Universal Protocol

API Reference

Version: 1.22

Version Date: 2019-09-06

THIS IS NOT A SALES DOCUMENT

Document Control

Version	Prepared By:	Comments	Date
1.0	Paul Kettley	Initial Draft	2017-02-01
1.1	Paul Kettley	Added `BUNDLE` element to `PINCREDENTIALS` Added error codes for couponing Added `DISCOUNT` object to individual `SKU`	2017-02-27
1.2	Paul Kettley	Added `MAKE`, `COLOUR`, `MODEL`, `CLASSID`, `CLASS` to `VEHICLE` Added `TRIPTIME` to `TRIP` Added `MANUFACTURERCODE` to `SKU` Added `TYPE` to `SKU` Added `EXCLUDED` to `OFFER` Added `MERCHANTCATEGORY` to `OFFER`	2017-03-13
1.3	Paul Kettley	Added `SETVALUE` to `TYPE` in `DISCOUNT` Added `SETVALUE` to `DISCOUNT` Added `CHECKTYPE` to `OFFER` Added `SKUGROUPS` to `OFFER` Added `SKUGROUPS` object section	2017-03-29
1.4	Paul Kettley	Added `SCODE` to `PINCREDENTIALS`	2017-04-13
1.5	Paul Kettley	Added `DENOMINATIONS` to `ROAD` Added `MAXTRIPS` to `TOLLING` Added `TRIPDATEALLOWED` to `TOLLING` Added `SERVICEFEE` to `RESPONSE`	2017-04-25
1.6	Paul Kettley	Added `VALIDPERIOD` Added `VALIDPERIOD` array to `COUPON.VALIDITY` Added `FAREPRODUCT` Added `PASS`	2017-05-18
1.7	Paul Kettley	Added Wallet (Pays) Service section Added `STATE` Added `PARCEL`	2017-10-10
1.8	Fallon Harvey	Added Wallet (Pays) XML examples	2018-01-26
1.9	Fallon Harvey	Updated minor formatting errors Added `DIGITALLOCKERID` to `CUSTOMER`	2018-02-05

1.10	Paul Kettley	Added Parcel Service Examples Added `PLATFORM` element to `REQUEST` Added `TRACEID` element to `RESPONSE` Added `AID` element to `RESPONSE`	2018-03-21
1.11	Paul Kettley	Added `PIN` to `CARD` which is required for some `REDEEM` and `STATUS` request types. Added `COMMENT` element to `REQUEST` for optional transaction narrative	2018-04-20
1.12	Paul Kettley	Added `PHONE` element to `REQUEST` for Direct Recharge transactions. Added Store and Forward comment to `RESULTTEXT` element in `RESPONSE` Added `COUNTRY` element to `REQUEST`	2018-04-20
1.13	Paul Kettley	Added `BARCODE` to `ROAD` object	2018-04-24
1.14	Paul Kettley	Added `TITLE` to `PRODUCTINFO` used for supplying product title within the HOST Added `GIFTING` to `RESPONSE` used for customer to gift digital products to another person	2018-05-24
1.15	Paul Kettley	Added `PIN` to `COUPON` redemption Added `PINREQUIRED` to `VALIDITY` Added RESULTTEXT to result text code for PIN requirement in COUPON redemption	2018-08-29
1.16	Paul Kettley	Added QR Code Scaling to section 1.8.6 Added New Invert control character to turn background to black and text to white	2018-10-05
1.17	Paul Kettley	Added Graphics Line Weighting section **1.8.4** Corrected Version number on front page	2018-10-15
1.18	Paul Kettley	Added details regarding `REFUND:DIRECT` for `PAYS`	2018-12-05
1.19	Paul Kettley	Added additional result text for coupon basket validation	2019-01-29
1.20	Paul Kettley	Added `ACTIVATIONDATE` field to response for distribution, this field is used when a product will be activated on a specific date. `AMOUNT` field has become mandatory for `REFUND` and `REDEEM:RESERVE` for PAYS `CARD` field has become mandatory when PAYS phone transaction is being initiated. `CARD` field has become not mandatory when PAYS transaction is being initiated using QR Code.	2019-02-04
1.21	Paul Kettley	Added Operator `PROMPT` Result Codes Added `PROMPT` object	2019-03-22
1.22	Paul Kettley	`CANCEL:AUTOMATIC` – Change to CLIENT Added `SERIAL` to `CARD` Removed SOAP usage statement	2019-09-06

Contents

References

Document	Description

Main Terminology

Term/Abbreviation	Definition
Integrator	An integrator can be classified as a EPOS, POS, online retailer, device or switch. This is the system that will integrate with the API
Merchant Account	A merchant account represents an account on the transaction platform that all requests are billed again. The merchant account is identified using the `TERMINALID` and the `AUTHORIZATION` details
POX	POX stands for Plain Old XML
Json	JavaScript object notation
SOAP	Simple Object Access Protocol
API	Application Programming Interface
User	Represents a user of the integrator system, also known as a cashier, customer service representative

1. Introduction

1.1. API Reference

The Universal Protocol API supports several different protocols including REST via Json, POX.

epay' s API has predictable, resource-orientated URLS, and uses HTTP response codes as well as our own API errors. epay use built-in HTTP features like HTTP verbs, which are understood by off-the-shelf HTTP clients.

To make the API as explorable as possible, accounts are setup for test mode and live mode, and epay have separated environments to switch between these modes, requests made to the test environment will never incur any costs.

1.2. Audience

This document is intended for developers who are needing to integrate the API into their system, this document is not intended to be used as a commercial sales document.

1.2.1. API Endpoint

The API uses the HTTPS communication method, to adhere to PCI compliance epay only support standard ciphers. The API supports multiple end points for different countries and requires integrators to support automated failover across URLs and for load balancing.

1.2.2. Certificates

The API employs wildcard certificates for disaster recovery purposes, which means epay can maintain certificates around without any problems. Wildcard certificates are valid through s-tunnel and browsers, however ***WinHttp*** does not support the use of wildcard certificates, so error checking on invalid common names should be disabled in the code.

1.3. Authentication

Authenticate your account when using the API by including your username/password in the request. Your username/password carry many privileges, so be sure to keep them secret. Do not share your credentials in publicly accessible areas such as GitHub or client-side code.

Authentication to the API is performed via each request using the authentication element of the API.

All API requests must be made over HTTPS. Calls made over plain HTTP will fail within the live environment. API requests without authentication will also fail.

1.4. Errors

The Universal Protocol API uses conventional HTTP response codes to indicate the success or failure of an API request.

Not all errors map cleanly onto HTTP response codes, however when a request is valid but does not complete successfully (e.g., card declined), epay will return an API error code. To understand all the different types of error conditions, refer to the list of codes in the appendix.

It is **mandatory** that a integrator performs automatic cancellations in the event of any HTTP error which results in no response being returned.

1.4.1. Handling Errors

epay' s API raise exceptions for different reasons, such as invalid cards, invalid parameters, invalid values and authentication errors or network unavailability. epay recommend writing code that gracefully handles all possible API exceptions. Please refer to the list of codes in the documentation which will provide recommendations on when to perform a system cancellation based on the result of the previous request.

1.5. Idempotent Requests

The API does **not** support idempotency for retrying requests, which means that re-attempting a request can result in the same operation being attempted twice. In the event of a failure epay recommend making sure that the integrator have cancelled the previous request.

1.6. Request IDs

Each API request must have a unique identifier (even for cancellations). The integrator will use in each request the `TXID` field to be populated with an identifier.

All responses will contain an appropriate API response identifier found within the `HOSTTXID` field.

1.7. Date & Time Fields

All date time fields are required to be represented as date/times without UTC and follow the ISO8601 format, for more information please see http://en.wikipedia.org/wiki/ISO_8601

1.8. Receipting

If the integrator has opted to support the receipt text returned by the API epay recommend implementing the following control characters to tell the printer to alter the font, sizes and emphasis for each line. Please note that some products require the integrator to support the receipt text returned and therefore epay recommend implementing the controls.

1.8.1. Text Control

Control Codes	Description
<!TL!>	Left justify text
<!TR!>	Right justify text
<!TC!>	Centre text
<!TS!>	Split text where <!|!> is shown *
<!|!>	Split text separator *
<!INV!>	Invert Background and font colour Background becomes black and font colour becomes white

* Split Text is where a text string is to be printed on 1 line but some text is left justified and some is right justified. At the start of the line there is the Control Code <!TS!> (and there may be other Control Codes too). Part way through the text there is the <!|!> split separator indicating the end of the left justified text and the start of the right justified text. The line is terminated with LF in the usual way.

1.8.2. Characters per line

Control Codes	Description
<!C42!>	42 Characters per line
<!C32!>	32 Characters per line
<!C24!>	24 Characters per line

1.8.3. Font Size

Control Codes	Description
<!FDH!>	Double Height
<!FDW!>	Double Width
<!FSH!>	Single Height
<!FSW!>	Single Width

1.8.4. Graphic

Control Codes	Description
<!G!> **<!G{N}!>**	Print a solid line across the width of the receipt default is 1 Added a number between 1-255 after the G will result in creating a thicker line.
<!L:filename!>	Print a logo from a file held on the integrator's system. These files can be provided to integrators as part of initial product setup. epay recognise that some integrators cannot print logos.

1.8.5. Barcode

The receipt text can also contain barcode where printing is controlled by the following control codes:

	Value	Description
1	<!B	Identifies that a barcode is to be printed
2	M or P	Type of barcode to be printed M = Mandatory Barcode P = Product code
3	L or S	Height of barcode L = Large S = Small
4	13 or 128	Barcode type 13 = EAN 13 128 = Code 128
5	:	Delimiter
6	Barcode	Numeric barcode value
7	!>	Identifies end of the barcode control

Example of a small mandatory code 128 barcode:

<!BMS128:12121226789456123547895 4!>

1.8.6. QR Code

The receipt text can also contain QR codes where printing is controlled by the following control codes:

	Value	Description
1	<!B	Identifies that a barcode is to be printed
2	M or P	Type of barcode to be printed M = Mandatory Barcode P = Product code
3	L or S	Size of QR Code L = Large S = Small
4	QR	QR Code Identifier
5	Error correction	Used for setting the amount of error correction 00 = Lowest .. 03 = Highest
6	QR Version	[Deprecated] Default value set to 99
7	Print Scaling	[Deprecated] **04** - Smallest 05 - Small **06** - Medium 07 – Medium (Little Larger then 6) **08** - Large 99 – Legacy Value (default to Large)
8	Text	Text to be used to generated the QR code
9	!>	Identifies end of the barcode control

Example of a QR-Code markup code:

<!BMSQR019999:https://www.google.co.uk/?#q=epay!>

2. Attributes

The following list of attributes are used within one or more of the service methods and are marked as mandatory where necessary for all requests. Where attributes that are marked as conditional please refer to the service sections.

Name	Request / Response	Mandatory Conditional Optional	Data Type	Description
TYPE	Request Response	Mandatory	String	The type of method being attempted, i.e. SALE, REDEEM Please refer to **section 4**
MODE	Request Response	Mandatory	String	The mode method being attempted, i.e. DIRECT, CHECK Please refer to **section 4**
TERMINALID	Request Response	Mandatory	String	The terminal ID the transaction will occur against
SHOPID	Request	Mandatory	String	The Store ID the transaction will occur against
TXID	Request	Mandatory	String	Integrator's transaction reference from system. Used for reconciliation to cross reference transactions and to match up Refunds and Reversals. Typically, this identifier may be a combination of store number, till number, transaction number and item number. Therefore, each transaction shall be individually identified. This field should be the integrator's unique identifier for the transaction. Uniqueness will not be validated at epay but it must be "unique enough" to identify the original transaction for CAPTURE and REPRINT modes, where it can be entered in the TXNREF element
LOCALDATETIME	Request	Mandatory	Date Time	The local date and time from the integrator's system
PRODUCTID	Request	Mandatory in all requests apart from `CANCEL`	String	The product identifier an integrator is requesting. Can be a unique API product ID or EAN
CASHIER	Request	Optional	String	The user ID or name for who made the request from the integrator's system
MSGNO	Request	Optional	Unsigned Integer	Incremented number for every message (unique within scope of the retail group)

SEQUENCE	Request	Optional	Unsigned Integer	Incremented number for all successful messages (successfully transported) – used to identify missed responses (unique within scope of the retail group)
TILLID	Request	Optional	String	ID for the till conducting the transaction, integrator supplied, unique within the `SHOPID`
RETAILERID	Request	Optional	String	API supplied ID for the retail group

TXREF TYPE="type"	Request	Conditional	String	Attribute "TYPE" values:
				`HOST`
				`CLIENT`
				
				This element should contain the transaction identifier of the original transaction. This can be either the epay generated `HOSTTXID` from the original `RESPONSE` message or the integrator generated `TXID` of the original message
				
				Example:
				
				`<TXREF TYPE="CLIENT">0007743</TXREF>`

AMOUNT	Request Response	Conditional	Signed Integer	Amount of purchase price, redemption value or pay-out. Amount is represented in minor currency units
CURRENCY	Request	Conditional	String	ISO Character or code for currency
RESULT	Response	Mandatory	Unsigned Integer	Numeric code for error conditions. 0 for success. Result codes are defined in document in the appendix
RESULTTEXT	Response	Mandatory	String	Short textual description of the result. This is not intended to be printed or displayed to the user or customer. If a content provider's system is not available and epay are able to switch to "Store and Forward" the `RESULTTEXT` value will start with `$SAF$`
ECHO	Request Response	Optional	String	Used to pair request and response on a-sync interfaces. It will be echoed unmodified in the response message. This is not recorded on the epay Host
SERVERDATETIME	Response	Mandatory	Date Time	API Host date and timestamp in UTC
HOSTTXID	Response	Mandatory	String	API Supplied Unique Host transaction ID
STATUS	Response	Conditional	String	Status of a card
SERVICEFEE	Response	Conditional	Signed Integer	Amount of Service Fee to be applied to total value. Amount is represented in minor currency units.

				Used for services such as invoice payment.
PLATFORM	Request	Optional	String	Used for inter-communication between processing centres to state the platform that holds the entity of the merchant
TRACEID	Response	Optional	String	Used for inter-communication between processing centres, a responding processing centre can return their own trace identifier.
AID	Response	Optional	String	Used to return the service provider unique transaction identifier.
COMMENT	Request	Optional	String	Transaction Narrative
PHONE	Request	Optional	String	Mobile / Phone Number which should be used for direct recharge to a mobile number, also known as mobile top-up.
CARDSTATUS	Response	Optional	String	Available to some card profiles. The status of the card: • `CREATED`: *Card is available in System and waiting for activation.* • `NOTACTIVATED`: *Card not activated and cannot be used.* • `ACTIVATED`: *Card activated and can be used.* • `DEACTIVATED`: *Card deactivated and cannot be used or activated.* • `FREETOUSE`: *Card ready to use. No activation necessary.*
COUNTRY	Request	Optional	String	ISO 3166 standard country code. Alpha-2 or Alpha-3 supported
ACTIVATIONDATE	Response	Optional	Date Time	Activation date of product if to be performed later.

3. Objects

The following list of objects are used within one or more of the service methods and are marked as mandatory where necessary for all requests, where attributes are marked as conditional please refer to the service sections.

Name	Request/Response	Mandatory Conditional Optional	Description
AUTHORIZATION	Request	Mandatory	The authorization details for merchant account to be authenticated against
BASKET	Request	Conditional	The basket information at the time of request, this object can be supplied on all requests
ADVERTISEMENT	Response	Conditional	The advertisement object is supplied automatically by the API if an advertisement has been setup for the account or has been requested through the API
CARD	Request	Conditional	Used for supplying card details, activation barcodes or account references
PINCREDENTIALS	Response	Conditional	Used for supplying the PIN information unique for the request
RECEIPT	Request Response	Optional	Used for requesting receipt text from the API and for the API to return with Customer or Merchant receipts
BALANCE	Response	Conditional	Used for providing the Balance of an account or gift card
PAYMENTINFORMATION	Request	Conditional	Used to supply how a customer has paid for a product or service.
DEVICE	Request	Optional	Used for supplying details of the device which performed the request
PRODUCTINFO	Response	Conditional	Used for supplying product details
COUPON	Request Response	Conditional	Used for supplying details of an individual coupon
VALIDITY	Response	Conditional	Used for supplying details of validity regarding a coupon
OFFER	Response	Conditional	Used for supplying details of an offer regarding a coupon
DISCOUNT	Response	Conditional	Used for supplying discount details regarding a coupon
SKU	Request Response	Conditional	Used for supplying a SKU list in a request or response
LOTTERY	Request	Conditional	Used for supplying lottery selections by a customer
TOLLING	Request Response	Conditional	Used for supplying details during a tolling product request
VEHICLE	Request	Conditional	Used for supplying details of a vehicle

	Response		

Name	Request/Response	Mandatory Conditional Optional	Description
ROAD	Response	Conditional	Used for supplying details of a road
TRIP	Request Response	Conditional	Used for supplying details of past trips or future trip purchases
CUSTOMER	Request Response	Conditional	Used for supplying details of a customer
ADDRESS	Request Response	Conditional	Used for detailing an address
CUSTOMDATA	Request Response	Conditional	Used for storing additional meta data
VALIDPERIOD	Response	Conditional	Used to represent a time frame either based on a full date and time, date or time.
FAREPRODUCT	Request Response	Conditional	Used to represent a product for ticketing
PASS	Request Response	Conditional	Used to represent a product for ticketing
PARCEL	Request Response	Conditional	Used to represent a parcel for collection or delivery.
GIFTING	Response	Conditional	Used to represent details used to allow a customer to gift their purchase to someone else
PROMPTS	Response	Conditional	An array of `PROMPT` Used to provide integrator ability to provide further prompts on their device.

3.1. Authorization

This is an object representing the merchant account API credentials. The integrator must supply this object in **every** transactional request, it is not required for a diagnostic test.

The Authorization Object

Name	Required	Data Type	Description
USERNAME	Yes	String	API username
PASSWORD	Yes	String	API password

POX

```
<AUTHORIZATION>
       <USERNAME></USERNAME>
       </PASSWORD></PASSWORD>
</AUTHORIZATION>
```

Json

```
"AUTHORIZATION": {
        "PASSWORD": "",
        "USERNAME": ""
    }
```

3.2. Basket

This is an object representing a basket. The integrator does not have to supply this object in every transactional request, however epay recommend that at least the basket `ID` and `TOTAL` is supplied for future data processing offline.

