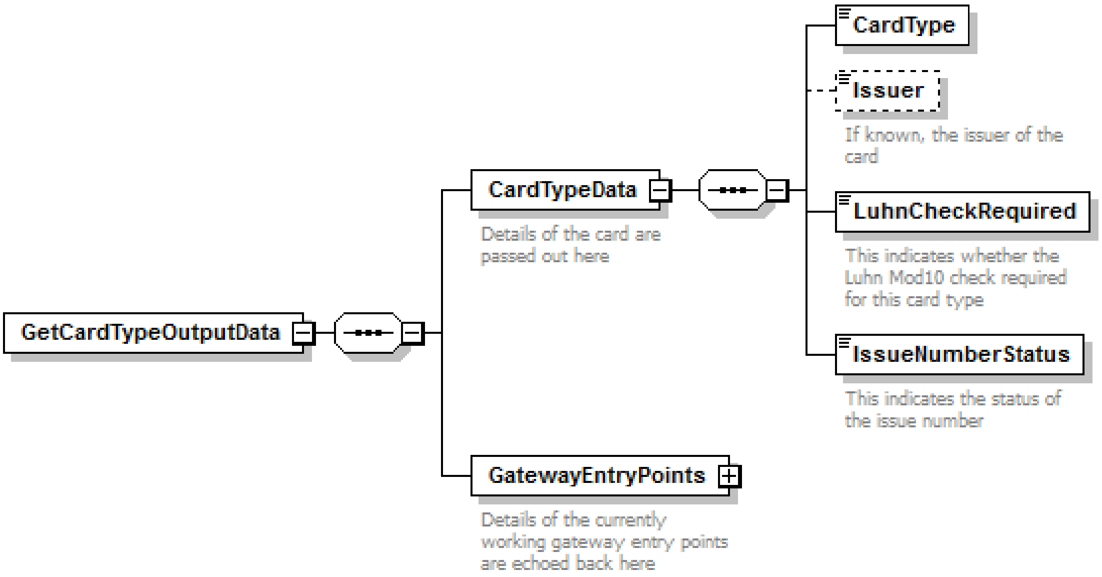
This describes the GetCardTypeResult child node of the root GetCardTypeResponse node


StatusCode
A code giving the status of the transaction
GetCardTypeResult
Message
A more verbose message describing the status of the transaction
ErrorMessages
Extra error information is passed out here

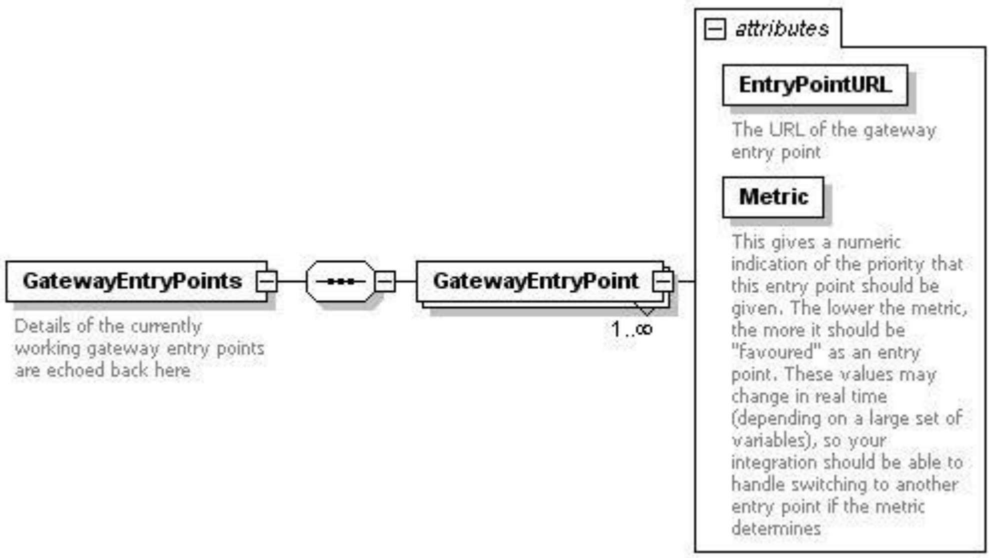
This describes the ErrorMessages child node of the root GetCardTypeResult node



This describes the GetCardTypeOutputData child node of the root GetCardTypeResponse node


GetCardTypeOutputData
CardTypeData
Details of the card are passed out here
CardType
Issuer
If known, the issuer of the card
LuhnCheckRequired
This indicates whether the Luhn Mod10 check required for this card type
IssueNumberStatus
This indicates the status of the issue number
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here

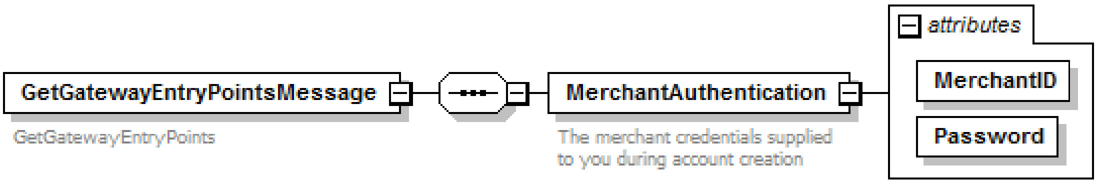
This describes the GatewayEntryPoints child node of the root GetCardTypeOutputData node


attributes
EntryPointURL
The URL of the gateway entry point
Metric
This gives a numeric indication of the priority that this entry point should be given. The lower the metric, the more it should be "favoured" as an entry point. These values may change in real time (depending on a large set of variables), so your integration should be able to handle switching to another entry point if the metric determines
GatewayEntryPoints
Details of the currently working gateway entry points are echoed back here
GatewayEntryPoint
1..∞

GetGatewayEntryPoints

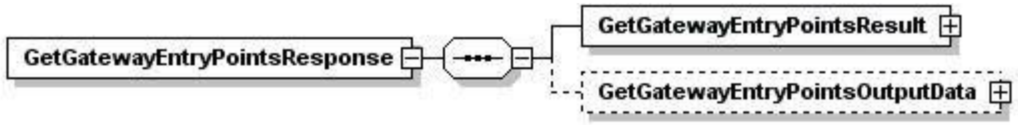
Request – GetGatewayEntryPointsMessage

This describes the root GetGatewayEntryPointsMessage node of the GetGatewayEntryPointsrequest message

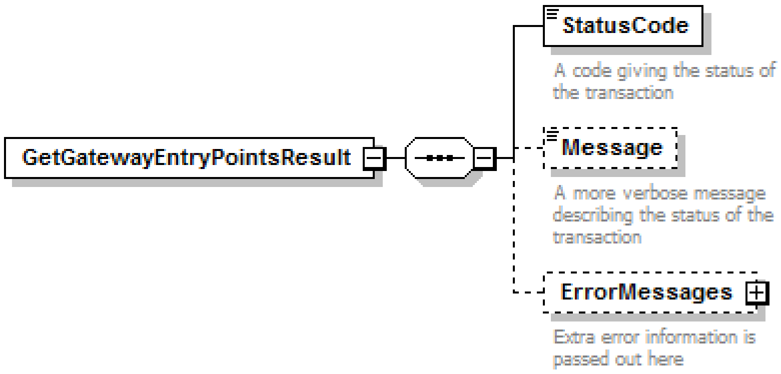

GetGatewayEntryPointsMessage
GetGatewayEntryPoints
MerchantAuthentication
The merchant credentials supplied to you during account creation
attributes
MerchantID
Password

Response – GetGatewayEntryPointsResponse

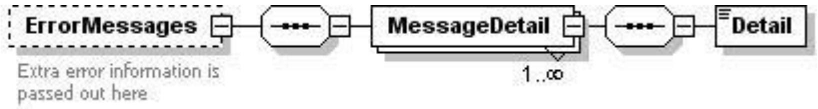
This describes the root GetCardTypeMessage node of the GetCardType request message


GetGatewayEntryPointsResponse
GetGatewayEntryPointsResult
GetGatewayEntryPointsOutputData

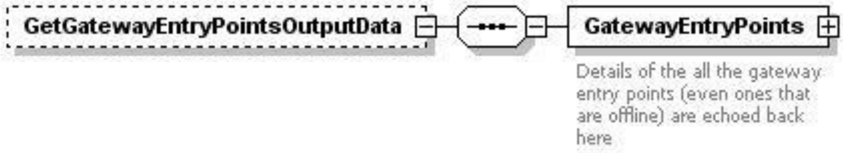
This describes the GetGatewayEntryPointsResult child node of the root GetGatewayEntryPointsResponse


GetGatewayEntryPointsResult
StatusCode
A code giving the status of the transaction
Message
A more verbose message describing the status of the transaction
ErrorMessages
Extra error information is passed out here

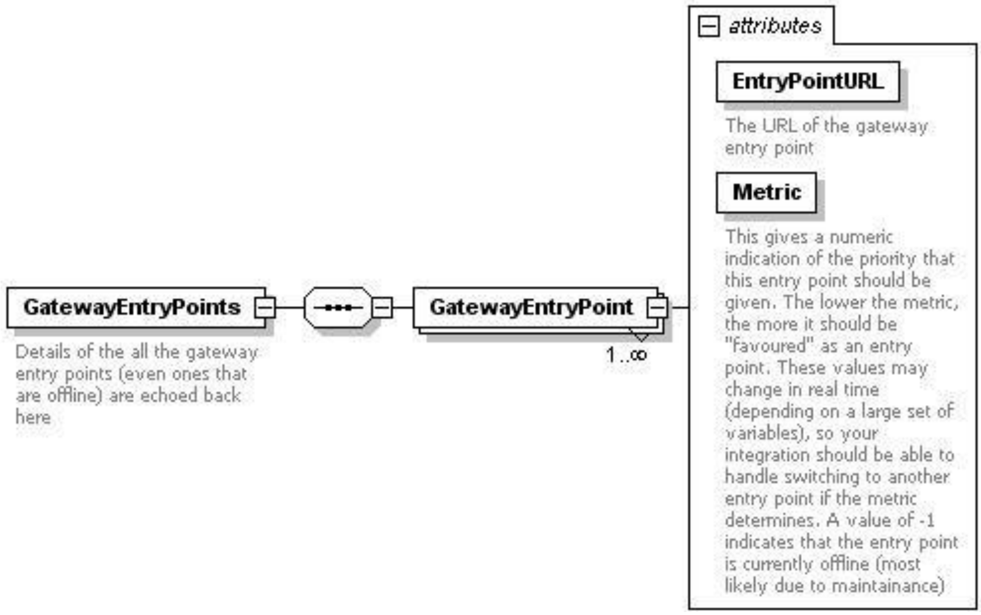
This describes the ErrorMessages child node of the root GetGatewayEntryPointsResult node



This describes the GetGatewayEntryPointsOutputData child node of the root GetGatewayEntryPointsResult


GetGatewayEntryPointsOutputData
GatewayEntryPoints
Details of the all the gateway entry points (even ones that are offline) are echoed back here

This describes the GatewayEntryPoints child node of the root GetGatewayEntryPointsOutputData node


attributes
EntryPointURL
The URL of the gateway entry point
Metric
This gives a numeric indication of the priority that this entry point should be given. The lower the metric, the more it should be "favoured" as an entry point. These values may change in real time (depending on a large set of variables), so your integration should be able to handle switching to another entry point if the metric determines. A value of -1 indicates that the entry point is currently offline (most likely due to maintainance)
GatewayEntryPoints
Details of the all the gateway entry points (even ones that are offline) are echoed back here
GatewayEntryPoint
1..∞

Appendix 3: Example Messages

CardDetailsTransaction

Request – CardDetailsTransaction

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <CardDetailsTransaction xmlns="https://www.thepaymentgateway.net/">
        <PaymentMessage>
          <MerchantAuthentication MerchantID="MerchantID" Password="Password" />
          <TransactionDetails Amount="1000" CurrencyCode="826">
            <MessageDetails TransactionType="SALE" />
            <OrderID>ORDER-1234</OrderID>
            <OrderDescription>A Test Order</OrderDescription>
            <TransactionControl>
              <EchoCardType>TRUE</EchoCardType>
              <EchoAVSCheckResult>TRUE</EchoAVSCheckResult>
              <EchoCV2CheckResult>TRUE</EchoCV2CheckResult>
              <EchoAmountReceived>TRUE</EchoAmountReceived>
              <DuplicateDelay>20</DuplicateDelay>
              <AVSOverridePolicy>BPPF</AVSOverridePolicy>
              <CV2OverridePolicy>FF</CV2OverridePolicy>
              <ThreeDSecureOverridePolicy>FALSE</ThreeDSecureOverridePolicy>
            </TransactionControl>
          </TransactionDetails>
          <CardDetails>
            <CardName>Test Customer</CardName>
            <CardNumber>5600000000005390</CardNumber>
            <ExpiryDate Month="12" Year="09" />
            <CV2>123</CV2>
            <IssueNumber>1</IssueNumber>
          </CardDetails>
          <CustomerDetails>
            <BillingAddress>
              <Address1>123 Test Street</Address1>
              <Address2>Test Address Line 2</Address2>
              <Address3>Test Address Line 3</Address3>
              <Address4>Test Address Line 4</Address4>
              <City>Testville</City>
              <State>Middlesex</State>
              <PostCode>TW11 8TT</PostCode>
              <CountryCode>826</CountryCode>
            </BillingAddress>
            <EmailAddress>test@mycompanyname.net</EmailAddress>
            <PhoneNumber>020889898952</PhoneNumber>
            <CustomerIPAddress>123.123.123.123</CustomerIPAddress>
          </CustomerDetails>
        </PaymentMessage>
        </CardDetailsTransaction>
    </soap:Body>
  </soap:Envelope>
```

Response – CardDetailsTransactionResponse (Not requiring 3D Secure authentication)

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <CardDetailsTransactionResponse xmlns="https://www.thepaymentgateway.net/">
        <CardDetailsTransactionResult AuthorisationAttempted="true">
          <StatusCode>0</StatusCode>
          <Message>Auth Code: 123456</Message>
             </CardDetailsTransactionResult>
        <TransactionOutputData CrossReference="07010101010010101010102">
          <AuthCode>123456</AuthCode>
          <AddressNumericCheckResult>UNKNOWN</AddressNumericCheckResult>
          <PostCodeCheckResult>UNKNOWN</PostCodeCheckResult>
          <CV2CheckResult>PASSED</CV2CheckResult>
          <CardTypeData>
             <CardType>MAESTRO_INTERNATIONAL</CardType>
                 <Issuer>HSBC</Issuer>
          </CardTypeData>
          <AmountReceived>1000</AmountReceived>
          <GatewayEntryPoints>
             <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
             <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
          </GatewayEntryPoints>
        </TransactionOutputData>
          </CardDetailsTransactionResponse>
    </soap:Body>
  </soap:Envelope>
```