The Basket Object

Name	Required	Data Type	Description
ID	Yes	String	Integrator's unique ID for the basket that can be used for lookup later in data processing The ID must be the same in all requests until the basket is closed.
QUANTITY	No	Unsigned Integer	The number of items in the basket
ITEMS	No	Array	Array of basket Items
TOTAL	Yes	Signed Integer	Total value of basket in minor currency units. If total is not known set to zero (0)

POX

```
<BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
        </ITEMS>
       <TOTAL>1000</TOTAL>
</BASKET>
```

Json

```
"BASKET": {
    "ID": "",
    "ITEMS": [{
        "CATALOGID": "",
        "COST": 0,
        "ID": "",
        "QUANTITY": 0,
        "SKU": "",
        "VAT": 0,
        "VATAPPLIED": true,
        "VATRATE": 0.0
    }],
    "QUANTITY": 0
}
```

3.3. Basket Item

This is an object representing an item within a basket. The integrator does not have to supply this object in every transactional request, however epay recommend that at least the basket ID is supplied for future data processing offline, please refer to **section 6**.

The Basket Item Object

Name	Required	Data Type	Description
ID	Yes	String	Integrator's unique ID for the basket item that can be used for lookup later in data processing
QUANTITY	Yes	Unsigned Integer	The number of items in a basket
SKU	Yes	String	The Sku for the product or service Also known as a GTIN https://en.wikipedia.org/wiki/Global_Trade_Item_Number
COST	Yes	Unsigned Integer	The cost of the product or service, this will be used to cross reference the value credited back to the merchant account
ACTUALCOST	Yes	Unsigned Integer	The actual cost within the basket, if the item was discounted by a coupon, this should the value after the coupon has been applied. This will be used to cross reference the value credited back to the merchant account
VAT	No	Unsigned Integer	The VAT cost of the product or service
VATAPPLIED	No	Boolean	Whether VAT is needed to be applied to product or service
VATRATE	No	Double (5, 3)	VAT Rate applied to product, should only be provided if VAT Applied is set to true
CATALOGID	No	String	Catalog ID used to identify product by integrator, can either be the same as the Sku or integrator's own identifier
DESCRIPTION	No	String	Description of product

POX

```
<BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>05017726109448</SKU>
                <COST>100</COST>
                <ACTUALCOST>100</ACTUALCOST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>05017726109448</CATALOGID>
            </ITEM>
        </ITEMS>
</BASKET>
```

Json

```
"BASKET": {
    "ID": "",
    "ITEMS": [{
        "CATALOGID": "",
        "COST": 0,
         "ACTUALCOST": 0,
        "ID": "",
        "QUANTITY": 0,
        "SKU": "",
        "VAT": 0,
        "VATAPPLIED": true,
        "VATRATE": 0.0
    }],
    "QUANTITY": 0
}
```

3.4. Advertisement

This is an object representing an array of advertisements that is to be provided in the response back to the integrator. It will not always be provided in every response.

The advertisement object

Name	Required	Data Type	Description
ADVERT	Yes	Array OF ADVERT	Array of adverts to be printed

POX

```
<ADVERTISEMENT>
        <ADVERT>
        </ADVERT>
</ADVERTISEMENT>
```

Json

```
"ADVERTISEMENT": []
```

3.4.1. Advert

This is an object representing an individual advertisement that is to be provided in the response back to the integrator. It will not always be provided in every response.

The Advert object

Name	Required	Data Type	Description
LINE	Yes	Array of LINE	Array of lines

POX

```
<ADVERTISEMENT>
        <ADVERT>
          <LINE></LINE>
           <LINE></LINE>
        </ADVERT>
        <ADVERT>
          <LINE></LINE>
        </ADVERT>
</ADVERTISEMENT>
```

Json

```
"ADVERTISEMENT": [{
    "LINE": ["", "", ""]
}, {
    "LINE": ["", "", ""]
}, {
    "LINE": ["", "", ""]
}]
```

3.5. Card

This is an object representing a card, barcode or account reference scanned, swiped or keyed at the point of sale. This object is only needed when the product or service is requested via the usage of a card, barcode or account reference for identifying the product or customer.

The card object

Name	Required	Data Type	Description
PAN	Yes	String	Used for providing either: • top-up reference from a card • Top-up reference from Stripe2 when swiped through integrator's system • Scanned or keyed barcode from a card or mobile application • Activation barcode • Account reference from a bill • Redemption code / secret code
EAN	No	String	Used for providing the EAN from a physical product scanned or keyed through the integrator's system
STRIPE1	No	String	Used for providing Stripe1 from a card swipe
STRIPE2	No	String	Used for providing Stripe2 from a card swipe
STRIPE3	No	String	Used for providing Stripe3 from a card swipe
EXPIRY_YEAR	No	Unsigned Integer	Used for providing the Expiry year from a card Stripe2. This also can be keyed in
EXPIRY_MONTH	No	Unsigned Integer	Used for providing expiry month from a card Stripe2. This also can be keyed in
PIN	No	String Value can be plain or encrypted (3DES or other) as agreed between epay and integrator, can be a hexadecimal or Base64 encoded string	Required for some products when performing a REDEEM or status request type.
SERIAL	No	String	Will contain the public serial number of the card. *This field will only be returned in a* `RESPONSE` *and is not used in a* `REQUEST`.

POX

```
<CARD>
        <EAN></EAN>
        <PAN></PAN>
        <STRIPE1></STRIPE1>
        <STRIPE2></STRIPE2>
        <STRIPE3></STRIPE3>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
         <SERIAL></SERIAL>
</CARD>
```

Json

```
"CARD": {
        "EAN": "",
        "EXPIRY_MONTH": 0,
        "EXPIRY_YEAR": 0,
        "PAN": "",
        "STRIPE1": "",
        "STRIPE2": "",
        "STRIPE3": "",
        "SERIAL": ""
    }
```

3.6. PIN Credentials

This is an object representing a unique PIN or code for a requested product such as digital content or vouchers. This object will only be provided in the response to a product that is delivered to the customer using a code.

The PIN Credentials object

Name	Required	Data Type	Description
PIN	Yes	String	Will contain the unique code for the customer
SERIAL	Yes	String	Will contain the unique serial number to identify the code The `SERIAL` will be returned also as part of a cancelation for reference.
VALIDFROM	No	Date Time	Will contain the start date of when the code can be used from
VALIDTO	No	Date Time	Will contain the expiry date of when the code is valid until
EAN	No	String	EAN of Product
BUNDLE	No	Array of `PINCREDENTIALS`	An array of `PINCREDENTIALS` for providing additional codes attached to the primary product. Used when product is part of a bundle
SCODE	No	String	Secret Code for PIN

POX

```
<PINCREDENTIALS>
    <PIN>TEST23456</PIN>
    <SERIAL>987676288</SERIAL>
    <EAN>4260505539009</EAN>
    <VALIDTO>3000-01-01 00:00:00</VALIDTO>
    <SCODE></SCODE>
    <BUNDLE>
        <PINCREDENTIALS>
            <PIN>TEST90123456</PIN>
            <SERIAL>987676289</SERIAL>
            <EAN>4251404500085</EAN>
            <VALIDTO>3000-01-01 00:00:00</VALIDTO>
            <BUNDLE>
                <PINCREDENTIALS>
                    <PIN>TEST0123456</PIN>
                    <SERIAL>987676289</SERIAL>
                    <EAN>4251404500085</EAN>
                    <VALIDTO>3000-01-01 00:00:00</VALIDTO>
                </PINCREDENTIALS>
            </BUNDLE>
        </PINCREDENTIALS>
    </BUNDLE>
</PINCREDENTIALS>
```

Json

```
"PINCREDENTIALS": {
    "PIN": "TEST23456",
    "SERIAL": "987676288",
    "EAN": "4260505539009",
    "VALIDTO": "3000-01-01 00:00:00",
    "SCODE": "",
    "BUNDLE": [{
            "PIN": "TEST123456",
            "SERIAL": "987676289",
            "EAN": "4251404500085",
            "VALIDTO": "3000-01-01 00:00:00"
        },
        {
            "PIN": "TEST90123456",
            "SERIAL": "987676289",
            "EAN": "4251404500085",
            "VALIDTO": "3000-01-01 00:00:00",
            "BUNDLE": [{
                "PIN": "TEST0123456",
                "SERIAL": "987676289",
                "EAN": "4251404500085",
                "VALIDTO": "3000-01-01 00:00:00"
            }]
        }
    ]
}
```

3.7. Receipt

This is an object representing a request for receipt text to be supplied by the API. This object exists in both the request and response.

Though the request fields are not required, the object receipt is still needing to be supplied for the API to be generated. epay recommend providing the default language required for the integrator.

The Receipt object

Name	Required	Data Type	Description
LANGUAGE	No	String	Language of receipt text required. Supported for future use. **Is only required in the request**
TYPE	No	String	Type of receipt text required. Supported for future use. **Is only required in the request**
CHARSPERLINE	No	Unsigned Integer	Length of each line. Supported for future use, currently set to 32 as standard **Is only required in the request**
MERCHANT	No	Array	Array of lines representing the merchant receipt. **Only supplied in the response**
CUSTOMER	No	Array	Array of lines representing the customer receipt. **Only supplied in the response**

POX - Request

```
<RECEIPT>
    <LANGUAGE>ENG</LANGUAGE>
</RECEIPT>
```

Json - Request

```
{
  "RECEIPT": { "LANGUAGE": "ENG" }
}
```

POX - Response

```
<RECEIPT>
    <MERCHANT>
      <LINE>* Merchant Copy *</LINE>
    </MERCHANT>
    <CUSTOMER>
      <LINE>* Customer Copy *</LINE>
    </CUSTOMER>
 </RECEIPT>
```

Json - Response

```
"RECEIPT": {
    "CUSTOMER": ["", "", ""],
    "MERCHANT": ["", "", ""]
}
```

3.8. Balance

This is an object representing the balance of an account either for a service or product such as a gift card, the balance object itself is an array of the currency object described in **section 3.8.1**. This object will only be provided in the response for some methods called for a service or product.

The Balance object

Name	Required	Data Type	Description
CURRENCY	Yes	Array	Array of Currency object

POX

```
<BALANCE>
	<CURRENCY STANDARD="1">
		<ISOCODE>978</ISOCODE>
		<NAME>Euro</NAME>
		<FRACTION>2</FRACTION>
		<BALANCE>2000</BALANCE>
		<RESERVERD>0</RESERVED>
	</CURRENCY>
	<CURRENCY STANDARD="0">
		<ISOCODE>1</ISOCODE>
		<NAME>Points</NAME>
		<FRACTION>0</FRACTION>
		<BALANCE>9</BALANCE>
		<RESERVERD>0</RESERVED>
	</CURRENCY>
</BALANCE>
```

Json

```
  "BALANCE": {
    "CURRENCY": [
      {
        "STANDARD": "1",
        "ISOCODE": "978",
        "NAME": "Euro",
        "FRACTION": "2",
        "BALANCE": "2000",
        "RESERVED": "0"
      },
      {
        "STANDARD": "0",
        "ISOCODE": "1",
        "NAME": "Points",
        "FRACTION": "0",
        "BALANCE": "9",
        "RESERVED": "0"
      }
    ]
  }
```

3.8.1. Currency

This object is used to represent the balance of an account in the form of a currency. This object will only be provided in the response for some methods called for a service or product.

The currency object

Name	Required	Data Type	Description
STANDARD	Yes	String Attribute	Attribute within the main object. Attribute standard values are as follows, however are detailed in the specific product documentation • 1 • 2
ISOCODE	No	Unsigned Integer	Numeric ISO Country Code
NAME	No	String	Currency Name
FRACTION	No	Unsigned Integer	Number of decimal places
BALANCE	No	Signed Integer	Balance based on the minor denomination for the currency
RESERVED	No	Signed Integer	

POX

```
<CURRENCY STANDARD="1">
       <ISOCODE>978</ISOCODE>
       <NAME>Euro</NAME>
       <FRACTION>2</FRACTION>
       <BALANCE>2000</BALANCE>
       <RESERVERD>0</RESERVED>
</CURRENCY>
```

Json

```
      "CURRENCY": [
      {
        "STANDARD": "1",
        "ISOCODE": "978",
        "NAME": "Euro",
        "FRACTION": "2",
        "BALANCE": "2000",
        "RESERVED": "0"
      }
    ]
```

3.9. Product Info

This object is used to product information about the product and URLs used by the service or product which are either uniquely generated for the customer for each request or are used for other redemption mechanisms. This object will only be provided in the response for some methods called for a service or product.

The product info object

Name	Required	Data Type	Description
URL	Yes	String	Full URL for the product, either contains a unique URL for the consumer or a generic URL used for a specific purpose
TITLE	No	String	Title of product

POX

```
<PRODUCTINFO>
  <URL>https://www.google.co.uk</URL>
</PRODUCTINFO>
```

Json

```
{
  "PRODUCTINFO": { "URL": "https://www.google.co.uk" }
}
```

3.10. Device

This object is used to represent the device details such as hardware and software version, this can be used for reference when querying requests against the API, the device details are not used by the API for routing.

Name	Required	Data Type	Description
TERMINALTYPE	No	String	Type of device, this field is freeform
HWVERSION	No	String	Hardware Version, this field is freeform
HWSERIALNUMBER	No	String	Serial number for device, this field is freeform
SWVERSION	No	String	Version of software running on device, this field is freeform

POX

```
<DEVICE>
	<TERMINALTYPE></TERMINALTYPE>
	<HWVERSION></HWVERSION>
	<HWSERIALNUMBER></HWSERIALNUMBER>
	<SWVERSION></SWVERSION>
</DEVICE>
```

Json

```
"DEVICE": {
	 "HWSERIALNUMBER": "",
	 "HWVERSION": "",
	 "SWVERSION": "",
	 "TERMINALTYPE": ""
    }
```

3.11. Payment Information

This object is used to represent how the customer has paid for the product or service and can be used to share tokenization information for future payments that can be used with an integrator's payment gateway.

If the customer intends to initially pay for a product or service in a different way compared to how they want to for future payments, please use the `INITIALMETHOD` field to show this, for example in a `Buy Now Pay Later` scheme the customer would provide a preferred payment method, this method should be provide in the `METHOD` field with the `INITIALMETHOD` being set as `Buy Now Pay Later`.

For security reasons you should not provide full payment information within this object.

Name	Required	Data Type	Description
METHOD	Yes	String	Method of payment, please use one of the below. **Global Customers** American Express Discover and Diners Mastercard Visa Click to Pay Apple Pay Google Pay Microsoft Pay **Asia Pacific Customers** China UnionPay JCB Alipay WeChat Pay FPX BECS Direct Debit **European Customers** Cartes Bancaires Bancontact EPS Giropay iDEAL Multibanco Przelewy24 SOFORT Klarna Bacs Direct Debit SEPA Direct Debit **North American & Latin American Customers** Klarna Interac OXXO

			ACH Debit Pre-Authorized Debits ACH Credit Transfer **Other** Cash Buy Now Pay Later Coupon Reward Card Loyalty Card Unknown *Unknown should only be used if we have not provided a method in the list or you are unable to obtain this information.*
INITIALMETHOD	No	String	Initial method of payment, please use types from the above. This field is used when a customer may pay for a product/service by Cash however any future payment is done via another payment method that can supply a fingerprint.
CUSTOMERISPRESENT	Yes	Boolean	Customer was present at time of purchase. Default value is `FALSE`
CUSTOMEROPTEDIN	Yes	Boolean	Customer has opted into the usage of their payment information. This should be used explicitly by the integrator to let epay know that they have confirmed the customer has opted in for their payment information to be used for any future payments. Default value is `FALSE`
FINGERPRINT	No	String	Uniquely identifies the payment, also known as tokenized data. For the likes of Bacs/Becs the fingerprint should uniquely identify the bank account.
SCHEME	No	String	Card Scheme, you can use fully qualified card scheme like `Mastercard` or provide IIN.
SCHEMEIDENTIFIER	No	String	Used to provide scheme identifier from provider, if the provider requires several scheme identifiers

			please use a comma separator within the field. The field placement will need to be agreed with epay.
LAST4	No	String	The last four digits of the card or account number. If the method is using a wallet, this should be the last four digits of the device account number. For the likes of Bacs/Becs provide the last 4 digits of the bank number.
REFERENCE	No	String	Used to provide a payment reference.
BRAND	No	String	Used to provide information about the card brand.
COUNTRY	No	String	ISO code representing the country of the payment.
PAYMENTPROVIDER	No	String	Name of integrator's payment provider.

POX

```
<PAYMENTINFORMATION>
    <METHOD>Visa</METHOD>
    <INITIALMETHOD></INITIALMETHOD>
    <CUSTOMERISPRESENT>TRUE</CUSTOMERISPRESENT>
    <CUSTOMEROPTEDIN>TRUE</CUSTOMEROPTEDIN>
    <FINGERPRINT>xas83401mah127340</FINGERPRINT>
    <SCHEME>Mastercard</SCHEME>
    <SCHEMEIDENTIFIER>Mastercard</SCHEMEIDENTIFIER>
    <LAST4>8372</LAST4>
    <REFERENCE>ad23787123982377291</REFERENCE>
    <BRAND>Santander</BRAND>
    <COUNTRY>826</COUNTRY>
    <PAYMENTPROVIDER>Euronet</PAYMENTPROVIDER>
</PAYMENTINFORMATION>
```

Json

```
"PAYMENTINFORMATION": {
  "METHOD": "Visa",
  "INITIALMETHOD": "",
  "CUSTOMERISPRESENT": "TRUE",
  "CUSTOMEROPTEDIN": "TRUE",
  "FINGERPRINT": "xas83401mah127340",
  "SCHEME": "Mastercard",
  "SCHEMEIDENTIFIER": "023865739108",
  "LAST4": "8372",
  "REFERENCE": "ad23787123982377291",
  "BRAND": "Santander",
  "COUNTRY": "826",
  "PAYMENTPROVIDER": "Euronet"
}
```

3.12. Coupons

This is an object representing the coupons scanned at a Point of sale device, the coupons object itself is an array of the coupon object described in **section 3.13**. This object will be used in the request and response for methods relating to coupons.