Response – CardDetailsTransactionResponse (Requiring 3D Secure authentication)

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <CardDetailsTransactionResponse xmlns="https://www.thepaymentgateway.net/">
        <CardDetailsTransactionResult AuthorisationAttempted="true">
            <StatusCode>3</StatusCode>
            <Message>Issuer authentication Required</Message>
                </CardDetailsTransactionResult>
        <TransactionOutputDataCrossReference="07010101010010101010102">
            <AddressNumericCheckResult>UNKNOWN</AddressNumericCheckResult>
            <PostCodeCheckResult>UNKNOWN</PostCodeCheckResult>
            <CV2CheckResult>UNKNOWN</CV2CheckResult>
            <CardTypeData>
                <CardType>MAESTRO_INTERNATIONAL</CardType>
                <Issuer>HSBC</Issuer>
            </CardTypeData>
            <AmountReceived>1000</AmountReceived>
            <ThreeDSecureOutputData>
                <PaREQ>
eJxdUtFygjAQfPcrGD+AJCBQnJgZKg/1Aeu0/kAGrjVtCRigaL++CRpBmWEme3vcLbuh+4MCSN8h7xSwmePQDJqGf4Ijit
W85m6ACY79EM8Nqeld8gbHy1mjX1CNqCQjLnY9iiy0dAYqP3DZ2oIu8fz4vNmyMIoWMaHoCke+BLVJWRyHHgkouqCRlbwE
tlGiEF3prCtVV4q3eiNFAzM25lUnW3VmT15IkQUj3akf1ve9Ky6jcj3JldBSZAirHj3Kp7vOFJrpopMoWJYm/fTd7pNTln
7/bb+SFUWmY+wveAvMwzgi2AscEix9ssQLiob6xKbSaGYEuxhrly5opGsjJLE9pmVamdjQKQUytz5YNDbAqa4k6G90erfz
RC00OXtVBSjnQ1Xl0tlD02rjy5rLsxZt6Jtdj+7Q9ctd8Hmrswx8TDx/4Uf6ibH58ytxp0notHCMySBKjNFRZGfqdfYmmq CGO8tmFN3f538EA8lt
                </PaREQ>
                <ACSURL>https://www.bank.com/acs</ACSURL>
            </ThreeDSecureOutputData>
            <GatewayEntryPoints>
                <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
                <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
            </GatewayEntryPoints>
        </TransactionOutputData>
            </CardDetailsTransactionResponse>
    </soap:Body>
</soap:Envelope>
```

CrossReferenceTransaction

Request – CrossReferenceTransaction

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <CrossReferenceTransaction xmlns="https://www.thepaymentgateway.net/">
        <PaymentMessage>
          <MerchantAuthentication MerchantID="MerchantID" Password="Password" />
                  <TransactionDetails Amount="1000" CurrencyCode="826">
            <MessageDetails TransactionType="COLLECTION" NewTransaction="FALSE"
                                 CrossReference="07010101010101010101010101" />
            <OrderID>ORDER-1234</OrderID>
                     <OrderDescription>A test order</OrderDescription>
            <TransactionControl>
               <EchoCardType>TRUE</EchoCardType>
               <EchoAVSCheckResult>TRUE</EchoAVSCheckResult>
               <EchoCV2CheckResult>TRUE</EchoCV2CheckResult>
               <EchoAmountReceived>TRUE</EchoAmountReceived>
               <DuplicateDelay>60</DuplicateDelay>
               <AVSOverridePolicy>BPPF</AVSOverridePolicy>
               <ThreeDSecureOverridePolicy>FALSE</ThreeDSecureOverridePolicy>
            </TransactionControl>
          </TransactionDetails>
        </PaymentMessage>
     </CrossReferenceTransaction>
    </soap:Body>
  </soap:Envelope>
```

Response – CrossReferenceTransactionResponse

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
    <CrossReferenceTransactionResponse xmlns="https://www.thepaymentgateway.net/">
      <CrossReferenceTransactionResult AuthorisationAttempted="True">
        <StatusCode>0</StatusCode>
        <Message>Auth Code:123456</Message>
      </CrossReferenceTransactionResult>
      <TransactionOutputData CrossReference="070101010100101010102">
        <AuthCode>123456</AuthCode>
        <AddressNumericCheckResult>PASSED</AddressNumericCheckResult>
        <PostCodeCheckResult>PASSED</PostCodeCheckResult>
        <CV2CheckResult>PASSED</CV2CheckResult>
        <CardTypeData>
          <CardType>MAESTRO_INTERNATIONAL</CardType>
          <Issuer>HSBC</Issuer>
        </CardTypeData>
        <AmountReceived>1000</AmountReceived>
        <GatewayEntryPoints>
          <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
          <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
        </GatewayEntryPoints>
      </TransactionOutputData>
        </CrossReferenceTransactionResponse>
   </soap:Body>
 </soap:Envelope>
```

ThreeDSecureAuthentication

Request – ThreeDSecureAuthentication

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                              xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                              xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <ThreeDSecureAuthentication xmlns="https://www.thepaymentgateway.net/">
        <ThreeDSecureMessage>
          <MerchantAuthentication MerchantID="MerchantID" Password="Password" />
          <ThreeDSecureInputData CrossReference="070101010101010101010104">
            <PaRES>
eJycVF1zojAU/SsO+7hjE75EnZiOrdL6wbZFbN3HCBFoIWgCovz6DWots7MPu8sLN4d7zz3nXibo9pAmrT3lIs7YQFFvoN
KizM+CmIUDZenZ7a5yi5EXcUpHC+oXnGLkUCFISFtxMFAMo6tZmgW1noLR89ClAqMLHZZsNxoCn0dZx/2IsBwj4u/uJj+w
qumG2bG6PQQuCEopn4y+PkAEzggCX9XPRR0JqeSQBJiydZpX8f7oH9bb5FhkfMN3KYl4EUQI1BkoIDnFGoQWtDSzBfW+2e
vrFgInHG1rumGaFZJbhVC2bCJIeuZyJEfc1ToIXE+IHrYZozJDWrzGCHyJ2xKGYeMx9dpOjSJvhVEep01RRh+qfVNF4IQj
kZO8EHiIwCVCPtnvcZoeQhHl60212cS7kL/vWZoJdiwLLqS4OgVRP8a1i/p9qhomYcbjPEqxfs75AhCopYDL4sBlsxgt4p
DJtpy25P/BxECJ8nzbB6Asy5tSv8l4CKR0CGAPyIRAxOE35VxFgwnbZBjdE5ax2CdJXJFcrt+heZQFrWvrP1F6bs2qAnd8
35a0bV81WLtGoK6aSgs0dP0N3e8KuSBtERH1xOTSDa03SVtLdzJQpHqPEyY2GU9FI/63DpTtaZJtadAWn0JPzUCTexSHVO
T/Y+Aq/kzxSpKC4tG93uH6Gjw6u1cBO6/p6iF7nezJWh0OEGhmInA1LePmsq5jvVAaSWVAferkVTRNTUPf2lWUv+/4h1fp
jhmVsITcv3cTaL0n88fh+sFerbzxk/Hifzhe96gebd1dmIcP+4ncpeP5kyDaslNses727cFxFnlXM+zvUzHbVy8e8Epn/G
yFwdT+6cYf9nFWTJeJ7Yy6hv3ojO9mKp2PA1A6w2C+upst3cMoGE209ZtdWXM4OBtpiG8A9eCbN9cvAQAAAP//
            </PaRES>
          </ThreeDSecureInputData>
        </ThreeDSecureMessage>
      </ThreeDSecureAuthentication>
    </soap:Body>
</soap:Envelope>
```

Response – ThreeDSecureAuthenticationResponse

```xml
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                        xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <ThreeDSecureAuthenticationResponse xmlns="https://www.thepaymentgateway.net/">
        <ThreeDSecureAuthenticationResult AuthorisationAttempted="True">
            <StatusCode>0</StatusCode>
            <Message>Auth Code:123456</Message>
        </ThreeDSecureAuthenticationResult>
        <TransactionOutputData CrossReference="07010101010010101010102">
            <AuthCode>123456</AuthCode>
            <AddressNumericCheckResult>PASSED</AddressNumericCheckResult>
            <PostCodeCheckResult>PASSED</PostCodeCheckResult>
            <CV2CheckResult>PASSED</CV2CheckResult>
            <ThreeDSecureAuthenticationCheckResult>PASSED</ThreeDSecureAuthenticationCheckResult>
            <CardTypeData>
                <CardType>MAESTRO_INTERNATIONAL</CardType>
                <Issuer>HSBC</Issuer>
            </CardTypeData>
            <AmountReceived>1000</AmountReceived>
            <GatewayEntryPoints>
                <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
                <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
            </GatewayEntryPoints>
        </TransactionOutputData>
            </ThreeDSecureAuthenticationResponse>
    </soap:Body>
  </soap:Envelope>
```

GetCardType

Request – GetCardType

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
    <GetCardType xmlns="https://www.thepaymentgateway.net/">
      <GetCardTypeMessage>
        <MerchantAuthentication MerchantID="MerchantID" Password="Password" />
        <CardNumber>5600000000005390</CardNumber>
      </GetCardTypeMessage>
    </GetCardType>
   </soap:Body>
</soap:Envelope>
```

Response – GetCardTypeResponse

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
    <GetCardTypeResponse xmlns="https://www.thepaymentgateway.net/">
        <GetCardTypeResult>
        <StatusCode>0</StatusCode>
        </GetCardTypeResult>
      <GetCardTypeOutputData>
        <CardTypeData>
          <CardType>MAESTRO_INTERNATIONAL</CardType>
          <Issuer>HSBC</Issuer>
          <LuhnCheckRequired>True</LuhnCheckRequired>
          <IssueNumberStatus>IGNORED_IF_SUBMITTED</IssueNumberStatus>
        </CardTypeData>
        <GatewayEntryPoints>
          <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
          <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
        </GatewayEntryPoints>
      </GetCardTypeOutputData>
      </GetCardTypeResponse>
   </soap:Body>
  </soap:Envelope>
```

GetGatewayEntryPoints

Request – GetGatewayEntryPoints

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                              xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                              xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <GetGatewayEntryPoints xmlns="https://www.thepaymentgateway.net/">
         <GetGatewayEntryPointsMessage>
             <MerchantAuthentication MerchantID="MerchantID" Password="Password" />
         </GetGatewayEntryPointsMessage>
     </GetGatewayEntryPoints>
    </soap:Body>
</soap:Envelope>
```

Response – GetGatewayEntryPointsResponse

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                              xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                              xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
     <GetGatewayEntryPointsResponse xmlns="https://www.thepaymentgateway.net/">
               <GetGatewayEntryPointsResult>
                  <StatusCode>0</StatusCode>
               </GetGatewayEntryPointsResult>
              <GatewayEntryPointsOutputData>
            <GatewayEntryPoints>
                <GatewayEntryPoint EntryPointURL="https://gw1.paymentprocessor.net" Metric="100"/>
                <GatewayEntryPoint EntryPointURL="https://gw2.paymentprocessor.net" Metric="200"/>
            </GatewayEntryPoints>
               </GatewayEntryPointsOutputData>
           </GetGatewayEntryPointsResponse>
    </soap:Body>
  </soap:Envelope>
```

Appendix 4: Override Policy Codes & Explanations

OverrideAVSPolicy Codes

The OverrideAVSPolicy codes are 2 character codes which instruct the gateway how to handle the AVS checking for that particular transaction.

The first character determines the behaviour when 1 or more of the results of the address numeric or post code check are known.

The second and third characters determine the behaviour when dealing with partial matches – this is where either the address numeric check or the post code check returns partial matches.

The forth character determines the behaviour when none of the results of the address numeric or the post code check are known.

Character 1 Codes

Character Code	Explanation
E	This code means fail the transaction if either the address numeric check or post code check has failed
B	This code means fail the transaction only if both the address numeric check and the post code checks have failed
A	This code means fail the transaction only if the address numeric check has failed
P	This code means fail the transaction only if the post code check has failed
N	This code means pass the transaction even if both checks have failed

Character 2 Codes

Character Code	Explanation
P	Treat partial address numeric results as passes
F	Treat partial address numeric results as failures

Character 3 Codes

Character Code	Explanation
P	Treat partial post code results as passes
F	Treat partial post code results as failures

Character 4 Codes

Character Code	Explanation
P	This code means pass the transaction if both results of the AVS check are not known
F	This code means fail the transaction if both results of the AVS check are not known

Examples

EFFF – this is the strongest policy & transactions will only pass if both address numeric & post code checks have passed. Partial matches are treated as failures
EPFP – this policy means that transactions will only pass if both the address numeric & post code checks have passed, but if the results of both are unknown, then pass the transaction. Partial

address numeric results are treated as passes, but partial post code checks are treated as failures
BPPF – this policy means that the transaction will fail only if both the address numeric and post code checks have failed, but if the results of both are unknown, then fail the transaction. Both address numeric and post code partial results are treated as passes
NPPF – this policy means that the transaction will pass even if the results of the address numeric and post code checks are failed, but if the results are unknown, then fail the transaction (not a recommended policy!) . Both address numeric and post code partial results are treated as passes
NPPP – this is the weakest policy & transactions will pass regardless of the results of the address numeric & post code checks. Both address numeric and post code partial results are treated as passes

Questions

Q: Why would the results of the AVS check be unknown?

A: The main reasons for the results of the AVS checks being unknown are:

1) The relevant address data was not passed in with the transaction – the address numeric check is carried out across the Address1, Address1, Address3, Address4, City & State fields – if none of them are present, then the state of the address numeric check will be unknown. Similarly, the post code check is carried out of the field PostCode & if that is not present, then the state of the post code check will be unknown.
2) If the transaction is a cross reference transaction & the respective address information was not submitted with the transaction, or was not submitted or unknown for the transaction being referenced, then the result will carry forward to this transaction
3) If there was a problem contacting the provider, or the provider itself had a problem delivering the results of the AVS checks (least likely reason)

OverrideCV2Policy Codes

The OverrideAVSPolicy codes are 2 character codes which instruct the gateway how to handle the AVS checking for that particular transaction.

The first character determines the behaviour when 1 or more of the results of the address numeric or post code check are known.

The second character determines the behaviour when none of the results of the address numeric or the post code check are known.