The Coupons object

Name	Required	Data Type	Description
COUPON	Yes	Array	Array of Coupon object

POX

```
<COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <RESULT>0</RESULT>
            <CODE>13434393438121216483</CODE>
      </COUPON>
</COUPONS>
```

Json


```
"COUPONS": [{
     "CODE": "",
     "ID": "",
     "OFFER": {
         "BRAND": "",
         "CATEGORY": "",
         "DESCRIPTION": "",
         "DISCOUNT": {
             "AMOUNTOFF": 0,
             "PERCENTOFF": 0.0,
             "TYPE": "",
             "UNITOFF": 0
         },
         "ID": "",
         "PROMOTIONCODE": "",
         "SKUS": [{
             "DESCRIPTION": "",
             "DISCOUNTABLE": false,
             "ID": "",
             "QUANTITY": 0,
             "SKUVALUE": ""
         }],
         "TYPE": ""
     },
     "RESULT": 0,
     "VALIDITY": {
         "ACTIVE": 0,
         "EXPIRATIONDATE": "",
         "MINIMUMORDERVALUE": 0,
         "QUANTITY": 0,
         "REDEEMEDQUANTITY": 0,
         "STARTDATE": "0",
         "TYPE": "0"
     }
 }]
```

3.13. Coupon

This is an object used for representing an individual coupon scanned at a point of sale device, the coupon object will be used both in a request and response, where the response will contain the validity and offer relating to the individual coupon.

The coupon object

Name	Request/Response	Required	Data Type	Description
ID	Request Response	Yes	String	An ID to identify the position within the array, this will either be the Basket ID provided by the integrator or a unique ID generated by Accept
ISSUERID	Response	Yes	String	Issuer ID of the coupon registered on the provider system
CODE	Request Response	Yes	String	The coupon code provided by the integrator
ENTRYTYPE	Request	No	String	How the code was entered in the integrator's system
RESULT	Response	Yes	Unsigned Integer	Result Code for coupon based on the API method
VALIDITY	Response	Yes	Object	Object containing validity fields of coupon, see **section 3.14**
OFFER	Response	Yes	Object	Offer details, see **section 3.15**
PIN	Request	No	String	PIN Code used for redemption of coupon

POX

```
<COUPON>
    <ID>TESTCOUPON</ID>
    <RESULT>0</RESULT>
    <CODE>13434393438121216483</CODE>
    <VALIDITY>
        <ACTIVE>TRUE</ACTIVE>
        <STARTDATE>2017-01-01 00:00:00</STARTDATE>
        <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
        <QUANTITY>1</QUANTITY>
        <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
        <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
        <TYPE>DISCOUNT_VOUCHER</TYPE>
    </VALIDITY>
    <OFFER>
        <TYPE>DISCOUNT_VOUCHER</TYPE>
        <CATEGORY>BOGOFF</CATEGORY>
        <ID>OFFER1TEST</ID>
        <PROMOTIONCODE></PROMOTIONCODE>
        <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
        <DISCOUNT>
            <TYPE>UNIT</TYPE>
            <AMOUNTOFF>0</AMOUNTOFF>
            <PERCENTOFF>0</PERCENTOFF>
            <UNITOFF>1</UNITOFF>
        </DISCOUNT>
        <BRAND>Coca Cola</BRAND>
        <SKUS>
            <SKU>
                <ID>COKESKU</ID>
                <SKUVALUE>05017726109448</SKUVALUE>
                <QUANTITY>1</QUANTITY>
                <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
            </SKU>
        </SKUS>
    </OFFER>
    <PIN</PIN>
</COUPON>
```

Json


```
"COUPONS": [{
     "CODE": "",
     "ID": "",
     "OFFER": {
         "BRAND": "",
         "CATEGORY": "",
         "DESCRIPTION": "",
         "DISCOUNT": {
             "AMOUNTOFF": 0,
             "PERCENTOFF": 0.0,
             "TYPE": "",
             "UNITOFF": 0
         },
         "ID": "",
         "PROMOTIONCODE": "",
         "SKUS": [{
             "DESCRIPTION": "",
             "DISCOUNTABLE": false,
             "ID": "",
             "QUANTITY": 0,
             "SKUVALUE": ""
         }],
         "TYPE": ""
     },
     "RESULT": 0,
     "VALIDITY": {
         "ACTIVE": 0,
         "EXPIRATIONDATE": "",
         "MINIMUMORDERVALUE": 0,
         "QUANTITY": 0,
         "REDEEMEDQUANTITY": 0,
         "STARTDATE": "0",
         "TYPE": "0"
     },
    "PIN": ""
 }]
```

3.14. Validity

The validity object is used to represent the details of a coupon's validation characteristics.

The validity object

Name	Required	Data Type	Description
ACTIVE	Yes	Unsigned Integer	Value stating if the coupon is valid. 0 = not active 1 = active
STARTDATE	Yes	Date Time	Start date and time when the coupon is valid
EXPIRATIONDATE	No	Date Time	Expiry date and time when the coupon is no longer valid, if the field is not supplied it will be treated as unlimited
QUANTITY	No	Unsigned Integer	Number of times the coupon can be used
REDEEMEDQUANTITY	No	Unsigned Integer	Number of times the coupon has been used
MINIMUMORDERVALUE	No	Unsigned Integer	Total basket value in minor currency units that qualifies for the coupon to be used
TYPE	Yes	String	Type of offer: DISCOUNT_VOUCHER GIFT_VOUCHER
VALIDPERIODS	No	Array of `VALIDPERIOD`	Array of VALIDPERIOD used to state if valid only within certain time frames
PINREQUIRED	No	Boolean	When set the redemption call requires a redemption PIN to be supplied. Default value is `FALSE`

POX

```
    <VALIDITY>
        <ACTIVE>0</ACTIVE>
        <STARTDATE>2017-01-01 00:00:00</STARTDATE>
        <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
        <QUANTITY>1</QUANTITY>
        <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
        <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
        <TYPE>DISCOUNT_VOUCHER</TYPE>
        <PINREQUIRED>false</PINREQUIRED>
    </VALIDITY>
```

Json

```
"VALIDITY": {
            "ACTIVE": 0,
            "EXPIRATIONDATE": "",
            "MINIMUMORDERVALUE": 0,
            "QUANTITY": 0,
            "REDEEMEDQUANTITY": 0,
            "STARTDATE": "",
            "TYPE": "",
             "PINREQUIRED": false
        }
```

3.15. Offer

The offer object is used to represent the details of a campaign/offer that relates to a coupon, including which SKUs should be within a basket and the discount to apply.

The offer object

Name	Required	Data Type	Description

TYPE	Yes	String	Type of offer
			
			`DISCOUNT_VOUCHER`
			`GIFT_VOUCHER`
			`CATEGORY_VOUCHER`

CATEGORY	No	String	Categorization of offer
ID	No	String	Offer ID corresponding to the campaign on the coupon provider system
PROMOTIONCODE	No	String	Promotion code that is used at till, if the integrator requires further action
DESCRIPTION	Yes	String	Description of offer
DISCOUNT	Yes	Object	`discount` object see **section 3.16**
BRAND	No	String	Brand title

CHECKTYPE	No	String	Type of check required for offer to be validate by integrator
			
			`ONE` – One SKU in the SKUS or `SKUGROUPS` exist in the `BASKET`
			`ALL` – All `SKU` in the `SKUS` or `SKUGROUPS` exists in the `BASKET`
			`ONEINEACHGROUP` – One of `SKU` in each GROUP exists in the `BASKET`
			
			If not provided the default is `ALL`

SKUS	No	Array of `SKU` objects	Array of `SKU` objects that can be used by the integrator to check the basket with. See **section 3.17** If `TYPE` is set to `DISCOUNT_VOUCHER` or `GIFT_VOUCHER`, the integrator should review this field.
SKUGROUPS	No	Array of `SKUGROUP` objects	Array of `SKUGROUP` objects that is used to group `SKU` objects in to groups for certain `CHECKTYPE`.
EXCLUDED	No	Array of `SKU` objects	Array of SKU objects that are to be excluded by the integrator. See **section 3.17** If `TYPE` is set to `DISCOUNT_VOUCHER` or `GIFT_VOUCHER`, the integrator should review this field.

MERCHANTCATEGORY	No	String	Merchant Category is used when an offer relates to a category which does not have a SKU list associated. i.e. Entire Meat Counters If `TYPE` is set to `CATEGORY_VOUCHER` the integrator should review this field

POX

```
<OFFER>
    <TYPE>DISCOUNT_VOUCHER</TYPE>
    <CATEGORY>BOGOFF</CATEGORY>
    <ID>OFFER1TEST</ID>
    <PROMOTIONCODE></PROMOTIONCODE>
    <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
    <DISCOUNT>
        <TYPE>UNIT</TYPE>
        <AMOUNTOFF>0</AMOUNTOFF>
        <PERCENTOFF>0</PERCENTOFF>
        <UNITOFF>1</UNITOFF>
    </DISCOUNT>
    <BRAND>Coca Cola</BRAND>
    <SKUS>
        <SKU>
            <ID>COKESKU</ID>
            <SKUVALUE>05017726109448</SKUVALUE>
            <QUANTITY>1</QUANTITY>
            <DISCOUNTABLE>TRUE</DISCOUNTABLE>
            <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
        </SKU>
    </SKUS>
</OFFER>
```

Json

```
"OFFER": {
    "BRAND": "",
    "CATEGORY": "",
    "DESCRIPTION": "",
    "DISCOUNT": {
        "AMOUNTOFF": 0,
        "PERCENTOFF": 0.0,
        "TYPE": "",
        "UNITOFF": 0
    },
    "ID": "",
    "PROMOTIONCODE": "",
    "SKUS": [{
        "DESCRIPTION": "",
        "DISCOUNTABLE": false,
        "ID": "",
        "QUANTITY": 0,
        "SKUVALUE": ""
    }],
    "TYPE": ""
}
```

3.16. Discount

The discount object is used to represent what discount is to be applied to a basket, please refer to the discountable field in the SKU item.

The discount object

Name	Required	Data Type	Description
TYPE	Yes	String	Type of discount: AMOUNT PERCENT UNIT ITEM SETVALUE ITEM is used when to tell the integrator that there is a discount applied at the item level. The integrator should look for any item with a discount set and the field `DISCOUNTABLE` set to `TRUE`
AMOUNTOFF	No	Unsigned Integer	Amount off value, set if `TYPE` is set to AMOUNT Must be in minor currency units
PERCENTOFF	No	Double (5, 3)	Percentage off value, set if `TYPE` is set to PERCENT
UNITOFF	No	Unsigned Integer	Number of items/units to mark as free Set if `TYPE` is set to UNIT
SETVALUE	No	Unsigned Integer	Value used to be set against a `SKU`, set if `TYPE` is set to `SETVALUE` Must be in minor currency units

POX

```
        <DISCOUNT>
            <TYPE>UNIT</TYPE>
            <AMOUNTOFF>0</AMOUNTOFF>
            <PERCENTOFF>0</PERCENTOFF>
            <UNITOFF>1</UNITOFF>
        </DISCOUNT>
```

Json

```
"DISCOUNT": {
                "AMOUNTOFF": 0,
                "PERCENTOFF": 0.0,
                "TYPE": "",
                "UNITOFF": 0,
                "SETVALUE": 0
            }
```

3.17. SKU

The SKU object is used to represent a SKU item within a basket or a SKU that is used within a campaign/offer.

The SKU object

Name	Required	Data Type	Description
ID	Yes	String	Unique ID of SKU Item
SKUVALUE	Yes	String	SKU value either EAN or UPC Also known as a GTIN https://en.wikipedia.org/wiki/Global_Trade_Item_Number
QUANTITY	Yes	Unsigned Integer	Number of items that should be in basket Default should be set to one (1)
DISCOUNTABLE	Yes	Boolean	Used to mark item in basket that can have the discount applied to, must be set to true if using `DISCOUNT.TYPE` of `UNIT`
DESCRIPTION	Yes	String	Description of product or service, this is used to be printed from the device to instruct user to scan items when baskets are not provided automatically through the device, i.e. a separate device is used for the transaction
DISCOUNT	No	Object	`DISCOUNT` Object, used when a discount is applied to the `SKU` item. The `DISCOUNTABLE` field must be set to `TRUE`.
MANUFACTURERCODE	No	String	Manufacturer Code, used to relate to many SKU items. If `SKUVALUE` is empty integrator must check if `MANUFACTURERCODE` is provided and determine if one or more SKUs are related.
TYPE	Yes	String	Type of product or service. The `TYPE` field could be used to group types of products from exclusion.

POX

```
            <SKU>
                <ID>COKESKU</ID>
                <SKUVALUE>05017726109448</SKUVALUE>
                <QUANTITY>1</QUANTITY>
                <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
            </SKU>
```

Json

```
{
    "DESCRIPTION": "",
    "DISCOUNTABLE": false,
    "ID": "",
    "QUANTITY": 0,
    "SKUVALUE": ""
}
```

3.18. SKU Group

The SKU Group object is used to represent a group of SKU items within a campaign/offer, where an action is required by the integrator to validate the offer.

The SKU group object

Name	Required	Data Type	Description
ID	Yes	String	ID of Group
SKUS	No	Array of `SKU` objects	Array of `SKU` objects that can be used by the integrator to check the basket with. See **section 3.17** If `TYPE` is set to `DISCOUNT_VOUCHER` or `GIFT_VOUCHER`, the integrator should review this field.

POX

```
<SKUGROUPS>
    <GROUP>
        <ID>1</ID>
        <SKUS>
            <SKU>
                <ID>COKESKU</ID>
                <SKUVALUE>05017726109448</SKUVALUE>
                <QUANTITY>1</QUANTITY>
                <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
            </SKU>
        </SKUS>
    </GROUP>
</SKUGROUPS>
```

Json

```
"SKUGROUPS": [
    {
      "ID": "1",
      "SKUS": [
        {
          "ID": "COKESKU",
          "SKUVALUE": "05017726109448",
          "QUANTITY": "1",
          "DISCOUNTABLE": "TRUE",
          "DESCRIPTION": "Diet Coke 500ml"
        }
      ]
    }
  ]
```

3.19. Lottery

The lottery object is used to store the lottery details the customer has requested as part of playing a lottery game.

The lottery object

Name	Required	Data Type	Description
DAYS	Yes	Unsigned Integer	Days required to be played on See day matrix table below in **section 3.19.1**
GAMES	Yes	Unsigned Integer	Number of games to be played Weeks or Game Draws
NUMBERS	Yes	Array of Lines Object	Used to represent the array of line chosen by a customer
BARCODE	No	String	Barcode used for quick purchase
TICKETID	No	String	Ticket ID used for pay-out

POX

```
<LOTTERY>
    <DAYS></DAYS>
    <GAMES></GAMES>
    <NUMBERS>
        <LINE></LINE>
    </NUMBERS>
    <TICKETID></TICKETID>
</LOTTERY>
```

Json

```
"LOTTERY": {
        "DAYS": 0,
        "GAMES": 0,
        "NUMBERS": [{
                "",
                ""],
            "TICKETID": ""
        }
```

3.19.1. Day Matrix

Days are represented as BITs and are set out as below.

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
0x0100	0x0200	0x0400	0x0800	0x1000	0x2000	0x4000

3.20. Tolling

The tolling object is used to represent the details of a customer paying for one or more trips via a toll road or for a past trip on a toll road. The tolling object is used in a request and response method for a tolling product. It is recommended to review the PCD for tolling as not all service providers support the purchasing of past trips or the purchasing of past trips and future trips combined.

The tolling object

Name	Request/Response	Required	Data Type	Description
VEHICLE	Request Response	Yes	Object	Vehicle details see **section 3.22**
ROADS	Response	Conditional	Array of `ROAD`	Array of road object, see **section 3.21** The `ROADS` object is returned in an enquiry for a tolling product and is only used for the customer to select from.
TRIPS	Request	Conditional	Array of `TRIP`	Array of `TRIP` object, see **section 3.23** The `TRIPS` object is required when performing a request to pay for a trip in the future.
PASTTRIPS	Request Response	Conditional	Array of `TRIP`	Array of `TRIP` object representing past trips, see **section 3.23.** If the `PASTTRIPS` object is returned during an enquiry it will represent any outstanding tolls required to be paid for The `PASTTRIPS` object is required in the request if the customer has chosen to pay for one or more past trips
MAXTRIPS	Response	Yes	Unsigned Integer	Setting for tolling provider for the maximum number of trips allowed by consumer If value is set to one (1), the consumer cannot choose a number of trips
TRIPDATEALLOWED	Response	Yes	Boolean	If set to `TRUE` th e consumer can choose a date to be specified
PROVIDER	Response	No	String	Tolling Provider name
STATES	Response	No	Array of `STATE`	Array of `STATE` object, **see section 3.23**.

				The STATES object is required if the state of registration is required. If its been provided the integrator must provide a selection list.
TARIFFDATE	Request	No	DateTime	If set with a date the tariffs supplied will be for the date specified. This is used for past tariffs.

POX

```
<TOLLING>
    <VEHICLE>
        <LPN></LPN>
        <CLASS></CLASS>
        <PURPOSEOFUSE></PURPOSEOFUSE>
    </VEHICLE>
     <STATES>
        <STATE>
            <ID>QLD</ID>
            <DESCRIPTION>Queensland</DESCRIPTION>
        </STATE>
    </STATES>
    <ROADS>
        <ROAD>
            <ID></ID>
            <CODE></CODE>
            <DESCRIPTION></DESCRIPTION>
            <TARIFF>0</TARIFF>
            <SERVICEFEE>0</SERVICEFEE>
        </ROAD>
    </ROADS>
    <TRIPS>
        <TRIP>
            <ROAD>
                <ID></ID>
                <CODE></CODE>
                <DESCRIPTION></DESCRIPTION>
                <TARIFF>0</TARIFF>
                <SERVICEFEE>0</SERVICEFEE>
            </ROAD>
            <TRIPS>0</TRIPS>
            <TOTAL>0</TOTAL>
        </TRIP>
    </TRIPS>
    <MAXTRIPS>1</MAXTRIPS>
    <TRIPDATEALLOWED>FALSE</TRIPDATEALLOWED>
</TOLLING>
```

Json


```
"TOLLING": {
    "PASTTRIPS": [
      {
        "ID": "",
        "ROAD": {
          "CODE": "",
          "DESCRIPTION": "",
          "ID": "",
          "SERVICEFEE": 0,
          "TARIFF": 0
        },
        "TOTAL": 0,
        "TRIPDATE": "",
        "TRIPS": 0
      }
    ],
    "ROADS": [
      {
        "CODE": "",
        "DESCRIPTION": "",
        "ID": "",
        "SERVICEFEE": 0,
        "TARIFF": 0
      }
    ],
    "STATES": [
      {
        "DESCRIPTION": "",
        "ID": ""
      }
    ],

    "TRIPS": [
      {
        "ID": "",
        "ROAD": {
          "CODE": "",
          "DESCRIPTION": "",
          "ID": "",
          "SERVICEFEE": 0,
          "TARIFF": 0
        },
        "TOTAL": 0,
        "TRIPDATE": "",
        "TRIPS": 0
      }
    ],
    "VEHICLE": {
      "CLASS": "",
      "LPN": "",
      "PURPOSEOFUSE": ""
    },
    "MAXTRIPS": 1,
    "TRIPDATEALLOWED": "TRUE"
  }
```

3.21. Road

The road object is used to represent a road which is used with tolling products.