Character 1 Codes

Character Code	Explanation
P	This code means pass the transaction if the CV2 check has failed
F	This code means fail the transaction if the CV2 check has failed

Character 2 Codes

Character Code	Explanation
P	This code means pass the transaction if both results of the AVS check are not known
F	This code means fail the transaction if both results of the AVS check are not known

Examples

FF – this is the strongest policy & transactions will only pass if the CV2 check has passed

FP – this policy means that transactions will only pass if the CV2 has passed, but if the results are unknown, then pass the transaction

PF – this policy means that the transaction will pass if the CV2 failed, but if the result of the check is unknown, then fail the transaction (not a recommended policy!)

PP – this is the weakest policy & transactions will pass regardless of the results of the CV2 check

Questions

Q: Why would the CV2 result be unknown?

A: The main reasons for the result being unknown are:

1) The CV2 was not submitted with the transaction
2) If the transaction is a cross reference transaction & the CV2 code was not submitted as an override, or was not submitted or unknown for the original transaction being referenced, then that result will carry forward to this transaction
3) If there was a problem contacting the provider, or the provider itself had a problem delivering the results of the CV2 check (least likely reason)

Appendix 5: List of Card Types

Below is a list of the card types that may be returned by the gateway

Card Type	Full Name
UNKNOWN	Card type not known
VISA	Visa
VISA_DEBIT	Visa Debit (was Delta)
VISA_ELECTRON	Visa Electron
VISA_PURCHASING	Visa Purchasing
MASTERCARD	MasterCard
MASTERCARD_DEBIT	MasterCard Debit
MAESTRO	Maestro
LASER	Laser
ATM	ATM
SOLO	Solo
JCB	JCB
PLATIMA	Platima
AMERICAN_EXPRESS	American Express
DINERS_CLUB	Diner's Club
GE_CAPITAL	GE Capital

Appendix 6: The 3D Secure System

The 3D Secure System Explained

The 3D Secure system is a scheme implemented by the card schemes (primarily Visa, who call it Verified By Visa or VbV and MasterCard, who call it MasterCard SecureCode).

The basic concept of the system is to tie the financial authorisation process with an online authentication. This authentication is based on a 3 domain model (that is the 3D in the name). The three domains are: Acquirer Domain (the commerce), the Issuer Domain (the bank issuer of the credit card) and finally the Interoperability Domain (Worldwide credit card and support).

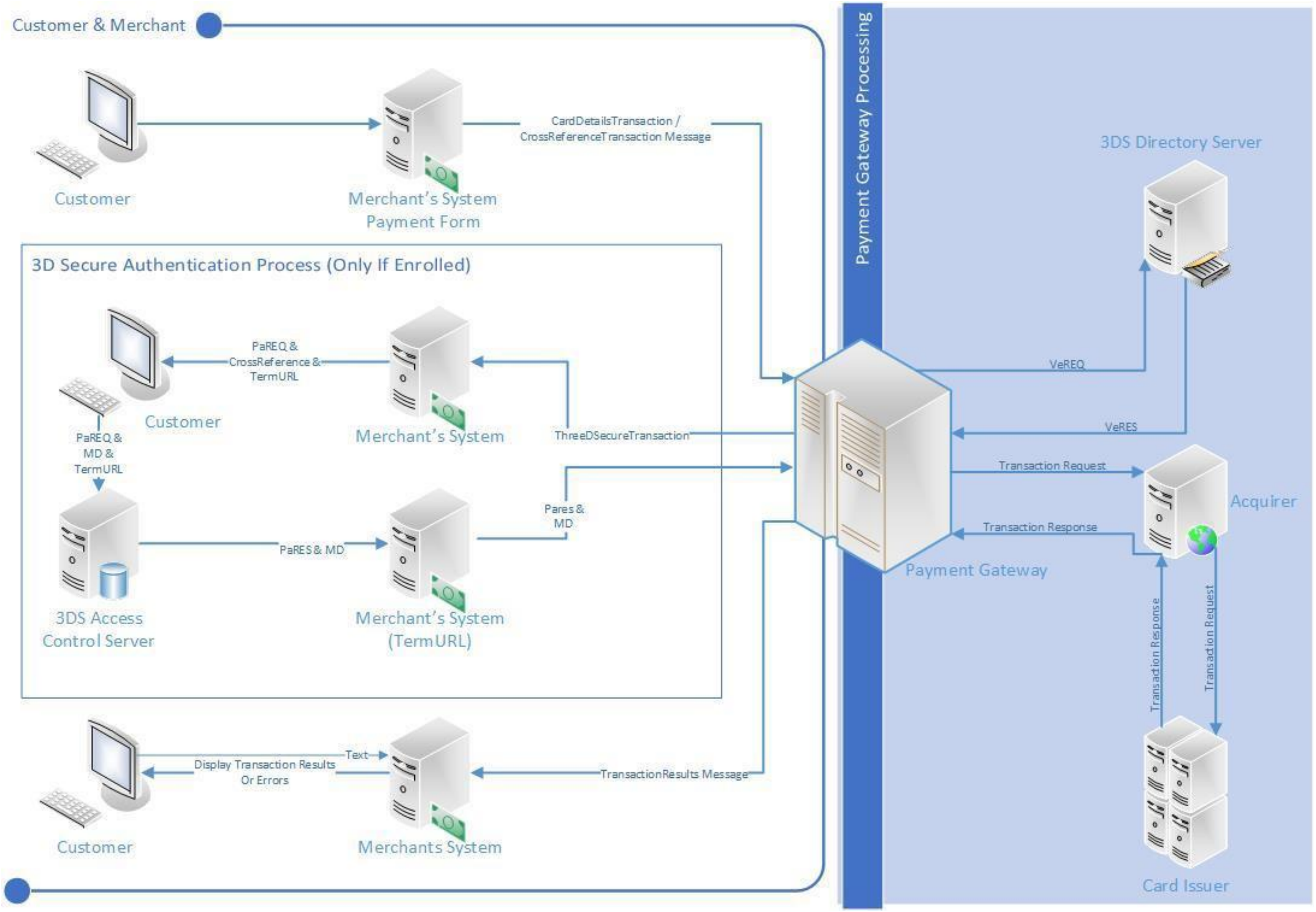
The transaction is effectively broken into 2 messages. During the initial message the card number is checked to see if it enrolled on the card-issuing organisation's (usually a bank) 3D Secure scheme. If the card is enrolled on the scheme, the transaction is "paused" & the message ends, informing the merchant's website (and so the customer) that they must authenticate their card. This happens by the customer being redirected to their card-issuing organisation's website & validating their card directly with them. They are the redirected back to the merchant's website which passes the authentication payment response generated by the card-issuer's website during the authentication process back to the payment gateway with the second message of the transaction. The gateway then verifies the authentication payment response with the card-issuer directly & depending on the results of this the transaction is resumed or rejected.

Listed below are the steps that a 3D Secure transaction takes and a diagram below:

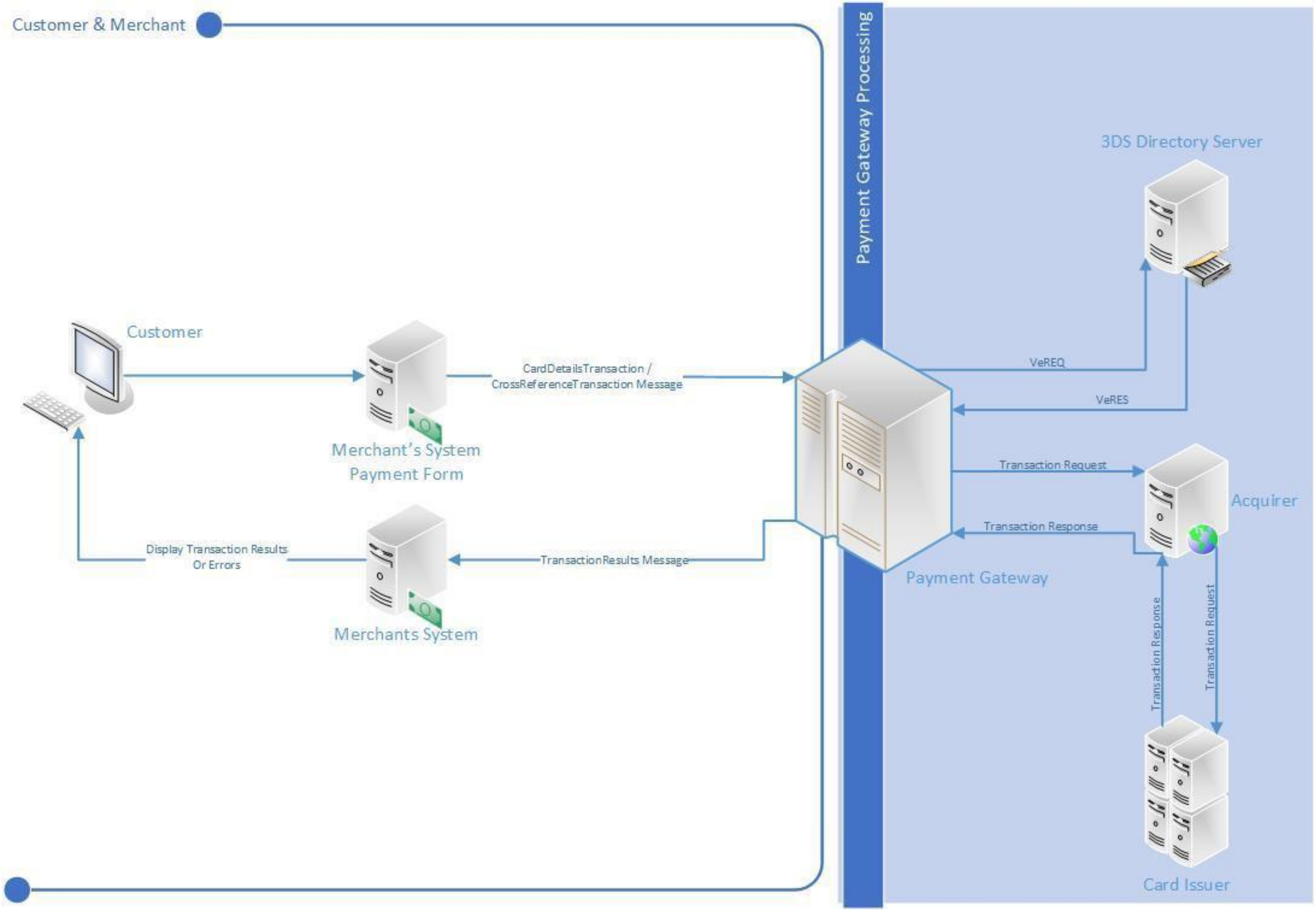
1) The cardholder navigates to the merchant's website, & fills in their credit card details into the merchant's payment form (this form may reside on the merchant's servers or on the payment processing servers).
2) The credit card information is submitted to the payment gateway by the merchant's payment form (using a CardDetailsTransaction message).
3) The payment gateway contacts the Directory Server to query whether this credit card is enrolled (or needs to be enrolled) in the 3D Secure scheme.
4) The Directory Server passes the enrolment status information back to the payment gateway, which in turn either continues processing the transaction as normal (if the card is not enrolled), or it passes the URL of the cardholder's bank's Access Control Server (ACSURL) and additional data from which a Payment Request string (PaREQ) back to the merchant's payment form. This will be done using the CardDetailsTransactionResponse message.
5) The customer is then redirected by the payment form to their bank's Access Control Server & they are greeted with the last 4 digits of their credit card & the identification text they specified when registering their card for 3D Secure. The customer validates their card details using their 3D Secure password, which is validated by their bank's Access Control Server
6) The Access Control Server then initiates a redirect of the customer's browser back to a secure processing page on the merchant's website (TermUrl), which forwards the payment response string (PaRES) from the Access Control Server to the payment gateway using a ThreeDSecureAuthentication message.

7) Depending on the contents of the payment response (PaRES), the transaction is either declined immediately (following a 3D Secure Authentication failure) or the transaction is then submitted to the bank for authorisation. The results of the transaction are then passed back to the merchant’s system using a ThreeDSecureAuthenticationResponse which displays the payment result to the customer.

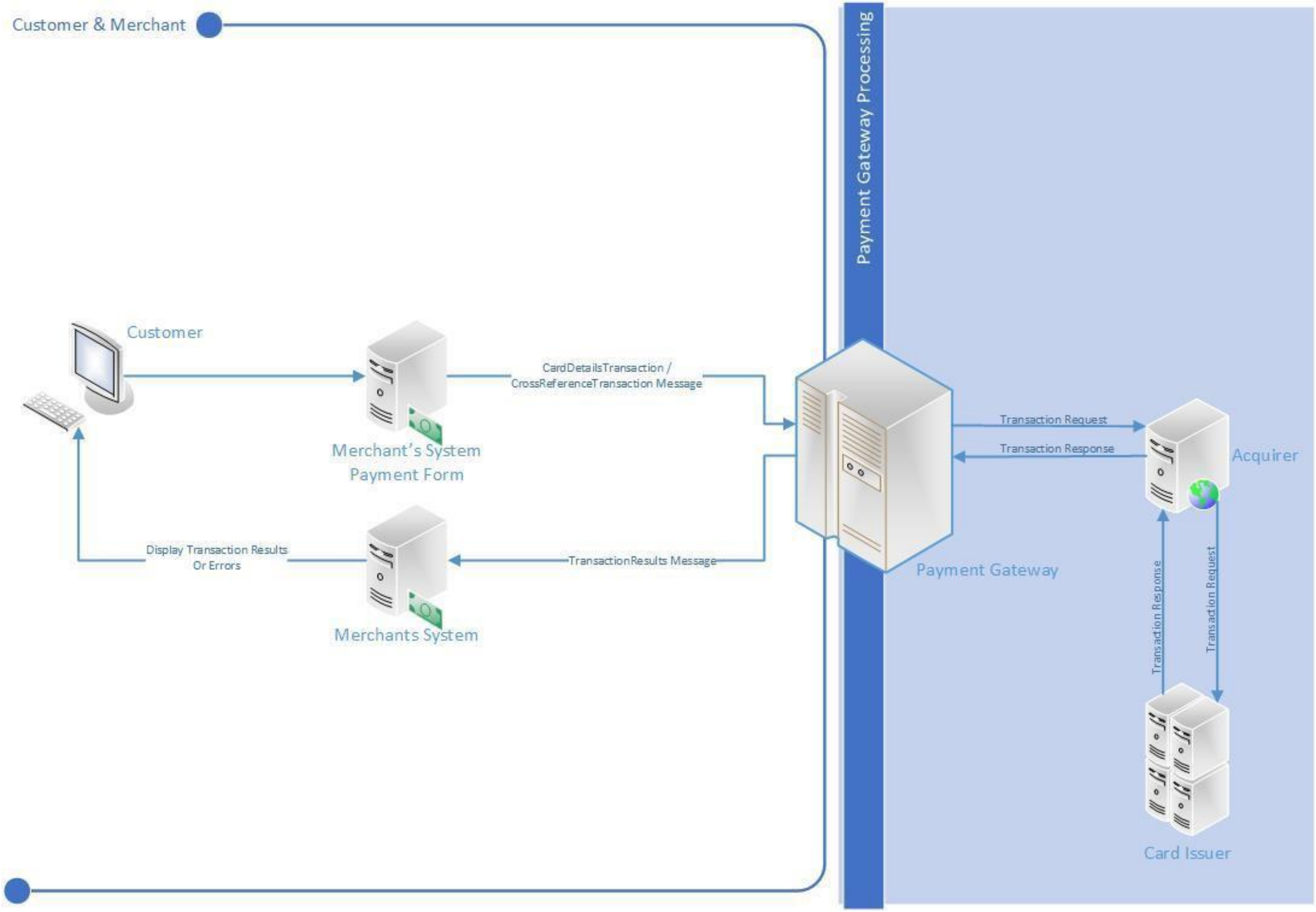
API Transaction Flow – Including 3D Secure Authentication