The road object

Name	Required	Data Type	Description
ID	Yes	String	Road ID
CODE	Yes	String	Road code
DESCRIPTION	Yes	String	Title of road or road name
TARIFF	Yes	Unsigned Integer	Tariff for road
SERVICEFEE	No	Unsigned Integer	Service fee applied to road
DENOMINATIONS	No	Array of AMOUNT	Array of mount values that a consumer can choose from
WEEKENDONLY	No	Boolean	Used to state if Road Tariff can only be used on Weekend if a trip Date is a weekend only date.
CLASSID	No	String	If the field is set with a class ID this tariff can only be selected if the class of vehicle matches. If no value is supplied tariff applies to all classes.
BARCODE	No	String	Barcode used to identity toll road chosen. Mainly used for pre-tender purchasing by non-integrated retailers.

POX

```
<ROAD>
    <ID></ID>
    <CODE></CODE>
    <DESCRIPTION></DESCRIPTION>
    <TARIFF>0</TARIFF>
    <SERVICEFEE>0</SERVICEFEE>
    <DENOMINATIONS>
        <AMOUNT>1000</AMOUNT>
    </DENOMINATIONS>
    <WEEKENDONLY>FALSE</WEEKENDONLY>
    <CLASSID></CLASSID>
    <BARCODE></BARCODE>
</ROAD>
```

Json

```
{
  "ID": "",
  "CODE": "",
  "DESCRIPTION": "",
  "TARIFF": 0,
  "SERVICEFEE": 0,
  "DENOMINATIONS": [
    1000,
    2000
  ],
  "WEEKENDONLY": 0,
  "CLASSID": "",
  "BARCODE": ""
}
```

3.22. Vehicle

The vehicle object is used to represent a vehicle which is used with tolling products.

The vehicle object

Name	Required	Data Type	Description
LPN	Yes	String	License plate number
CLASS	No	String	Type of vehicle
PURPOSEOFUSE	Yes	String	Purpose of use for trips
MAKE	No	String	Make of vehicle
MODEL	No	String	Model of vehicle
COLOUR	No	String	Colour of vehicle
CLASSID	Yes	String	Class ID of vehicle
STATE	No	String	ID of state from STATE object

POX

```
<VEHICLE>
    <LPN></LPN>
    <CLASS></CLASS>
    <PURPOSEOFUSE></PURPOSEOFUSE>
    <MAKE></MAKE>
    <MODEL></MODEL>
    <COLOUR></COLOUR>
    <CLASSID></CLASSID>
    <STATE></STATE>
</VEHICLE>
```

Json

```
"VEHICLE": {
    "LPN": "",
    "CLASS": "",
    "PURPOSEOFUSE": "",
    "MAKE": "",
    "MODEL": "",
    "COLOUR": "",
    "CLASSID": "",
    "STATE": ""
  }
```

3.23. Trip

The trip object is used to represent a trip which is used with tolling products, the trip object can be used for detailing future trips or past trips that have not been paid for.

The trip object

Name	Required	Data Type	Description
ROAD	Yes	Object	Details of road, see **section 3.21**
TRIPS	Yes	Unsigned Integer	Number of trips
TOTAL	Yes	Unsigned Integer	Total fee for road in minor currency units {ROAD.TOLLROADTARIFF} * {TRIPS}) + {ROAD.SERVICEFEE}
TRIPDATE	No	Date Time	Date of trip *If the trip date is not supplied in a request the current trip date will be taken using the time zone of the tolling service*
TRIPTIME	No	Date Time	Time of trip *If the trip time is not supplied in a request the current trip date will be taken using the time zone of the tolling service*
ID	Yes if Past Trip	String	Trip identifier used by tolling service if trip was in the past

POX

```
<TRIP>
    <ROAD>
        <ID></ID>
        <CODE></CODE>
        <DESCRIPTION></DESCRIPTION>
        <TARIFF>0</TARIFF>
        <SERVICEFEE>0</SERVICEFEE>
    </ROAD>
    <TRIPS>0</TRIPS>
    <TOTAL>0</TOTAL>
    <TRIPDATE></TRIPDATE>
    <ID></ID>
</TRIP>
```

Json

```
{
        "ID": "",
        "ROAD": {
            "CODE": "",
            "DESCRIPTION": "",
            "ID": ""
            "SERVICEFEE": 0,
            "TARIFF": 0
        },
        "TOTAL": 0,
        "TRIPDATE": "",
        "TRIPS": 0
}
```

3.24. State

The state object issued to represent a state of registration for a vehicle.

The state object

Name	Required	Data Type	Description
ID	Yes	String	ID of State
DESCRIPTION	Yes	String	Long description of state

POX

```
<STATE>
    <ID></ID>
    <DESCRIPTION></DESCRIPTION>
</STATE>
```

Json

```
{
        "ID": "",
        "DESCRIPTION": ""
}
```

3.25. Customer

The customer object is used to represent the details of a customer.

The customer object

Name	Required	Data Type	Description
TITLE	No	String	Title for customer
FIRSTNAME	No	String	First name of customer
MIDDLENAME	No	String	Middle names for customer
LASTNAME	No	String	Surname of customer
FULLNAME	No	String	Represent full name of customer, if this is provided the previous fields should be blank.
AGE	No	Unsigned Integer	Age of customer
DATEOFBIRTH	No	Date Time	Date of birth for customer
DELIVERYADDRESS	No	ADDRESS Object	Delivery Address
BILLINGADDRESS	No	ADDRESS Object	Billing Address
PHONENUMBER	No	String	Phone number
FAXNUMBER	No	String	Fax Number
EMAILADDRESS	No	String	Email address of customer
MOBILENUMBER	No	String	Mobile number for customer
PASSPORTNUMBER	No	String	Passport number for customer
DRIVINGLICENSENO	No	String	Driving license number
CUSTOMERID	No	String	Unique customer ID used by integrator
SOCIALNUMBER	No	String	Social Number
DIGITALLOCKERID	No	String	Used to supply Digital Locker Id

POX

```
<CUSTOMER>
    <TITLE></TITLE>
    <FIRSTNAME></FIRSTNAME>
    <MIDDLENAME></MIDDLENAME>
    <LASTNAME></LASTNAME>
    <FULLNAME></FULLNAME>
    <AGE>0</AGE>
    <DATEOFBIRTH></DATEOFBIRTH>
    <DELIVERYADDRESS>
        <ADDRESS1></ADDRESS1>
        <ADDRESS2></ADDRESS2>
        <ADDRESS3></ADDRESS3>
        <CITY></CITY>
        <COUNTY></COUNTY>
        <POSTALCODE></POSTALCODE>
        <COUNTRY></COUNTRY>
    </DELIVERYADDRESS>
    <BILLINGADDRESS>
        <ADDRESS1></ADDRESS1>
        <ADDRESS2></ADDRESS2>
        <ADDRESS3></ADDRESS3>
        <CITY></CITY>
        <COUNTY></COUNTY>
        <POSTALCODE></POSTALCODE>
        <COUNTRY></COUNTRY>
    </BILLINGADDRESS>
    <PHONENUMBER></PHONENUMBER>
    <FAXNUMBER></FAXNUMBER>
    <EMAILADDRESS></EMAILADDRESS>
    <MOBILENUMBER></MOBILENUMBER>
    <PASSPORTNUMBER></PASSPORTNUMBER>
    <DRIVINGLICENSENO></DRIVINGLICENSENO>
    <CUSTOMERID></CUSTOMERID>
    <SOCIALNUMBER></SOCIALNUMBER>
    <DIGITALLOCKERID></DIGITALLOCKERID>
</CUSTOMER>
```

Json


```
"CUSTOMER": {
     "AGE": 0,
     "BILLINGADDRESS": {
         "ADDRESS1": "",
         "ADDRESS2": "",
         "ADDRESS3": "",
         "CITY": "",
         "COUNTRY": "",
         "COUNTY": "",
         "POSTALCODE": ""
     },
     "CUSTOMERID": "",
     "DATEOFBIRTH": "",
     "DELIVERYADDRESS": {
         "ADDRESS1": "",
         "ADDRESS2": "",
         "ADDRESS3": "",
         "CITY": "",
         "COUNTRY": "",
         "COUNTY": "",
         "POSTALCODE": ""
     },
     "DRIVINGLICENSENO": "",
     "EMAILADDRESS": "",
     "FAXNUMBER": "",
     "FIRSTNAME": "",
     "FULLNAME": "",
     "LASTNAME": "",
     "MIDDLENAME": "",
     "MOBILENUMBER": "",
     "PASSPORTNUMBER": "",
     "PHONENUMBER": "",
     "SOCIALNUMBER": "",
     "TITLE": "",
     "DIGITALLOCKERID": ""
 }
```

3.26. Address

The address object is used to represent the details of an address that can be used to represent a delivery or billing address.

The address object

Name	Required	Data Type	Description
ADDRESS1	No	String	Address line 1
ADDRESS2	No	String	Address line 2
ADDRESS3	No	String	Address line 3
CITY	No	String	City
COUNTY	No	String	County
POSTALCODE	No	String	Postal code / zip code
COUNTRY	No	String	Country

POX

```
<ADDRESS1></ADDRESS1>
<ADDRESS2></ADDRESS2>
<ADDRESS3></ADDRESS3>
<CITY></CITY>
<COUNTY></COUNTY>
<POSTALCODE></POSTALCODE>
<COUNTRY></COUNTRY>
```

Json

```
{
    "ADDRESS1": " ",
    "ADDRESS2": " ",
    "ADDRESS3": " ",
    "CITY": " ",
    "COUNTY": " ",
    "POSTALCODE": " ",
    "COUNTRY": " "
}
```

3.27. CustomData

The `CUSTOMDATA` object is used to represent additional meta data that does not fit into any other object, this object can be sent in the request and response.

The custom data object

Name	Required	Data Type	Description
ITEMS	Yes	Array of `ITEM` object	Array of ITEM object

3.27.1. Item

The `ITEM` object is used to represent an individual item object within the `CUSTOMDATA` object that can be used to store any information that does not fit into any other object.

The item object

Name	Required	Data Type	Description
VALUES	Yes	Array of `VALUE` object	Array of `VALUE` object

3.27.2. Value

The `VALUE` object is used to represent one key/value that does not fit into any other object.

The value object

Name	Required	Data Type	Description
KEY	Yes	String *Attribute*	Key to uniquely identify the value
VALUE	Yes	String *InnerText*	Value in string format for the information to be stored

POX

```
<CUSTOMDATA>
    <ITEM>
        <VALUE KEY="VRM">ABC123</VALUE>
        <VALUE KEY="STARTDATE">2011-01-07</VALUE>
        <VALUE KEY="ENDDATE">2011-01-07</VALUE>
    </ITEM>
    <ITEM>
        <VALUE KEY="VRM">DEF456</VALUE>
        <VALUE KEY="STARTDATE">2011-01-07</VALUE>
        <VALUE KEY="ENDDATE">2011-01-07</VALUE>
    </ITEM>
</CUSTOMDATA>
```

Json

```
"CUSTOMDATA": [
    [{
        "KEY": "",
        "Value": ""
    }, {
        "KEY": "",
        "Value": ""
    }]
]
```

3.28. ValidPeriod

The `VALIDPERIOD` object is used to represent a time frame either based on a full date and time, date or time.

Name	Required	Data Type	Description

TYPE	Yes	String	Type of Period
			
			`DATETIME`
			`DATE`
			`TIME`

VALIDFROM	Yes	Date/Time	Period start
VALIDTO	Yes	Date/Time	Period end

POX

```
<VALIDPERIOD>
    <TYPE>DATE</TYPE>
    <VALIDFROM>2011-01-07</VALIDFROM>
    <VALIDTO>2011-01-08</VALIDTO>
</VALIDPERIOD>
```

Json

```
{
  "TYPE": "DATE",
  "VALIDFROM": "2011-01-07",
  "VALIDTO": "2011-01-08"
}
```

3.29. FareProduct

The `FAREPRODUCT` object is used to represent a product for ticketing.

Name	Required	Data Type	Description
ID	Yes	String	ID representing Fare
PRICE	Yes	Unsigned Integer	Price of product
GST	No	Unsigned Integer	GST of product
TITLE	Yes	String	Title of product

POX

```
<FAREPRODUCT>
    <ID></ID>
    <PRICE></PRICE>
    <GST></GST>
    <TITLE></TITLE>
</FAREPRODUCT>
```

Json

```
{
  "ID": "",
  "PRICE": 0,
  "GST": 0,
  "TITLE": ""
}
```

3.30. Pass

The PASS object is used to represent a product for ticketing.

Name	Required	Data Type	Description
ID	Yes	String	ID representing Pass
ORIGIN	No	String	Starting Origin if used for travel
DESTINATION	No	String	End Destination if used for travel
STARTDATE	Yes	DateTime	Start Date
EXPIRYDATE	Yes	DateTime	Expiry Date of Pass
FAREPRODUCTID	Yes	String	ID of FAREPRODUCT

POX

```
<PASS>
    <ID></ID>
    <ORIGIN></ORIGIN>
    <DESTINATION></DESTINATION>
    <STARTDATE></STARTDATE>
    <EXPIRYDATE></EXPIRYDATE>
    <FAREPRODUCTID></FAREPRODUCTID>
</PASS>
```

Json

```
{
  "ID": " ",
  "ORIGIN": " ",
  "DESTINATION": " ",
  "STARTDATE": " ",
  "EXPIRYDATE": " ",
  "FAREPRODUCTID": " "
}
```

3.31. Parcel

The `PARCEL` object is used to represent a parcel for collection or delivery.

Name	Required	Data Type	Description
ID	Yes	String	ID representing Parcel
COLLECTIONCODE	No	String	Collection Code
CUSTOMER	No	`CUSTOMER` object	Represents customer who owns the parcel
PROVIDERID	No	String	ID used by the parcel provider
STATE	Yes	String	State of parcel
DATECREATED	Yes	DateTime	Date/Time Parcel was created within provider database
DATERECEIVED	No	DateTime	Date/Time Parcel was received
DATECOLLECTED	No	DateTime	Date/Time Parcel was collected

POX

```
<PARCEL>
    <ID></ID>
    <COLLECTIONCODE></COLLECTIONCODE>
    <CUSTOMER>
        <TITLE></TITLE>
        <FIRSTNAME></FIRSTNAME>
        <MIDDLENAME></MIDDLENAME>
        <LASTNAME></LASTNAME>
        <FULLNAME></FULLNAME>
        <AGE>0</AGE>
        <DATEOFBIRTH></DATEOFBIRTH>
        <DELIVERYADDRESS>
            <ADDRESS1></ADDRESS1>
            <ADDRESS2></ADDRESS2>
            <ADDRESS3></ADDRESS3>
            <CITY></CITY>
            <COUNTY></COUNTY>
            <POSTALCODE></POSTALCODE>
            <COUNTRY></COUNTRY>
        </DELIVERYADDRESS>
        <BILLINGADDRESS>
            <ADDRESS1></ADDRESS1>
            <ADDRESS2></ADDRESS2>
            <ADDRESS3></ADDRESS3>
            <CITY></CITY>
            <COUNTY></COUNTY>
            <POSTALCODE></POSTALCODE>
            <COUNTRY></COUNTRY>
        </BILLINGADDRESS>
        <PHONENUMBER></PHONENUMBER>
        <FAXNUMBER></FAXNUMBER>
        <EMAILADDRESS></EMAILADDRESS>
        <MOBILENUMBER></MOBILENUMBER>
        <PASSPORTNUMBER></PASSPORTNUMBER>
        <DRIVINGLICENSENO></DRIVINGLICENSENO>
        <CUSTOMERID></CUSTOMERID>
        <SOCIALNUMBER></SOCIALNUMBER>
    </CUSTOMER>
    <PROVIDERID></PROVIDERID>
    <STATE></STATE>
    <DATECREATED></DATECREATED>
    <DATERECEIVED></DATERECEIVED>
    <DATECOLLECTED></DATECOLLECTED>
</PARCEL>
```

Json


```
{
  "ID": "",
  "COLLECTIONCODE": "",
  "PROVIDERID": "",
  "STATE": "",
  "CUSTOMER": {
    "AGE": 0,
    "BILLINGADDRESS": {
      "ADDRESS1": "",
      "ADDRESS2": "",
      "ADDRESS3": "",
      "CITY": "",
      "COUNTRY": "",
      "COUNTY": "",
      "POSTALCODE": ""
    },
    "CUSTOMERID": "",
    "DATEOFBIRTH": "",
    "DELIVERYADDRESS": {
      "ADDRESS1": "",
      "ADDRESS2": "",
      "ADDRESS3": "",
      "CITY": "",
      "COUNTRY": "",
      "COUNTY": "",
      "POSTALCODE": ""
    },
    "DRIVINGLICENSENO": "",
    "EMAILADDRESS": "",
    "FAXNUMBER": "",
    "FIRSTNAME": "",
    "FULLNAME": "",
    "LASTNAME": "",
    "MIDDLENAME": "",
    "MOBILENUMBER": "",
    "PASSPORTNUMBER": "",
    "PHONENUMBER": "",
    "SOCIALNUMBER": "",
    "TITLE": ""
  },
  "DATECREATED": "2017-02-02",
  "DATERECEIVED": "2017-02-02",
  "DATECOLLECTED": "2017-02-02"
}
```

3.32. Gifting

The `GIFTING` object is used to represent details used to allow a customer to gift their purchase to someone else.