API Transaction Flow – 3D Secure Disabled or Card Not Enrolled

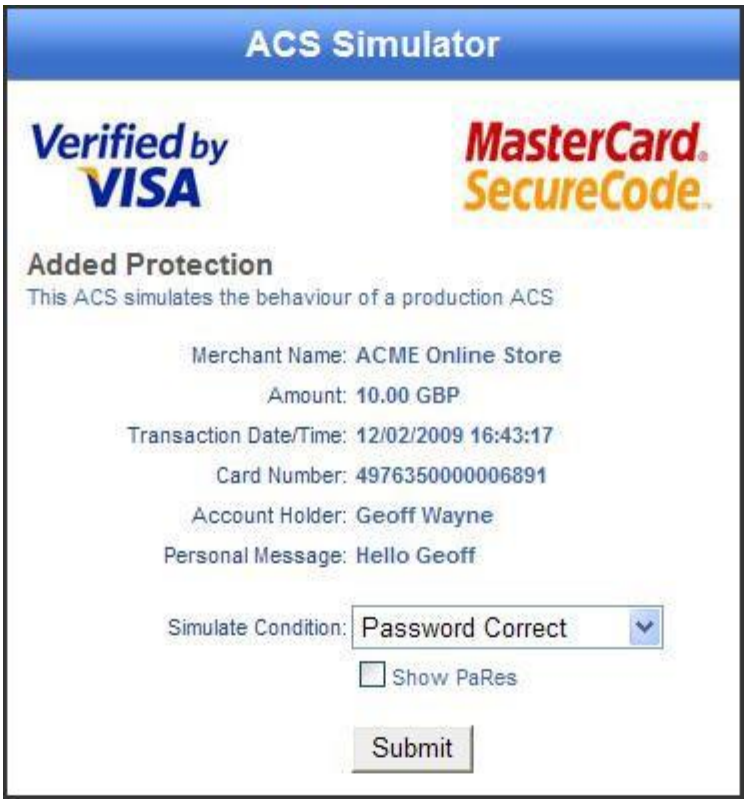


API Transaction Flow – 3D Secure Disabled



ACS Simulator

The test system comes complete with an ACS simulator, which allows your developer to simulate the most common responses that might come back from the cardholder's bank's access control server.



Added Protection

This ACS simulates the behaviour of a production ACS

Merchant Name: ACME Online Store

Amount: 10.00 GBP

Transaction Date/Time: 12/02/2009 16:43:17

Card Number: 4976350000006891

Account Holder: Geoff Wayne

Personal Message: Hello Geoff

Simulate Condition: Password Correct

□ Show PaRes

There are 4 possible conditions that can be simulated:

1) Password Correct – the case where the cardholder enters the correct 3D Secure password. Relates to a 3D Secure status of "**Y**"
2) Password Incorrect – the case where the cardholder enters the wrong 3D Secure password. Relates to a 3D Secure status of "**N**"
3) Attempted Processing – the case where the cardholder attempted to authenticate themselves, but this could not be completed for some reason. Proof of this attempt is returned with the payment response message. Relates to a 3D Secure status of "**A**"
4) Unknown Error – the case where an unexpected error occurred whilst trying to authenticate the cardholder. Relates to a 3D Secure status of "**U**"

Appendix 7: Country (ISO 3166-1) Codes

ISO Code	Country
826	United Kingdom
840	United States
036	Australia
004	Afghanistan
248	Åland Islands
008	Albania
012	Algeria
016	American Samoa
020	Andorra
024	Angola
660	Anguilla
010	Antarctica
028	Antigua and Barbuda
032	Argentina
051	Armenia
533	Aruba
040	Austria
031	Azerbaijan
044	Bahamas
048	Bahrain
050	Bangladesh
052	Barbados
112	Belarus
056	Belgium
084	Belize
204	Benin
060	Bermuda
064	Bhutan
068	Bolivia
070	Bosnia and Herzegovina
072	Botswana
074	Bouvet Island
076	Brazil
086	British Indian Ocean Territory
096	Brunei Darussalam
100	Bulgaria
854	Burkina Faso
108	Burundi
116	Cambodia
120	Cameroon

124	Canada
132	Cape Verde
136	Cayman Islands
140	Central African Republic
148	Chad
152	Chile
156	China
162	Christmas Island
166	Cocos (Keeling) Islands
170	Colombia
174	Comoros
178	Congo
180	Congo, Democratic Republic of the
184	Cook Islands
188	Costa Rica
384	Côte d'Ivoire
191	Croatia
192	Cuba
196	Cyprus
203	Czech Republic
208	Denmark
262	Djibouti
212	Dominica
214	Dominican Republic
218	Ecuador
818	Egypt
222	El Salvador
226	Equatorial Guinea
232	Eritrea
233	Estonia
231	Ethiopia
238	Falkland Islands (Malvinas)
234	Faroe Islands
242	Fiji
246	Finland
250	France
254	French Guiana
258	French Polynesia
260	French Southern Territories
266	Gabon
270	Gambia
268	Georgia
276	Germany
288	Ghana

292	Gibraltar
300	Greece
304	Greenland
308	Grenada
312	Guadeloupe
316	Guam
320	Guatemala
831	Guernsey
324	Guinea
624	Guinea-Bissau
328	Guyana
332	Haiti
334	Heard Island and McDonald Islands
336	Holy See (Vatican City State)
340	Honduras
344	Hong Kong
348	Hungary
352	Iceland
356	India
360	Indonesia
364	Iran, Islamic Republic of
368	Iraq
372	Ireland
833	Isle of Man
376	Israel
380	Italy
388	Jamaica
392	Japan
832	Jersey
400	Jordan
398	Kazakhstan
404	Kenya
296	Kiribati
408	Korea, Democratic People's Republic of
410	Korea, Republic of
414	Kuwait
417	Kyrgyzstan
418	Lao People's Democratic Republic
428	Latvia
422	Lebanon
426	Lesotho
430	Liberia
434	Libyan Arab Jamahiriya
438	Liechtenstein