Name	Required	Data Type	Description
GIFTINGCODE	Yes	String	Unique gifting code for purchase
SERVICEURL	Yes	String	Gifting service URL
GIFTURL	Yes	String	Unique gift URL for purchase that can be used and represented as a QR Code
EXPIRYDATE	Yes	DateTime	Expiration date of Gifting code

POX

```
<GIFTING>
    <CODE>A1212AGKB1</CODE>
    <SERVICEURL>https://www.mygifttoyou.com</SERVICEURL>
    <GIFTURL>https://www.mygifttoyou.com/gift/?uid=A1212AGKB1</GIFTURL>
    <EXPIRYDATE>2018-07-05 00:00:00</EXPIRYDATE>
</GIFTING>
```

Json


```
{
  "GIFTING": {
    "CODE": "A1212AGKB1",
    "SERVICEURL": "https://www.mygifttoyou.com",
    "GIFTURL": "https://www.mygifttoyou.com/gift/?uid=A1212AGKB1",
    "EXPIRYDATE": "2018-07-05 00:00:00"
  }
}
```

3.33. Prompt

The PROMPT object is used to represent an instruction for a integrator to further execute on their device before re-submitting a new request.

Name	Required	Data Type	Description
UNIQUENAME	Yes	String	The unique name for the prompt
ICON	No	String	Icon file name to be displayed If `TYPE` = `Button` then should be shown as button option. Icons would be provided by epay
PROMPTTEXTID	Yes	String	Prompt Text ID used to lookup local translations
PROMPTTEXT	No	String	Default text displayed in a prompt window. It gives the user guidance on the information required.
DEFAULTVALUE	No	String	The default response to a prompt that the user can select. Value will be a string and require the integrator to convert to the type required.
ORDER	Required	Integer	The order of a prompt relative to other prompts in a prompt window.
ERRORMESSAGEID	Yes	String	Error Message ID used to lookup local translations
ERRORMESSAGE	No	String	Default text displayed if a user supplies inappropriate information to a prompt.
TYPE	Yes	String	The type of Prompt. Value: One of the following values: `String`, `Integer`, `Float`, `Password`, `Time`, `Lookup`, `Fixed Choice`, `Dynamic Choice`, `Multiline String`, `Formatted String`, `Real Time Dynamic Choice`, `Button`
FILE	No Required for prompt type `Lookup`	String	This creates a menu or list that is populated by the values in a specified file.

			The file attribute contains a filename that should be stored a known location by the integrator. Each line of text is displayed as an item in a list.
FORMAT	No Required for prompt type `Formatted String`	String	This creates a prompt window that accepts one or more characters, provided they are in the predefined format. The attribute contains a regular expression that the user must match for the value to be accepted.
FIXEDCHOICE	No Required for prompt type **Fixed Choice**	String	This creates a menu this is populated with options that are specified. Options are separated by a pipe |. If left blank the integrator can specify the options locally.

POX

```
<PROMPT>
    <UNIQUENAME></UNIQUENAME>
    <ICON></ICON>
    <PROMPTTEXTID></PROMPTTEXTID>
    <PROMPTTEXT></PROMPTTEXT>
    <DEFAULTVALUE></DEFAULTVALUE>
    <ORDER>0</ORDER>
    <ERRORMESSAGEID></ERRORMESSAGEID>
    <ERRORMESSAGE></ERRORMESSAGE>
    <TYPE></TYPE>
    <FILE></FILE>
    <FORMAT></FORMAT>
    <FIXEDCHOICE></FIXEDCHOICE>
</PROMPT>
```

Json


```
{
   "UNIQUENAME":"",
   "ICON":"",
   "PROMPTTEXTID":"",
   "PROMPTTEXT":"",
   "DEFAULTVALUE":"",
   "ORDER":0,
   "ERRORMESSAGEID":"",
   "ERRORMESSAGE":"",
   "TYPE":"",
   "FILE":"",
   "FORMAT":"",
   "FIXEDCHOICE":""
}
```

4. API Methods

All API Methods requests require the following mandatory fields and objects to be supplied in each request.

Name	Attribute/Object
TYPE	Attribute
MODE	Attribute
TERMINALID	Attribute
AUTHORIZATION	Object
SHOPID	Attribute
TXID	Attribute
LOCALDATETIME	Attribute

All API methods will respond with the following mandatory fields.

Name	Attribute/Object
SERVERDATETIME	Attribute
TERMINALID	Attribute
RESULT	Object

4.1. Sale

The purchase of products and services by end customers is handled through the `SALE` method. The integrator can perform a sale against a merchant account, check if a product/service can be sold through a merchant account, as well as reserve and capture a product like a pre-authorization and confirmation request.

Modes

The `SALE` method contains several modes which perform certain functions on a merchant account for a product or service.

Mode	Financial	Description
CHECK	Depends on service	The `CHECK` mode allows an integrator to determine if a product or service can be sold through a merchant account, if the supplier has a check method then it will be called by the API as well. Some product and services use the `CHECK` method for eligibility using the customer identifiers such as mobile or card details, this therefore can result in a financial record being created on a merchant account
DIRECT	Yes	The `DIRECT` mode means that the financial request is performed for the product against the supplier, this can either be obtaining a PIN for a PIN-based product or activating of a card and adding value to an account linked to the card. This method will result in a financial record being created if the result is successful
RESERVE	No	The `RESERVE` mode allows the integrator to "book" a product or service with the intention to confirm it later. If a confirmation is not sent an integrator should expect any action to be rolled back. Not all suppliers support the `RESERVE` mode and therefore this method would result in the same process of a `CHECK` mode. This method will not result in a financial record being created if the result is successful
CAPTURE	Yes	The `CAPTURE` mode allows the integrator to confirm the purchase of a product or service by confirming an original reservation. A `CAPTURE` cannot be sent without a `RESERVE` being performed, as the `CAPTURE` requires the original `TXID` OR `HOSTTXID` of the reservation. This method will result in a financial record being created if the result is successful
REPRINT	No	Reprints the merchant copy of an original `SALE`

POX

```
<REQUEST TYPE="SALE" MODE="CHECK">
```

```
<REQUEST TYPE="SALE" MODE="DIRECT">
```

```
<REQUEST TYPE="SALE" MODE="RESERVE">
```

```
<REQUEST TYPE="SALE" MODE="CAPTURE">
```

```
<REQUEST TYPE="SALE" MODE="REPRINT">
```

Json

```
{
    "TYPE": "SALE",
    "MODE": "CHECK"
}
```

```
{
    "TYPE": "SALE",
    "MODE": "DIRECT"
}
```

```
{
    "TYPE": "SALE",
    "MODE": "RESERVE"
}
```

```
{
    "TYPE": "SALE",
    "MODE": "CAPTURE"
}
```

```
{
    "TYPE": "SALE",
    "MODE": "REPRINT"
}
```

4.2. Cancel

The cancellation of previous requests is handled through the `CANCEL` methods. An integrator can perform a cancellation of any request apart from the diagnostic method.

Modes

The `CANCEL` method contains several modes which perform certain functions on a merchant account for a product or service.

Mode	Financial	Description
MANUAL	Yes	The `MANUAL` mode is used when a customer has requested a previous transaction to be refunded. This method will result in a financial record being created if the result is successful
AUTOMATIC	Yes	The `AUTOMATIC` mode is used when a system fault has occurred or the API has not responded within the timeout window set by the integrator This method should be automatically performed by the integrator without ANY user intervention System faults can include, timeouts, paper jams, device faults, and service unavailable result codes returned by the API This method will result in a financial record being created if the result is successful

Cancelations for `MANUAL` and `AUTOMATIC` can either use the original `TXID` or the `HOSTTXID` from the response. It's recommended that the `HOSTTXID` is used in a `MANUAL` mode request, and the original `TXID` in the `AUTOMATIC` mode request.

When using the `TYPE` as `CLIENT` the integrator should specific the original `TXID`, when using the `TYPE` as `HOST` the integrator should use the `HOSTTXID` from the original response.

Examples

```
<TXREF TYPE="HOST">EP9110002000077743</TXREF>
```

```
<TXREF TYPE="CLIENT">4FGDFAU4503243834</TXREF>
```

POX

```
<REQUEST TYPE="CANCEL" MODE="MANUAL">
```

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
```

Json

```
{
    "TYPE": "CANCEL",
    "MODE": "MANUAL"
}
```

```
{
    "TYPE": "CANCEL",
    "MODE": "AUTOMATIC"
}
```

4.3. Redeem

The redemption of coupons, payouts and other services where value is given back to the customer is handled through the `REDEEM` method. The integrator can perform a redemption, check if a customer is eligible for being rewarded a value as well as reserve and capture the redemption like a pre-authorization and confirmation request.

As an example, the redeem method can be used for the following:

- Redemption of a coupon
- Redemption of a lottery ticket
- Redemption of value from a gift card
- Redemption of value from a money transfer account

Modes

The `REDEEM` method contains several modes which perform certain functions on a merchant account for a product or service.

	Financial	Description
CHECK	Depends on service	The `CHECK` mode allows an integrator to determine if a redemption is valid and can be redeemed, if the supplier has a check method then it will be called by the API as well Some product and services use the `CHECK` method for eligibility such as a coupon validity or lottery ticket, this therefore can result in a financial record being created on a merchant account
DIRECT	Yes	The `DIRECT` mode means that the financial request is performed for the redemption against the supplier, this can be used to obtain value for a winning lottery ticket or a coupon offer This method will result in a financial record being created if the result is successful
RESERVE	No	The `RESERVE` mode allows the integrator to "book" the redemption with the intention to confirm it later. If a confirmation is not sent the integrator should expect any action to be rolled back Not all suppliers support the `RESERVE` mode and therefore this method would result in the same process of a `CHECK` mode This method will not result in a financial record being created if the result is successful
CAPTURE	Yes	The `CAPTURE` mode allows the integrator to confirm the redemption by confirming an original reservation A `CAPTURE` cannot be sent without a `RESERVE` being performed, as the `CAPTURE` requires the original `TXID` OR `HOSTTXID` of the reservation This method will result in a financial record being created if the result is successful
REPRINT	No	Reprint the last merchant copy receipt of a successful request

POX

```
<REQUEST TYPE="REDEEM" MODE="CHECK">
```

```
<REQUEST TYPE="REDEEM" MODE="DIRECT">
```

```
<REQUEST TYPE="REDEEM" MODE="RESERVE">
```

```
<REQUEST TYPE="REDEEM" MODE="CAPTURE">
```

```
<REQUEST TYPE="REDEEM" MODE="REPRINT">
```

Json

```
{
    "TYPE": "REDEEM",
    "MODE": "CHECK"
}
```

```
{
    "TYPE": "REDEEM",
    "MODE": "DIRECT"
}
```

```
{
    "TYPE": "REDEEM",
    "MODE": "RESERVE"
}
```

```
{
    "TYPE": "REDEEM",
    "MODE": "CAPTURE"
}
```

```
{
    "TYPE": "REDEEM",
    "MODE": "REPRINT"
}
```

4.4. Refund

The reloading of funds to account such as a gift card or money transfer account is handled through the refund method. The refund is not to be mistaken for the refund of a purchase back to the customer (see CANCEL method). The integrator can perform a refund to add value, check if the refund can be performed, as well as reserve and capture the refund like a pre-authorization and confirm request.

Modes

The REFUND method contains several modes which perform certain functions on a merchant account for a product or service.

Mode	Financial	Description
CHECK	Depends on service	The CHECK mode allows the integrator to determine if a REFUND can be performed on a merchant account, if the supplier has a CHECK method then it will be called by the API as well with the details of the customer account. Some product and services use the CHECK method for eligibility, this therefore can result in a financial record being created on a merchant account
DIRECT	Yes	The DIRECT mode means that the financial request is performed for the REFUND against the supplier, this can be used to load funds onto the customer's account for product and services such as reloadable gift cards and money transfer accounts. This method will result in a financial record being created if the result is successful
RESERVE	No	The RESERVE mode allows the integrator to "book" the REFUND with the intention to confirm it later. If a confirmation is not sent the integrator should expect any action to be rolled back. Not all suppliers support the RESERVE mode and therefore this method would result in the same process of a CHECK mode. This method will not result in a financial record being created if the result is successful
CAPTURE	Yes	The CAPTURE mode allows the integrator to confirm the REFUND by confirming an original reservation. A CAPTURE cannot be sent without a RESERVE being performed, as the CAPTURE requires the original TXID OR HOSTTXID of the reservation. This method will result in a financial record being created if the result is successful.
REPRINT	No	Reprint the merchant copy of a previous successful request

POX

```
<REQUEST TYPE="REFUND" MODE="CHECK">
```

```
<REQUEST TYPE="REFUND" MODE="DIRECT">
```

```
<REQUEST TYPE="REFUND" MODE="RESERVE">
```

```
<REQUEST TYPE="REFUND" MODE="CAPTURE">
```

```
<REQUEST TYPE="REFUND" MODE="REPRINT">
```

Json

```
{
    "TYPE": "REFUND",
    "MODE": "CHECK"
}
```

```
{
    "TYPE": "REFUND",
    "MODE": "DIRECT"
}
```

```
{
    "TYPE": "REFUND",
    "MODE": "RESERVE"
}
```

```
{
    "TYPE": "REFUND",
    "MODE": "CAPTURE"
}
```

```
{
    "TYPE": "REFUND",
    "MODE": "REPRINT"
}
```

4.5. Status

The enquiry for the balance of an account for product or service is handled through the status method.

Modes

The `STATUS` method contains only one mode for acquiring the balance object.

Mode	Financial	Description
DIRECT	Depends on service	The `DIRECT` mode allows the integrator to obtain the `BALANCE` for a product or service using the customer account details such as gift card or money transfer account. Not all suppliers support a balance enquiry method therefore the result against one of these suppliers would be feature unavailable. Please be advised that a financial record could be created for a balance enquiry if there is a charge applied for this and the result of the request is successful.

POX

```
<REQUEST TYPE="STATUS" MODE="DIRECT">
```

Json

```
{
    "TYPE": "STATUS",
    "MODE": "DIRECT"
}
```

4.6. Product Info

The request to obtain product information such as the redemption URL is handled through the product info method, please refer to the product configuration documentation for which products support this feature.

Modes

The `PRODUCTINFO` method contains only one mode for acquiring the details of a product.

Mode	Financial	Description
DIRECT	No	The `DIRECT` mode allows the integrator to obtain the details for a product, often this method is used for obtaining a new URL for redemption a new code. This method will not result in a financial record being created if the result is successful.

POX

```
<REQUEST TYPE="PRODUCTINFO" MODE="DIRECT">
```

Json

```
{
    "TYPE": "PRODUCTINFO",
    "MODE": "DIRECT"
}
```

4.7. Diagnostic

A Diagnostic request allows the integrator to check if the API is running and responding without requiring a transaction to occur, this is only to check if the API is available, this will not perform a check against a supplier.

POX - Request

```
<REQUEST TYPE="DIAGNOSTIC" />
```

POX - Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE TYPE="DIAGNOSTIC">
    <SERVERDATETIME>2017-01-25 15:38:08</SERVERDATETIME>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
</RESPONSE>
```

5. Services

The following sections describe the generic types of product and services epay support and offer, whilst in the retail market products and services can be categorized differently epay have tried to provide the same generic functionality for them. Please refer to the table below for the mapping of categories to API functionality, the product configuration documents provided will also detail what type of API service is used.

Service	Categories
PIN	Mobile e-vouchers Code to content (C2C) Smart meter codes Electronic Software distribution (ESD)
ETU	Mobile Top-up Point of Sale Activation (POSA) Pre-paid Debit Cards Smart Meter Account Top-up Stored Value Gift Card
COUPON	Couponing Consumer Offers Vouchers Scratch cards
GIFTCARD	Closed loop gift card Open loop gift card
LOTTERY	Lottery
TOLLING	Road Tolling
SUBSCRIPTION	Know your customer (KYC) SIM Card Registration
ADVERTISEMENT	Advertisements Offers
BILLING	Invoice/Notice Payment
WALLET (Pays)	Pay by Wallet Pay by Mobile Application
PARCEL	Parcel Delivery Parcel Collection

5.1. PIN Service

PIN-based product or services are used to purchase products such as mobile airtime, long distance call time and electronic money. When a PIN product is purchased, a receipt is printed detailing a unique PIN and along with instructions on how to redeem the PIN.

For PIN product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields
SALE	CHECK	Check if integrator can purchase a product or service using the merchant account and a product identifier.	`PRODUCTID` `AMOUNT`
SALE	DIRECT	Purchase a PIN-Based product or service using a merchant account and a product identifier	`PRODUCTID` `AMOUNT`
SALE	RESERVE	Reserve a PIN-Based product or service using a merchant account and a product identifier. If the supplier does not support the reservation method a `CHECK` will be performed instead.	`PRODUCTID` `AMOUNT`
SALE	CAPTURE	Confirm the purchase of a PIN-Based product or service using the original `TxID` of a previous `RESERVE` request. If the supplier does not support the reservation method a `DIRECT` will be performed instead.	`TXREF` `TXREF TYPE=””`
SALE	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`
CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`
PRODUCTINFO	DIRECT	Obtain product details for a product or service that requires a URL to be used for redeeming the PIN.	`PRODUCTID` `AMOUNT`

Examples

SALE CHECK example - REQUEST

```
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>2</TXID>
    <PRODUCTID>UKPIN10300001</PRODUCTID>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

SALE CHECK example - RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>2</TXID>
    <HOSTTXID>EP9110002000077735</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
</RESPONSE>
```

SALE DIRECT example - REQUEST

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>3</TXID>
    <PRODUCTID>UKPIN10300001</PRODUCTID>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
</REQUEST>
```

SALE DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>3</TXID>
    <HOSTTXID>EP9110002000077734</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
    <PINCREDENTIALS>
        <PIN>898757550230217155706156</PIN>
        <SERIAL>0000000000000000403904</SERIAL>
        <VALIDTO>2017-02-06</VALIDTO>
    </PINCREDENTIALS>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

SALE RESERVE example - REQUEST

```
<REQUEST TYPE="SALE" MODE="RESERVE">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-1</TXID>
    <PRODUCTID>UKPIN10300001</PRODUCTID>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
</REQUEST>
```

SALE RESERVE example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="RESERVE" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:32:23</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-1</TXID>
    <HOSTTXID>EP9110002000077736</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
</RESPONSE>
```

SALE CAPTURE example - REQUEST

```
<REQUEST TYPE="SALE" MODE="CAPTURE">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-2</TXID>
    <PRODUCTID>UKPIN10300001</PRODUCTID>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="HOST">EP9110002000077736</TXREF>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
</REQUEST>
```

SALE CAPTURE example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CAPTURE" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:33:44</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-2</TXID>
    <HOSTTXID>EP9110002000077737</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>100</AMOUNT>
    <PINCREDENTIALS>
        <PIN>898757550230217155706156</PIN>
        <SERIAL>0000000000000000403906</SERIAL>
        <VALIDTO>2017-02-06</VALIDTO>
    </PINCREDENTIALS>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
    <ADVERTISEMENT>
        <ADVERT>
            <LINE>**************************</LINE>
            <LINE>         OFFER            </LINE>
            <LINE>**************************</LINE>
        </ADVERT>
    </ADVERTISEMENT>
</RESPONSE>
```

SALE REDIRECT example – REQUEST

```
<REQUEST TYPE="SALE" MODE="REPRINT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>3</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="HOST">EP9110002000077743</TXREF>
</REQUEST>
```

SALE REDIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="REPRINT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:45:35</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>3</TXID>
    <HOSTTXID>EP9110002000077744</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
    </RECEIPT>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-3</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="CLIENT">3</TXREF>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:36:26</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-3</TXID>
    <HOSTTXID>EP9110002000077738</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
    <PINCREDENTIALS>
        <SERIAL>0000000000000403906</SERIAL>
    </PINCREDENTIALS>
</RESPONSE>
```

CANCEL MANUAL example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="MANUAL">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>7</TXID>
    <PRODUCTID>UKPIN10300001</PRODUCTID>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="HOST">EP9110002000077739</TXREF>
</REQUEST>
```

CANCEL MANUAL example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="MANUAL" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:41:25</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>7</TXID>
    <HOSTTXID>EP9110002000077740</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
    <PINCREDENTIALS>
        <SERIAL>0000000000000000403909</SERIAL>
    </PINCREDENTIALS>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.2. ETU Service

ETU-based product or services are used to purchase products such as POSA, mobile airtime or meter credit. At the time of purchase the customer should present a card that can be swiped, scanned or keyed into the point of sale, the customer can then either choose a value to be applied or a fixed value related to the card IIN is selected. The customer should receive a receipt detailing the card (potentially in a masked format) and the value that has been applied. In addition, the receipt will detail any further instructions or codes that are to be used.

For ETU product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields

SALE	CHECK	Check if integrator can purchase a product or service using the merchant account and a product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

SALE	DIRECT	Purchase a ETU-Based product or service using a merchant account, product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

SALE	RESERVE	Reserve a ETU-Based product service using a merchant account, product identifier and `CARD`.
		
		If the supplier does not support the reservation method a `CHECK` will be performed instead.
			`PRODUCTID`
			`AMOUNT`
			`CARD`

SALE	CAPTURE	Confirm the purchase of a ETU-Based product or service using the original `TxID` of a previous `RESERVE` request. If the supplier does not support the reservation method a `DIRECT` will be performed instead.	`TXREF` `TXREF TYPE=””`
SALE	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`

CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user, the original `CARD` must be swiped.	`TXREF`
			`TXREF TYPE=””`
			`CARD`

CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system. The original `CARD` details must be provided.	`TXREF`
			`TXREF TYPE=””`
			`CARD`

The account reference or card top-up reference should be supplied in the CARD object which is a mandatory field for these types of product or services. If card is swiped then an integrator should supply all fields of the CARD object, if the card is scanned (barcode) then the full barcode should be supplied in the PAN field.

If a user scans a EAN13/UPC at the start of a request this should be supplied in the EAN field within the CARD object.

Examples

SALE CHECK example – REQUEST

```
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 13:29:03</LOCALDATETIME>
    <TXID>1</TXID>
    <PRODUCTID>UKETU10310002</PRODUCTID>
    <AMOUNT>1500</AMOUNT>
    <SHOPID>123456</SHOPID>
    <CARD>
        <PAN>63386396111111111119</PAN>
    </CARD>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

SALE CHECK example - RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>1</TXID>
    <HOSTTXID>EP9110002000077735</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>1500</AMOUNT>
</RESPONSE>
```

SALE DIRECT example – REQUEST

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>3</TXID>
    <PRODUCTID>UKETU10310002</PRODUCTID>
    <AMOUNT>1500</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>6338639611111111119</PAN>
    </CARD>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
</REQUEST>
```

SALE DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 13:18:01</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>3</TXID>
    <HOSTTXID>EP9110003000150244</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>1500</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-4</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>63386396111111111119</PAN>
    </CARD>
    <TXREF TYPE="CLIENT">UKAPIREFERENCE1-3</TXREF>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:36:26</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-4</TXID>
    <HOSTTXID>EP9110002000077738</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>1500</AMOUNT>
</RESPONSE>
```

CANCEL MANUAL example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="MANUAL">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UKAPIREFERENCE1-5</TXID>
    <PRODUCTID>UKETU10310002</PRODUCTID>
    <AMOUNT>1500</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>6338639611111111119</PAN>
    </CARD>
    <TXREF TYPE="HOST">EP9110003000150251</TXREF>
</REQUEST>
```

CANCEL MANUAL example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="MANUAL" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:41:25</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UKAPIREFERENCE1-5</TXID>
    <HOSTTXID>EP9110002000077740</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>1500</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.3. Coupon Service

Coupon product or services relates to the customer using either a paper or digital based code that entitles them to a reward either monetary or loyalty based. Most coupons are used when a customer is purchasing goods where the coupon relates to one or more products in the basket.

For Coupon product or services an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields
REDEEM	CHECK	Check if integrator can redeem a coupon, and obtain `OFFER`, `VALIDITY`, `DISCOUNT` and `SKU` details to validate `BASKET` against.	`COUPONS` `BASKET`
REDEEM	DIRECT	Redeems a coupon, `BASKET` data will be used to validate if the coupon can be redeemed.	`COUPONS` `BASKET`
REDEEM	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`

CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.
		
		Please note that **not** all coupon-providers support the capabilities of a customer based refund for a coupon.
			`TXREF`
			`TXREF TYPE=””`
			`COUPONS`

CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF`
			`TXREF TYPE=””`
			`COUPONS`

In each request the `PRODUCTID` field should be set to **COUPON**, it is recommended that the integrator supply the full basket in each `REDEEM:DIRECT` request, or use the `REDEEM:CHECK` to obtain the `SKU` list that is required to be cross-referenced, once the integrator has cross-referenced the basket, the integrator must supply a shortened basket based on the `SKU` list.

The `AMOUNT` field is **not** used within the responses and should **not** be used as the amount to deduct from a `BASKET` within the integrator's system, please refer to the `DISCOUNT` object which is part of the response for each `COUPON`.

The coupon service will require the usage of the objects described in sections **3.11** to **3.16.**

Examples

REDEEM CHECK example – REQUEST

```
<REQUEST TYPE="REDEEM" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2017-01-23 12:40:23</LOCALDATETIME>
    <TXID>TEST1</TXID>
    <CASHIER>TESTCASHIER</CASHIER>
    <PRODUCTID>COUPON</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <CODE>13434393438121216483</CODE>
            <ENTRYTYPE>SCANNED</ENTRYTYPE>
        </COUPON>
    </COUPONS>
</REQUEST>
```

REDEEM CHECK example – response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="REDEEM">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2017-01-24 11:12:18</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST1</TXID>
    <HOSTTXID>EP9110007000000024</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <RESULT>0</RESULT>
            <CODE>13434393438121216483</CODE>
            <VALIDITY>
                <ACTIVE>TRUE</ACTIVE>
                <STARTDATE>2017-01-01 00:00:00</STARTDATE>
                <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
                <QUANTITY>1</QUANTITY>
                <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
                <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
            </VALIDITY>
            <OFFER>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
                <CATEGORY>BOGOFF</CATEGORY>
                <ID>OFFER1TEST</ID>
                <PROMOTIONCODE></PROMOTIONCODE>
                <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
                <DISCOUNT>
                    <TYPE>UNIT</TYPE>
                    <AMOUNTOFF>0</AMOUNTOFF>
                    <PERCENTOFF>0</PERCENTOFF>
                    <UNITOFF>1</UNITOFF>
                </DISCOUNT>
                <BRAND>Coca Cola</BRAND>
                <SKUS>
                    <SKU>
                        <ID>COKESKU</ID>
                        <SKUVALUE>05017726109448</SKUVALUE>
                        <QUANTITY>1</QUANTITY>
                        <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                        <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
                    </SKU>
                </SKUS>
            </OFFER>
        </COUPON>
    </COUPONS>
</RESPONSE>
```

REDEEM DIRECT example – REQUEST

```
<REQUEST TYPE="REDEEM" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2017-01-23 12:40:23</LOCALDATETIME>
    <TXID>TEST1</TXID>
    <CASHIER>TESTCASHIER</CASHIER>
    <PRODUCTID>COUPON</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <BASKET>
        <ID>TESTBASKET</ID>
        <QUANTITY>1</QUANTITY>
        <ITEMS>
            <ITEM>
                <ID>1</ID>
                <QUANTITY>1</QUANTITY>
                <SKU>9555589203096</SKU>
                <COST>100</COST>
                <VAT>0</VAT>
                <VATAPPLIED>FALSE</VATAPPLIED>
                <VATRATE>0.0</VATRATE>
                <CATALOGID>9555589203096</CATALOGID>
            </ITEM>
        </ITEMS>
    </BASKET>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <CODE>13434393438121216483</CODE>
            <ENTRYTYPE>SCANNED</ENTRYTYPE>
        </COUPON>
    </COUPONS>
</REQUEST>
```

REDEEM DIRECT example – response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="REDEEM">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2017-01-24 11:12:18</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST1</TXID>
    <HOSTTXID>EP9110007000000024</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <RESULT>0</RESULT>
            <CODE>13434393438121216483</CODE>
            <VALIDITY>
                <ACTIVE>TRUE</ACTIVE>
                <STARTDATE>2017-01-01 00:00:00</STARTDATE>
                <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
                <QUANTITY>1</QUANTITY>
                <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
                <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
            </VALIDITY>
            <OFFER>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
                <CATEGORY>BOGOFF</CATEGORY>
                <ID>OFFER1TEST</ID>
                <PROMOTIONCODE></PROMOTIONCODE>
                <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
                <DISCOUNT>
                    <TYPE>UNIT</TYPE>
                    <AMOUNTOFF>0</AMOUNTOFF>
                    <PERCENTOFF>0</PERCENTOFF>
                    <UNITOFF>1</UNITOFF>
                </DISCOUNT>
                <BRAND>Coca Cola</BRAND>
                <SKUS>
                    <SKU>
                        <ID>COKESKU</ID>
                        <SKUVALUE>05017726109448</SKUVALUE>
                        <QUANTITY>1</QUANTITY>
                        <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                        <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
                    </SKU>
                </SKUS>
            </OFFER>
        </COUPON>
    </COUPONS>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-5</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="CLIENT">UKAPIREFERENCE1-3</TXREF>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <CODE>13434393438121216483</CODE>
            <ENTRYTYPE>SCANNED</ENTRYTYPE>
        </COUPON>
    </COUPONS>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:36:26</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-4</TXID>
    <HOSTTXID>EP9110002000077738</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <RESULT>0</RESULT>
            <CODE>134343934381212164 83</CODE>
            <VALIDITY>
                <ACTIVE>TRUE</ACTIVE>
                <STARTDATE>2017-01-01 00:00:00</STARTDATE>
                <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
                <QUANTITY>1</QUANTITY>
                <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
                <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
            </VALIDITY>
            <OFFER>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
                <CATEGORY>BOGOFF</CATEGORY>
                <ID>OFFER1TEST</ID>
                <PROMOTIONCODE></PROMOTIONCODE>
                <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
                <DISCOUNT>
                    <TYPE>UNIT</TYPE>
                    <AMOUNTOFF>0</AMOUNTOFF>
                    <PERCENTOFF>0</PERCENTOFF>
                    <UNITOFF>1</UNITOFF>
                </DISCOUNT>
                <BRAND>Coca Cola</BRAND>
                <SKUS>
                    <SKU>
                        <ID>COKESKU</ID>
                        <SKUVALUE>05017726109448</SKUVALUE>
                        <QUANTITY>1</QUANTITY>
                        <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                        <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
                    </SKU>
                </SKUS>
            </OFFER>
        </COUPON>
    </COUPONS>
</RESPONSE>
```

CANCEL MANUAL example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="MANUAL">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UPAPIREFERENCETEST1-6</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="HOST"></TXREF>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <CODE>13434393438121216483</CODE>
            <ENTRYTYPE>SCANNED</ENTRYTYPE>
        </COUPON>
    </COUPONS>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="MANUAL" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:36:26</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UPAPIREFERENCETEST1-6</TXID>
    <HOSTTXID>EP9110002000077739</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <COUPONS>
        <COUPON>
            <ID>TESTCOUPON</ID>
            <RESULT>0</RESULT>
            <CODE>13434393438121216483</CODE>
            <VALIDITY>
                <ACTIVE>TRUE</ACTIVE>
                <STARTDATE>2017-01-01 00:00:00</STARTDATE>
                <EXPIRATIONDATE>2017-01-31 23:59:59</EXPIRATIONDATE>
                <QUANTITY>1</QUANTITY>
                <REDEEMEDQUANTITY>0</REDEEMEDQUANTITY>
                <MINIMUMORDERVALUE>0</MINIMUMORDERVALUE>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
            </VALIDITY>
            <OFFER>
                <TYPE>DISCOUNT_VOUCHER</TYPE>
                <CATEGORY>BOGOFF</CATEGORY>
                <ID>OFFER1TEST</ID>
                <PROMOTIONCODE></PROMOTIONCODE>
                <DESCRIPTION>Buy 1 Coke get 1 free</DESCRIPTION>
                <DISCOUNT>
                    <TYPE>UNIT</TYPE>
                    <AMOUNTOFF>0</AMOUNTOFF>
                    <PERCENTOFF>0</PERCENTOFF>
                    <UNITOFF>1</UNITOFF>
                </DISCOUNT>
                <BRAND>Coca Cola</BRAND>
                <SKUS>
                    <SKU>
                        <ID>COKESKU</ID>
                        <SKUVALUE>05017726109448</SKUVALUE>
                        <QUANTITY>1</QUANTITY>
                        <DISCOUNTABLE>TRUE</DISCOUNTABLE>
                        <DESCRIPTION>Diet Coke 500ml</DESCRIPTION>
                    </SKU>
                </SKUS>
            </OFFER>
        </COUPON>
    </COUPONS>
</RESPONSE>
```

5.4. Gift Card Service

Gift card product or services are where a customer purchases a card that holds value at the time of purchase and can be redeemed later. Customers can also purchase gift card schemes where value can either be loaded or redeemed multiple times. Please be advised that there are product or services known as POSA which are not run through the gift card service however act as a gift card, these types of product or services are activated at the point of purchase; however, the redemption does not occur through our services.

For Gift card product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields

SALE	CHECK	Check if integrator can activate a Gift card product or service using a merchant account, product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

SALE	DIRECT	Purchase and load/activate a Gift card product or service using a merchant account, product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

REDEEM	RESERVE	Reserve the redemption of value from the Gift card product or service using a merchant account, product identifier and `CARD`.
		If the supplier does not support the reservation method a `CHECK` will be performed instead.
			`PRODUCTID`
			`AMOUNT`
			`CARD`

REDEEM	CAPTURE	Confirm the redemption of value from a Gift card product or service using the original `TxID` of a previous `RESERVE` request. If the supplier does not support the reservation method a `DIRECT` will be performed instead.	`TXREF` `TXREF TYPE=””`
REDEEM	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`

CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user, the original `CARD` must be swiped.	`TXREF`
			`TXREF TYPE=””`
			`CARD`

CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system. The original `CARD` details must be provided.	`TXREF`
			`TXREF TYPE=””`
			`CARD`

STATUS	DIRECT	Obtain balance of gift card product or service	`PRODUCTID` `CARD`

REFUND	CHECK	Check if integrator can LOAD funds onto a gift card product or service using a merchant account, product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

REFUND	DIRECT	Add funds onto a gift card product or service using a merchant account, product identifier and `CARD`.	`PRODUCTID`
			`AMOUNT`
			`CARD`

REFUND	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`

The account reference or card reference should be supplied in the CARD object which is a mandatory field for these types of product or services. If card is swiped then the integrator should supply all fields of the CARD object, if the card is scanned (barcode) then the full barcode should be supplied in the PAN field.

If a user scans a EAN13/UPC at the start of a request this should be supplied in the EAN field within the CARD object.

Examples

SALE DIRECT example – REQUEST

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <SHOPID>89012</SHOPID>
    <LOCALDATETIME>2011-08-09 15:49:07</LOCALDATETIME>
    <TXID>test221</TXID>
    <CASHIER>JChua</CASHIER>
    <PRODUCTID>NZGCD11260002</PRODUCTID>
    <CURRENCY>NZD</CURRENCY>
    <AMOUNT>15000</AMOUNT>
    <CARD>
        <PAN>60611500253239718888</PAN>
    </CARD>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

SALE DIRECT example – RESPONSE

```
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2011-08-09 05:49:07</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>test221</TXID>
    <HOSTTXID>EP9130004000000556</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>TRANSACTION SUCCESSFUL</RESULTTEXT>
    <CURRENCY>NZD</CURRENCY>
    <AMOUNT>15000</AMOUNT>
    <CARD>
        <PAN>6061150025323971888</PAN>
        <EXPIRY_YEAR>2012</EXPIRY_YEAR>
        <EXPIRY_MONTH>8</EXPIRY_MONTH>
    </CARD>
    <BALANCE>
        <CURRENCY STANDARD="1">
            <NAME>NZD</NAME>
            <FRACTION>2</FRACTION>
            <BALANCE>15000</BALANCE>
            <BALANCE_BEFORE>0</BALANCE_BEFORE>
            <RESERVED>0</RESERVED>
            <RESERVED_BEFORE>0</RESERVED_BEFORE>
        </CURRENCY>
    </BALANCE>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

REDEEM DIRECT example – REQUEST

```
<REQUEST TYPE="REDEEM" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <SHOPID>89012</SHOPID>
    <LOCALDATETIME>2011-08-09 16:32:14</LOCALDATETIME>
    <TXID>test225</TXID>
    <CASHIER>JChua</CASHIER>
    <PRODUCTID>NZGCD11260008</PRODUCTID>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>10000</AMOUNT>
    <CARD>
        <PAN>60611500253239718888</PAN>
    </CARD>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

REDEEM DIRECT example – RESPONSE