440	Lithuania
442	Luxembourg
446	Macao
807	Macedonia, the former Yugoslav Republic of
450	Madagascar
454	Malawi
458	Malaysia
462	Maldives
466	Mali
470	Malta
584	Marshall Islands
474	Martinique
478	Mauritania
480	Mauritius
175	Mayotte
484	Mexico
583	Micronesia, Federated States of
498	Moldova
492	Monaco
496	Mongolia
499	Montenegro
500	Montserrat
504	Morocco
508	Mozambique
104	Myanmar
516	Namibia
520	Nauru
524	Nepal
528	Netherlands
530	Netherlands Antilles
540	New Caledonia
554	New Zealand
558	Nicaragua
562	Niger
566	Nigeria
570	Niue
574	Norfolk Island
580	Northern Mariana Islands
578	Norway
512	Oman
586	Pakistan
585	Palau
275	Palestinian Territory, Occupied
591	Panama

598	Papua New Guinea
600	Paraguay
604	Peru
608	Philippines
612	Pitcairn
616	Poland
620	Portugal
630	Puerto Rico
634	Qatar
638	Reunion Réunion
642	Romania
643	Russian Federation
646	Rwanda
652	Saint Barthélemy
654	Saint Helena
659	Saint Kitts and Nevis
662	Saint Lucia
663	Saint Martin (French part)
666	Saint Pierre and Miquelon
670	Saint Vincent and the Grenadines
882	Samoa
674	San Marino
678	Sao Tome and Principe
682	Saudi Arabia
686	Senegal
688	Serbia
690	Seychelles
694	Sierra Leone
702	Singapore
703	Slovakia
705	Slovenia
90	Solomon Islands
706	Somalia
710	South Africa
239	South Georgia and the South Sandwich Islands
724	Spain
144	Sri Lanka
736	Sudan
740	Suriname
744	Svalbard and Jan Mayen
748	Swaziland
752	Sweden
756	Switzerland
760	Syrian Arab Republic

158	Taiwan, Province of China
762	Tajikistan
834	Tanzania, United Republic of
764	Thailand
626	Timor-Leste
768	Togo
772	Tokelau
776	Tonga
780	Trinidad and Tobago
788	Tunisia
792	Turkey
795	Turkmenistan
796	Turks and Caicos Islands
798	Tuvalu
800	Uganda
804	Ukraine
784	United Arab Emirates
581	United States Minor Outlying Islands
858	Uruguay
860	Uzbekistan
548	Vanuatu
862	Venezuela
704	Viet Nam
92	Virgin Islands, British
850	Virgin Islands, U.S.
876	Wallis and Futuna
732	Western Sahara
887	Yemen
894	Zambia
716	Zimbabwe

Appendix 8: Currency (ISO 4217) Codes

ISO Code	Currency
826	Pound Sterling
840	US Dollar
978	Euro
971	Afghani
12	Algerian Dinar
32	Argentine Peso
51	Armenian Dram
533	Aruban Guilder
36	Australian Dollar
944	Azerbaijanian Manat
44	Bahamian Dollar
48	Bahraini Dinar
764	Baht
590	Balboa
50	Bangladeshi Taka
52	Barbados Dollar
974	Belarusian Ruble
84	Belize Dollar
60	Bermudian Dollar
984	Bolivian Mvdol (Funds code)
68	Boliviano
986	Brazilian Real
96	Brunei Dollar
975	Bulgarian Lev
108	Burundian Franc
124	Canadian Dollar
132	Cape Verde Escudo
136	Cayman Islands Dollar
288	Cedi
952	CFA Franc BCEAO
950	CFA Franc BEAC
953	CFP franc
152	Chilean Peso
963	Code reserved for testing purposes
170	Colombian Peso
174	Comoro Franc
977	Convertible Marks
558	Cordoba Oro
188	Costa Rican Colon
191	Croatian Kuna
192	Cuban Peso

196	Cyprus Pound
203	Czech Koruna
270	Dalasi
208	Danish Krone
807	Denar
262	Djibouti Franc
678	Dobra
214	Dominican Peso
951	East Caribbean Dollar
818	Egyptian Pound
230	Ethiopian Birr
955	European Composite Unit (EURCO)
956	European Monetary Unit
958	European Unit of Account 17 (E.U.A.-17)
957	European Unit of Account 9 (E.U.A.-9)
238	Falkland Islands Pound
242	Fiji Dollar
348	Forint
976	Franc Congolais
292	Gibraltar pound
959	Gold (one Troy ounce)
600	Guarani
324	Guinea Franc
328	Guyana Dollar
332	Haiti Gourde
344	Hong Kong Dollar
980	Hryvnia
352	Iceland Krona
356	Indian Rupee
364	Iranian Rial
368	Iraqi Dinar
388	Jamaican Dollar
392	Japanese yen
400	Jordanian Dinar
404	Kenyan Shilling
598	Kina
418	Kip
233	Kroon
414	Kuwaiti Dinar
894	Kwacha
454	Kwacha
973	Kwanza
104	Kyat
981	Lari

428	Latvian Lats
422	Lebanese Pound
8	Lek
340	Lempira
694	Leone
430	Liberian Dollar
434	Libyan Dinar
748	Lilangeni
440	Lithuanian Litas
426	Loti
969	Malagasy Ariary
458	Malaysian Ringgit
470	Maltese Lira
795	Manat
480	Mauritius Rupee
943	Metical
484	Mexican Peso
979	Mexican Unidad de Inversion (UDI)
498	Moldovan Leu
504	Moroccan Dirham
566	Naira
232	Nakfa
516	Namibian Dollar
524	Nepalese Rupee
532	Netherlands Antillian Guilder
376	New Israeli Shekel
901	New Taiwan Dollar
949	New Turkish Lira
554	New Zealand Dollar
64	Ngultrum
999	No currency
408	North Korean Won
578	Norwegian Krone
604	Nuevo Sol
478	Ouguiya
776	Pa'anga
586	Pakistan Rupee
964	Palladium (one Troy ounce)
446	Pataca
858	Peso Uruguayo
608	Philippine Peso
962	Platinum (one Troy ounce)
72	Pula
634	Qatari Rial

320	Quetzal
512	Rial Omani
116	Riel
642	Romanian Leu
946	Romanian New Leu
462	Rufiyaa
360	Rupiah
643	Russian Ruble
646	Rwanda Franc
654	Saint Helena Pound
882	Samoan Tala
682	Saudi Riyal
941	Serbian Dinar
690	Seychelles Rupee
961	Silver (one Troy ounce)
702	Singapore Dollar
703	Slovak Koruna
90	Solomon Islands Dollar
417	Som
706	Somali Shilling
972	Somoni
710	South African Rand
410	South Korean Won
960	Special Drawing Rights
144	Sri Lanka Rupee
938	Sudanese Pound
968	Surinam Dollar
752	Swedish Krona
756	Swiss Franc
760	Syrian Pound
834	Tanzanian Shilling
398	Tenge
780	Trinidad and Tobago Dollar
496	Tugrik
788	Tunisian Dinar
800	Uganda Shilling
970	Unidad de Valor Real
990	Unidades de formento
784	United Arab Emirates dirham
860	Uzbekistan Som
548	Vatu
862	Venezuelan bolívar
704	Vietnamese đồng
947	WIR Euro

948	WIR Franc
886	Yemeni Rial
156	Yuan Renminbi
716	Zimbabwe Dollar
985	Zloty
997	No currency
998	No currency