```
<RESPONSE MODE="DIRECT" TYPE="REDEEM">
    <SERVERDATETIME>2011-08-09 06:32:14</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>test225</TXID>
    <HOSTTXID>EP9130004000000561</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>TRANSACTION SUCCESSFUL</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>10000</AMOUNT>
    <CARD>
        <PAN>6061150025323971888</PAN>
        <EXPIRY_YEAR>2012</EXPIRY_YEAR>
        <EXPIRY_MONTH>8</EXPIRY_MONTH>
    </CARD>
    <BALANCE>
        <CURRENCY STANDARD="1">
            <NAME>NZD</NAME>
            <FRACTION>2</FRACTION>
            <BALANCE>5000</BALANCE>
            <BALANCE_BEFORE>15000</BALANCE_BEFORE>
            <RESERVED>0</RESERVED>
            <RESERVED_BEFORE>0</RESERVED_BEFORE>
        </CURRENCY>
    </BALANCE>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

STATUS DIRECT example – REQUEST

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="STATUS" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <SHOPID>89012</SHOPID>
    <LOCALDATETIME>2011-08-26 16:55:10</LOCALDATETIME>
    <TXID>TEST511</TXID>
    <CASHIER>JChua</CASHIER>
    <PRODUCTID>NZGCD11260007</PRODUCTID>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>0</AMOUNT>
    <CARD>
        <PAN>60611500220637846888</PAN>
    </CARD>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

STATUS DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="STATUS">
    <SERVERDATETIME>2011-08-26 06:55:10</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST511</TXID>
    <HOSTTXID>EP9130004000000831</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>TRANSACTION SUCCESSFUL</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>0</AMOUNT>
    <CARD>
        <PAN>60611500220637846888</PAN>
        <EXPIRY_YEAR>2012</EXPIRY_YEAR>
        <EXPIRY_MONTH>8</EXPIRY_MONTH>
    </CARD>
    <BALANCE>
        <CURRENCY STANDARD="1">
            <NAME>NZD</NAME>
            <FRACTION>2</FRACTION>
            <BALANCE>10000</BALANCE>
            <BALANCE_BEFORE>10000</BALANCE_BEFORE>
            <RESERVED>2000</RESERVED>
            <RESERVED_BEFORE>2000</RESERVED_BEFORE>
        </CURRENCY>
    </BALANCE>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <TILLID>2</TILLID>
    <LOCALDATETIME>2011-08-05 11:00:54</LOCALDATETIME>
    <TXID>test216</TXID>
    <CASHIER>JChua</CASHIER>
    <PRODUCTID>NZGCD11260008</PRODUCTID>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>0</AMOUNT>
    <CARD>
        <PAN>60611500212864066624</PAN>
    </CARD>
    <TXREF TYPE="HOST">EP9130004000000430</TXREF>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <SERVERDATETIME>2011-08-05 01:00:54</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>test216</TXID>
    <HOSTTXID>EP9130004000000431</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>TRANSACTION SUCCESSFUL</RESULTTEXT>
    <CURRENCY>GBP</CURRENCY>
    <AMOUNT>2000</AMOUNT>
    <CARD>
        <PAN>60611500212864066624</PAN>
        <EXPIRY_YEAR>2012</EXPIRY_YEAR>
        <EXPIRY_MONTH>8</EXPIRY_MONTH>
    </CARD>
    <BALANCE>
        <CURRENCY STANDARD="1">
            <NAME>NZD</NAME>
            <FRACTION>2</FRACTION>
            <BALANCE>8000</BALANCE>
            <BALANCE_BEFORE>8000</BALANCE_BEFORE>
            <RESERVED>0</RESERVED>
            <RESERVED_BEFORE>2000</RESERVED_BEFORE>
        </CURRENCY>
    </BALANCE>
</RESPONSE>
```

5.5. Lottery Service

Lottery product or services are where a customer can purchase a ticket for the entry into a lottery to potentially win a prize. A customer may choose selected numbers through multiple lines, perform quick picks or use a registered card to play their same numbers. Not all lottery products are the same and the number ranges and days can change, please refer to the PCD for the configuration of lottery products.

For lottery product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields

SALE	CHECK	Check if integrator can play a lottery product or service using a merchant account, product identifier and `LOTTERY`.	`PRODUCTID`
			`AMOUNT`
			`LOTTERY`

SALE	DIRECT	Play a lottery product or service using a merchant account, product identifier and `LOTTERY`.	`PRODUCTID`
			`AMOUNT`
			`LOTTERY`

REDEEM	CHECK	Check if a lottery ticket has won a prize, if a lottery ticket ID is not performed then a result receipt will be returned instead Not all lottery providers support the REDEEM CHECK, therefore a FEATURE UNAVAILABLE error will be returned.	`PRODUCTID` `LOTTERY`
REDEEM	DIRECT	Payout a lottery ticket that may have won a prize, will perform the same as CHECK however the if there is a winning value it will be credited to your account and therefore required for the customer to be paid from your till.	`PRODUCTID` `LOTTERY`
CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`

If a user is registering a new CARD with selected numbers the integrator must put the account reference from the card in the `PAN` field and supply the selected numbers in the `NUMBERS` field. If a user is wanting to play with a card, the integrator must only supply the account reference and leave the `NUMBERS` field blank.

Not all lottery products use registered cards therefore please check the PCD for more information.

Examples

SALE DIRECT example – REQUEST

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UKLOTTERYTEST1</TXID>
    <PRODUCTID>UKLOT11410001</PRODUCTID>
    <AMOUNT>100</AMOUNT>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <LOTTERY>
       <DAYS>8192</DAYS>
       <GAMES>1</GAMES>
       <NUMBERS>0102030405</NUMBERS>
    </LOTTERY>
</REQUEST>
```

SALE DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 13:18:01</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UKLOTTERYTEST1</TXID>
    <HOSTTXID>EP9110003000150300</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

REDEEM DIRECT example – REQUEST

```
<REQUEST TYPE="REDEEM" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UKLOTTERYTEST7</TXID>
    <PRODUCTID>UKLOT11410001</PRODUCTID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <LOTTERY>
      <TICKETID>8545472342409234</TICKETID>
    </LOTTERY>
</REQUEST>
```

REDEEM DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="REDEEM">
    <SERVERDATETIME>2017-02-06 13:18:01</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UKLOTTERYTEST7</TXID>
    <HOSTTXID>EP9110003000150500</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>1000</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UKLOTTERYTEST2</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="CLIENT">UKLOTTERYTEST1</TXREF>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:36:26</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>UKLOTTERYTEST2</TXID>
    <HOSTTXID>EP9110003000150301</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
</RESPONSE>
```

CANCEL MANUAL example – REQUEST

```
<REQUEST TYPE="CANCEL" MODE="MANUAL">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2013-06-25 15:05:29</LOCALDATETIME>
    <TXID>UKLOTTERYTEST3</TXID>
    <SHOPID>123456</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TXREF TYPE="HOST">EP9110003000150300</TXREF>
</REQUEST>
```

CANCEL MANUAL example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="MANUAL" TYPE="CANCEL">
    <SERVERDATETIME>2017-02-06 12:41:25</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID> UKLOTTERYTEST3</TXID>
    <HOSTTXID>EP9110003000150400</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>100</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.6. Tolling Service

Tolling product or services are where a customer can pay a charge for travelling across a toll road using their vehicle, and pay for any outstanding journeys made that were not paid for.

For tolling product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields
SALE	CHECK	Used for obtain details of vehicle and roads available for purchasing trips on, and obtaining any past trips that have yet to be paid for. The response will include any tariff relating to the roads.	`PRODUCTID` `TOLLING`

SALE	DIRECT	Purchase for a trip or past trip against a vehicle, the integrator must perform a `SALE:CHECK` to obtain tariffs and toll road details.	`PRODUCTID`
			`TOLLING`
			`AMOUNT`

CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`

5.6.1. Tolling Product Types

The following section describes generic tolling product types, these product types are driven by setting a `PRODUCTID`, please refer to the PCD.

5.6.1.1. Pay & Go Toll

The Pay & Go toll product type is used to pay for a toll fee in advance or to pay for a trip made in the past. The traveler/ integrator will specify the toll road when selecting this option, a list of toll road can be acquired from performing a `SALE:CHECK`.

The customer journey can change based on certain responses from the API, however generally the flow will be like as follows:

1. Customer Requests to Pay for a Toll through a specified tolling service
2. User enters the LPN into the system and a request is made to the API
3. The API will return a response containing all toll roads and any past trips
4. If past trips are identified the customer is asked if they want to pay for their past trips.

If the customer chooses to pay for their past trips the journey will be as follows:

1. The system will determine what toll roads have past trips assigned and list them to be selected by the customer
2. The customer will choose which toll road to pay for
3. The system will determine how many past trips there are outstanding and provide the ability for the customer to choose how many trips to pay for
4. Based on the above the system will calculate the total fee to be paid and the customer will be asked to confirm
5. If the customer agrees to make the payment the details of the number of trips, toll road and total fee are sent in a request to the API
6. In the response, back from the API a receipt will be provided detailing the transaction details, toll road information and number of trips including total fee
7. The system will print a customer and/or merchant receipt; the customer will be provided the customer copy.

If the customer has chosen to pay for all the trips, the integrator can use the `TRIPS.TOTAL` field to set the `AMOUNT` field, if the customer has chosen to pay for several trips the total amount can be calculated using the items returned in the `PASTTRIPS` field.

`AMOUNT` += (`PASTTRIP.ROAD.TOLLROADTARIFF` * `TRIPS`) + (`PASTTRIP.ROAD.SERVICEFEE`)

If the customer chooses not to pay for their past trips or only future trips are available, the journey will be as follows:

1. The system will determine what toll roads are available and list them to be selected by the customer
2. The customer will choose which toll road to pay for
3. The system will provide the ability for the customer to choose how many trips to pay for between a range of 1-10 (depending on which tolling product, this many change, please refer to PCD)
4. Based on the above the integrator will calculate the total fee to be paid and the customer will be asked to confirm
5. If the customer agrees to make the payment the details of the number of trips, toll road and total fee are sent in a request to the API
6. In the response, back from the API a receipt will be provided detailing the transaction details, toll road information and number of trips including total fee
7. The integrator will print a customer and/or merchant receipt; the customer will be provided the customer copy.

If the customer has chosen to pay for several trips the total amount can be calculated using the selected `ROAD` items.

`AMOUNT` += (`ROAD.TOLLROADTARIFF` * `TRIPS`) + (`ROAD.SERVICEFEE`)

5.6.1.2. Account Top-up

The account top-up product type is used to add credit to an account that the traveler has opened with the toll service provider. Toll charges are automatically deducted from the account as and when they occur.

The customer journey will be as follows:

1. Customer Requests to add funds to their account
2. User enters the LPN into the system and a request is made to the API
3. The API will return a response containing any relevant information to the request
4. The integrator will list all available top-up amounts and the customer will choose from one of these (please refer to PCD)
5. The user will confirm the amount and LPN with the customer
6. If the customer agrees the system will make a request to the API
7. In the response, back from the API a receipt will be provided detailing the transaction details, toll account information and including total amount
8. The system will print a customer and/or merchant receipt; the customer will be provided the customer copy.

5.6.1.3. Notice & Invoice Payment

The toll notice/invoice payment product type is used to pay an outstanding balance which is sent to the customer by the tolling service provider as a demand for payment of a toll charge/penalty. The notice/invoice will contain a barcode which is encoded with certain details of the account and total fee. Certain tolling service providers may have different barcode specifications, therefore please refer to the additional specifications on tolling barcodes.

The customer journey will be as follows:

1. Customer Requests to pay their notice/invoice payment for a specific tolling service
2. User will scan the barcode on the notice/invoice payment document, the barcode will be set in the `PAN` field
3. The integrator will perform a `SALE:CHECK` again the API which will decode the barcode and return the correct total amount and details.
4. The user will confirm with the customer the details and the total amount
5. The system will make a `SALE:DIRECT` request to the API with the original barcode and `AMOUNT` field
6. In the response, back from the API a receipt will be provided detailing the transaction details, toll account information and including total amounts
7. The system will print a customer and/or merchant receipt; the customer will be provided the customer copy.

Examples

SALE CHECK example – REQUEST – Pay & Go Toll

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480001</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
        </VEHICLE>
    </TOLLING>
</REQUEST>
```

SALE CHECK example – RESPONSE – Pay & Go Toll

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110002000077735</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
            <CLASS>Car</CLASS>
        </VEHICLE>
        <ROADS>
            <ROAD>
                <ID>3</ID>
                <CODE>TD</CODE>
                <DESCRIPTION>Takitimu Drive</DESCRIPTION>
                <TARIFF>180</TARIFF>
                <SERVICEFEE>120</SERVICEFEE>
            </ROAD>
        </ROADS>
    </TOLLING>
    <PASTTRIPS>
        <TRIP>
            <ROAD>
                <ID>3</ID>
                <CODE>TD</CODE>
                <DESCRIPTION>Takitimu Drive</DESCRIPTION>
                <TARIFF>160</TARIFF>
                <SERVICEFEE>120</SERVICEFEE>
            </ROAD>
            <TRIPS>1</TRIPS>
            <TOTAL>280</TOTAL>
            <TRIPDATE>2016-12-31 08:00:00</TRIPDATE>
        </TRIP>
    </PASTTRIPS>
</RESPONSE>
```

SALE DIRECT example – REQUEST – Pay & Go Toll – Future Trip

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST1</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480001</PRODUCTID>
    <SHOPID>1</SHOPID>
    <AMOUNT>300</AMOUNT>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
            <PURPOSEOFUSE>PRIVATE</PURPOSEOFUSE>
        </VEHICLE>
        <TRIPS>
            <TRIP>
                <ROAD>
                    <ID>3</ID>
                    <CODE>TD</CODE>
                    <DESCRIPTION>Takitimu Drive</DESCRIPTION>
                    <TARIFF>180</TARIFF>
                    <SERVICEFEE>120</SERVICEFEE>
                </ROAD>
                <TRIPS>1</TRIPS>
                <TOTAL>300</TOTAL>
            </TRIP>
        </TRIPS>
    </TOLLING>
</REQUEST>
```

SALE DIRECT example – RESPONSE – Pay & Go Toll – Future Trip

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST1</TXID>
    <HOSTTXID>EP9110002000077734</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>300</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

SALE DIRECT example – REQUEST – Pay & Go Toll – Past Trip

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST2</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480001</PRODUCTID>
    <SHOPID>1</SHOPID>
    <AMOUNT>280</AMOUNT>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
            <PURPOSEOFUSE>PRIVATE</PURPOSEOFUSE>
        </VEHICLE>
        <PASTTRIPS>
            <TRIP>
                <ROAD>
                    <ID>3</ID>
                    <CODE>TD</CODE>
                    <DESCRIPTION>Takitimu Drive</DESCRIPTION>
                    <TARIFF>160</TARIFF>
                    <SERVICEFEE>120</SERVICEFEE>
                </ROAD>
                <TRIPS>1</TRIPS>
                <TOTAL>280</TOTAL>
                <TRIPDATE>2016-12-31 08:00:00</TRIPDATE>
            </TRIP>
        </PASTTRIPS>
    </TOLLING>
</REQUEST>
```

SALE DIRECT example – RESPONSE – Pay & Go Toll – Past Trip

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST2</TXID>
    <HOSTTXID>EP9110002000077735</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>280</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

SALE CHECK example – REQUEST – Account Top-up

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480002</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
        </VEHICLE>
    </TOLLING>
</REQUEST>
```

SALE CHECK example – RESPONSE – Account Top-up

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110002000077736</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
            <CLASS>Car</CLASS>
        </VEHICLE>
    </TOLLING>
</RESPONSE>
```

SALE DIRECT example – REQUEST – Account Top-up

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST3</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480002</PRODUCTID>
    <SHOPID>1</SHOPID>
    <AMOUNT>2000</AMOUNT>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <TOLLING>
        <VEHICLE>
            <LPN>CR9AS1</LPN>
            <PURPOSEOFUSE>PRIVATE</PURPOSEOFUSE>
        </VEHICLE>
    </TOLLING>
</REQUEST>
```

SALE DIRECT example – RESPONSE – Account Top-up

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST3</TXID>
    <HOSTTXID>EP9110002000077737</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>2000</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

SALE CHECK example – REQUEST – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST3</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480003</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>93438734234782423742348723401</PAN>
    </CARD>
</REQUEST>
```

SALE CHECK example – RESPONSE – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST3</TXID>
    <HOSTTXID>EP9110002000077746</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>3898</AMOUNT>
</RESPONSE>
```

SALE DIRECT example – REQUEST – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST4</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480003</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>93438734234782423742348723401</PAN>
    </CARD>
    <AMOUNT>3898</AMOUNT>
</REQUEST>
```

SALE DIRECT example – RESPONSE – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST4</TXID>
    <HOSTTXID>EP9110002000077777</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>3898</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.7. Subscription Service

For subscription product or services, an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields
SALE	CHECK	Check if integrator is eligible to create a subscription for a customer through a merchant account.	`PRODUCTID` `CUSTOMER`
SALE	DIRECT	Purchase a subscription for a customer	`PRODUCTID` `CUSTOMER`
CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`

If the subscription is card based such as a SIM registration the account reference should be supplied in the `CARD` object using the `PAN` field.

Examples

SALE DIRECT example - REQUEST

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST</TXID>
    <CASHIER>PK</CASHIER>
    <PRODUCTID>PLSIM12345678</PRODUCTID>
    <CURRENCY>PLN</CURRENCY>
    <SHOPID>1</SHOPID>
    <CARD>
        <PAN>89148000000745809013</PAN>
    </CARD>
    <PHONE>074343934843</PHONE>
    <CUSTOMER>
        <FIRSTNAME>A</FIRSTNAME>
        <LASTNAME>Other</LASTNAME>
        <MIDDLENAME>N</MIDDLENAME>
        <IDENTITY>012687014757</IDENTITY>
        <SOCIALNUMBER>1475712344</SOCIALNUMBER>
    </CUSTOMER>
</REQUEST>
```

SALE DIRECT example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 13:18:01</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110003000150244</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.8. Advertisement Service

For advertisements, an integrator can use the following methods against a merchant account to pull an advert/offer linked to that account.

The `PRODUCTID` must be set to **ADVERT**

Method	Mode	Description	Mandatory Fields
SALE	DIRECT	Pull one or more advertisement from the API to be printed for the customer	`PRODUCTID`

Examples

SALE:DIRECT example – request

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>RetailGroup1</USERNAME>
        <PASSWORD>LuNRdo0j1d6Dx</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>99901335</TERMINALID>
    <LOCALDATETIME>2017-01-23 12:40:23</LOCALDATETIME>
    <TXID>TEST1</TXID>
    <CASHIER>TEST</CASHIER>
    <PRODUCTID>ADVERT</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
</REQUEST>
```

SALE:DIRECT example – response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2017-01-24 11:12:18</SERVERDATETIME>
    <TERMINALID>99901335</TERMINALID>
    <TXID>TEST1</TXID>
    <HOSTTXID>EP9110007000000025</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <ADVERTISEMENT>
        <ADVERT>
            <LINE>************************</LINE>
            <LINE>         OFFER           </LINE>
            <LINE>************************</LINE>
        </ADVERT>
    </ADVERTISEMENT>
</RESPONSE>
```

5.9. Billing Service

For billing product or services the following methods can be used by a merchant account, please note that note all product or services are the same and therefore may require additional fields. Please refer to the product configuration document for more information.

Method	Mode	Description	Mandatory Fields
SALE	CHECK	Used for obtain details of account if valid and supported by service provider.	PRODUCTID CARD
SALE	DIRECT	Used to pay for a billing product or service, the CARD object is used as the object to supply the account reference or invoice reference	PRODUCTID CARD AMOUNT
CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.	TXREF TXREF TYPE=""
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	TXREF TXREF TYPE=""

Examples

SALE CHECK example – REQUEST – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST3</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480003</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>93438734234782423742348723401</PAN>
    </CARD>
</REQUEST>
```

SALE CHECK example – RESPONSE – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:30:19</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST3</TXID>
    <HOSTTXID>EP9110002000077746</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>3898</AMOUNT>
</RESPONSE>
```

SALE DIRECT example – REQUEST – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2016-03-11 14:37:02</LOCALDATETIME>
    <TXID>TEST4</TXID>
    <CASHIER>TESTRIG</CASHIER>
    <PRODUCTID>NZTXN13480003</PRODUCTID>
    <SHOPID>1</SHOPID>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
    </RECEIPT>
    <CARD>
        <PAN>93438734234782423742348723401</PAN>
    </CARD>
    <AMOUNT>3898</AMOUNT>
</REQUEST>
```

SALE DIRECT example – RESPONSE – Invoice Payment

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2017-02-06 12:27:34</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>TEST4</TXID>
    <HOSTTXID>EP9110002000077777</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>3898</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>* Merchant Copy *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>* Customer Copy *</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

5.10. Wallet (Pays) Service

Wallet services relates to the customer using a mobile application or account card as an alternative method of payment for goods or services. A wallet provider in most cases provide a mobile application that a customer will present to the cashier, the cashier will either scan a barcode on the mobile application or print a barcode (acquired from the wallet provider) for the customer to scan with the application.

For wallet services an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields

REDEEM	RESERVE	Reserve payment from a wallet provider	`CARD`
			`AMOUNT`
			
			The `CARD` field is mandatory when a PAYS phone transaction is being initiated, otherwise the `CARD` is field is not mandatory for a QR code transaction.

REDEEM	CAPTURE	Confirm payment with a wallet provider	`TXREF` `TXREF TYPE=””`
REDEEM	REPRINT	Returns the original merchant copy of the receipt supplied by the API for a transaction that was previously requested.	`TXREF` `TXREF TYPE=””`
CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user. A `CANCEL:MANUAL` is considered a refund of the entire transaction and should not be replaced by `REFUND:DIRECT`	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`

REFUND	DIRECT	Partial refund of related transaction at the request of the customer, this request must be instructed by the user.	`TXREF`
			`TXREF TYPE=””`
			`AMOUNT`

In each request the `PRODUCTID` field should be set to EAN relating to how the customer would like to pay (either Scan by Phone or Scan by POS), it is recommended that the integrator supply the basket in each `REDEEM:RESERVE` request.

The `AMOUNT` field is required in both the `REDEEM:RESERVE` and `REDEEM:CAPTURE` requests, and should **not** be altered after the `REDEEM:RESERVE` has been sent.

The `AMOUNT` field is required in `REFUND:DIRECT` for partial payments.

If the integrator chooses to provide the basket this will require the usage of the objects described in sections **3.2** to **3.3.**

Examples

REDEEM RESERVE example – REQUEST

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="REDEEM" MODE="RESERVE">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <ECHO>UhuraTestXMLUP</ECHO>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2018-01-26 10:48:47</LOCALDATETIME>
    <TXID>201801</TXID>
    <CASHIER>epay_tester</CASHIER>
    <PRODUCTID>AUGCD14560001</PRODUCTID>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <SHOPID>EPTST0000001</SHOPID>
    <CARD>
        <PAN>134745539590171622</PAN>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
    </CARD>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
        <TYPE>FULLTEXT</TYPE>
    </RECEIPT>
    <POSENTRY>01</POSENTRY>
</REQUEST>
```

REDEEM RESERVE example – response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="RESERVE" TYPE="REDEEM">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2018-01-26 10:48:47</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>201801</TXID>
    <HOSTTXID>EP9130068000001323</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <RECEIPT>
        <MERCHANT />
        <CUSTOMER />
    </RECEIPT>
</RESPONSE>
```

REDEEM CAPTURE example – REQUEST

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="REDEEM" MODE="CAPTURE">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <ECHO>UhuraTestXMLUP</ECHO>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2018-01-26 10:50:09</LOCALDATETIME>
    <TXID>201801_002</TXID>
    <CASHIER>epay_tester</CASHIER>
    <PRODUCTID>AUGCD14560001</PRODUCTID>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <SHOPID>EPTST0000001</SHOPID>
    <CARD>
        <PAN>134745539590171622</PAN>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
    </CARD>
    <TXREF TYPE="HOST">EP9130068000001323</TXREF>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
        <TYPE>FULLTEXT</TYPE>
    </RECEIPT>
    <POSENTRY>01</POSENTRY>
</REQUEST>
```

REDEEM CAPTURE example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CAPTURE" TYPE="REDEEM">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2018-01-26 10:50:09</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>201801_002</TXID>
    <HOSTTXID>EP9130068000001324</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>&lt;!TC!&gt;* MERCHANT COPY *</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>&lt;!TC!&gt;* CUSTOMER COPY *</LINE>
        </CUSTOMER>
    </RECEIPT>
    <CUSTOMDATA>
        <ITEM>
            <VALUE KEY="out_trade_no">EPWEC0000000982</VALUE>
            <VALUE KEY="transaction_id">42000000572018012613295704 70</VALUE>
        </ITEM>
    </CUSTOMDATA>
</RESPONSE>
```

CANCEL AUTOMATIC example – REQUEST

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="CANCEL" MODE="AUTOMATIC">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <ECHO>UhuraTestXMLUP</ECHO>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2018-01-26 10:52:52</LOCALDATETIME>
    <TXID>201801_004</TXID>
    <CASHIER>epay_tester</CASHIER>
    <PRODUCTID>AUGCD14560001</PRODUCTID>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <SHOPID>EPTST0000001</SHOPID>
    <CARD>
        <PAN>134745539590171622</PAN>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
    </CARD>
    <TXREF TYPE="CLIENT">201801_002</TXREF>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
        <TYPE>FULLTEXT</TYPE>
    </RECEIPT>
    <POSENTRY>01</POSENTRY>
</REQUEST>
```

CANCEL AUTOMATIC example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="AUTOMATIC" TYPE="CANCEL">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2018-01-26 10:52:52</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>201801_004</TXID>
    <HOSTTXID>EP9130068000001327</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>&lt;!TC!&gt;* MERCHANT COPY *</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;WeChat</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;* CANCELLATION SUCCESSFUL *</LINE>
            <LINE></LINE>
            <LINE>epay TXN ID: EP9130068000001327</LINE>
            <LINE>POS Date: 26/01/2018</LINE>
            <LINE>POS Time: 10:52:52</LINE>
            <LINE>Operator: epay_tester</LINE>
            <LINE>Expiry: </LINE>
            <LINE>Supplier ID: EPWEC0000000984</LINE>
            <LINE>Amount: $0.01</LINE>
            <LINE></LINE>
            <LINE>This refund has been</LINE>
            <LINE>successfully processed</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>&lt;!TC!&gt;* CUSTOMER COPY *</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;WeChat</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;* CANCELLATION SUCCESSFUL *</LINE>
            <LINE></LINE>
            <LINE>epay TXN ID: EP9130068000001327</LINE>
            <LINE>POS Date: 26/01/2018</LINE>
            <LINE>POS Time: 10:52:52</LINE>
            <LINE>Operator: epay_tester</LINE>
            <LINE>Expiry: </LINE>
            <LINE>Supplier ID: EPWEC0000000984</LINE>
            <LINE>Amount: $0.01</LINE>
            <LINE></LINE>
            <LINE>This refund has been</LINE>
            <LINE>successfully processed</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

CANCEL MANUAL example – REQUEST

```
<?xml version="1.0" encoding="UTF-8"?>
<REQUEST TYPE="CANCEL" MODE="MANUAL">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <ECHO>UhuraTestXMLUP</ECHO>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2018-01-26 10:52:52</LOCALDATETIME>
    <TXID>201801_005</TXID>
    <CASHIER>epay_tester</CASHIER>
    <PRODUCTID>AUGCD14560001</PRODUCTID>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <SHOPID>EPTST0000001</SHOPID>
    <CARD>
        <PAN>134745539590171622</PAN>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
    </CARD>
    <TXREF TYPE="HOST">EP9130068000001324</TXREF>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
        <TYPE>FULLTEXT</TYPE>
    </RECEIPT>
    <POSENTRY>01</POSENTRY>
</REQUEST>
```

CANCEL MANUAL example – RESPONSE

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="MANUAL" TYPE="CANCEL">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2018-01-26 10:52:52</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>201801_004</TXID>
    <HOSTTXID>EP9130068000001327</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>1</AMOUNT>
    <RECEIPT>
        <MERCHANT>
            <LINE>&lt;!TC!&gt;* MERCHANT COPY *</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;WeChat</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;* CANCELLATION SUCCESSFUL *</LINE>
            <LINE></LINE>
            <LINE>epay TXN ID: EP9130068000001327</LINE>
            <LINE>POS Date: 26/01/2018</LINE>
            <LINE>POS Time: 10:52:52</LINE>
            <LINE>Operator: epay_tester</LINE>
            <LINE>Expiry: </LINE>
            <LINE>Supplier ID: EPWEC0000000984</LINE>
            <LINE>Amount: $0.01</LINE>
            <LINE></LINE>
            <LINE>This refund has been</LINE>
            <LINE>successfully processed</LINE>
        </MERCHANT>
        <CUSTOMER>
            <LINE>&lt;!TC!&gt;* CUSTOMER COPY *</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;WeChat</LINE>
            <LINE></LINE>
            <LINE>&lt;!TC!&gt;* CANCELLATION SUCCESSFUL *</LINE>
            <LINE></LINE>
            <LINE>epay TXN ID: EP9130068000001327</LINE>
            <LINE>POS Date: 26/01/2018</LINE>
            <LINE>POS Time: 10:52:52</LINE>
            <LINE>Operator: epay_tester</LINE>
            <LINE>Expiry: </LINE>
            <LINE>Supplier ID: EPWEC0000000984</LINE>
            <LINE>Amount: $0.01</LINE>
            <LINE></LINE>
            <LINE>This refund has been</LINE>
            <LINE>successfully processed</LINE>
        </CUSTOMER>
    </RECEIPT>
</RESPONSE>
```

REFUND DIRECT example – REQUEST

```
<REQUEST TYPE="REFUND" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>PR</USERNAME>
        <PASSWORD>PR</PASSWORD>
    </AUTHORIZATION>
    <ECHO>UhuraTestXMLUP</ECHO>
    <TERMINALID>11112222</TERMINALID>
    <LOCALDATETIME>2018-12-05 09:16:47</LOCALDATETIME>
    <TXID>REFUND_DIRECT_20181205_01</TXID>
    <CASHIER>epay_test</CASHIER>
    <PRODUCTID>AUGCD14560001</PRODUCTID>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>6</AMOUNT>
    <SHOPID>EPTST0000001</SHOPID>
    <CARD>
        <PAN></PAN>
        <EXPIRY_YEAR></EXPIRY_YEAR>
        <EXPIRY_MONTH></EXPIRY_MONTH>
    </CARD>
    <TXREF TYPE="HOST">EP9130067000015343</TXREF>
    <RECEIPT>
        <LANGUAGE>ENG</LANGUAGE>
        <TYPE>FULLTEXT</TYPE>
    </RECEIPT>
    <POSENTRY>01</POSENTRY>
</REQUEST>
```

REFUND DIRECT example – RESPONSE

```
<RESPONSE MODE="DIRECT" TYPE="REFUND">
    <ECHO>UhuraTestXMLUP</ECHO>
    <SERVERDATETIME>2018-12-05 09:19:39</SERVERDATETIME>
    <TERMINALID>11112222</TERMINALID>
    <TXID>REFUND_DIRECT_20181205_01</TXID>
    <HOSTTXID>EP9130067000015344</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <CURRENCY>AUD</CURRENCY>
    <AMOUNT>6</AMOUNT>
    <CUSTOMDATA>
        <ITEM>
            <VALUE KEY="transaction_id">20181205220014517705005729 51</VALUE>
            <VALUE KEY="out_trade_no">_EPPAY0000011370</VALUE>
        </ITEM>
    </CUSTOMDATA>
</RESPONSE>
```

5.11. Parcel Service

Parcel services relates to the customer using a delivery service for a purchase they have made online where the delivery (containing one or more parcels) is delivered to a store within the epay retail network. A parcel delivery provider in most cases provide an integration to a e-commerce site that a customer can choose to use and select a location within the providers retail network, the customer will receive a collection or delivery code that they will present to the cashier, the cashier will either scan a barcode on the mobile application or key in a code (acquired from the parcel provider) to identify the parcels.

For parcel services an integrator can use the following methods against a merchant account.

Method	Mode	Description	Mandatory Fields
SALE	CHECK	Used for either listing collections or deliveries	`SHOPID` `PRODUCTID`

SALE	DIRECT	Used for confirming collection or delivery of parcel	`PARCELS`
			`PARCEL`
			`CUSTOMER`
			`SHOPID`
			`PRODUCTID`

CANCEL	MANUAL	Refund a transaction at the request of a customer, this request must be instructed by the user.	`TXREF` `TXREF TYPE=””`
CANCEL	AUTOMATIC	Void a transaction based on a system fault or timeout, this request must not be instructed by a user and should happen automatically by the system.	`TXREF` `TXREF TYPE=””`

In each request the `PRODUCTID` field should be set to EAN or PRODUCT ID from epay relating to the request from the customer (either a collection or delivery).

The `AMOUNT` field is not required in any request.

For process flows, please review the guidelines for the parcel service that can be acquired through the account manager.

Examples

SALE CHECK – List Collection – Request

```
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
       <USERNAME>JG</USERNAME>
       <PASSWORD>JG</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>44445555</TERMINALID>
    <PRODUCTID>UKETU12090006</PRODUCTID>
    <LOCALDATETIME>2018-03-15 13:43:00</LOCALDATETIME>
    <AMOUNT>0</AMOUNT>
    <SHOPID>1</SHOPID>
    <TXID>TEST</TXID>
    <RECEIPT />
</REQUEST>
```

SALE CHECK – List Collection – Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2018-03-21 09:32:49</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110071000000101</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>0</AMOUNT>
    <PARCELS>
        <PARCEL>
            <ID>5aa6aac5e138230aa499a129</ID>
            <COLLECTIONCODE>315413</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Alan</FIRSTNAME>
                <LASTNAME>Nguyen</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox95959593</PROVIDERID>
            <STATE>RECEIVED</STATE>
            <DATECREATED>2017-05-21 17:46:59</DATECREATED>
            <DATERECEIVED>2017-05-22 09:10:59</DATERECEIVED>
        </PARCEL>
    </PARCELS>
</RESPONSE>
```

SALE CHECK – List Deliveries – Request

```
<REQUEST TYPE="SALE" MODE="CHECK">
    <AUTHORIZATION>
       <USERNAME>JG</USERNAME>
       <PASSWORD>JG</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>44445555</TERMINALID>
    <PRODUCTID>UKETU12090005</PRODUCTID>
    <LOCALDATETIME>2018-03-15 13:43:00</LOCALDATETIME>
    <AMOUNT>0</AMOUNT>
    <SHOPID>1</SHOPID>
    <TXID>TEST</TXID>
    <RECEIPT />
</REQUEST>
```

SALE CHECK – List Deliveries – Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<RESPONSE MODE="CHECK" TYPE="SALE">
    <SERVERDATETIME>2018-03-21 09:34:36</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110071000000102</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>0</AMOUNT>
    <PARCELS>
        <PARCEL>
            <ID>5aa6aa90e138230aa4999f3f</ID>
            <COLLECTIONCODE>765011</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Brittany</FIRSTNAME>
                <LASTNAME>Long</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox96574002</PROVIDERID>
            <STATE>IN-FLIGHT</STATE>
            <DATECREATED>2017-12-05 16:46:59</DATECREATED>
        </PARCEL>
    </PARCELS>
</RESPONSE>
```

SALE DIRECT – Delivery – Request

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>JG</USERNAME>
        <PASSWORD>JG</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>44445555</TERMINALID>
    <PRODUCTID>UKETU12090001</PRODUCTID>
    <LOCALDATETIME>2018-03-15 13:43:00</LOCALDATETIME>
    <AMOUNT>0</AMOUNT>
    <SHOPID>1</SHOPID>
    <TXID>TEST</TXID>
    <RECEIPT />
    <PARCELS>
        <PARCEL>
            <ID>5aa6aa90e138230aa4999f3f</ID>
            <COLLECTIONCODE>765011</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Brittany</FIRSTNAME>
                <LASTNAME>Long</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox96574002</PROVIDERID>
            <STATE>IN-FLIGHT</STATE>
            <DATECREATED>2017-12-05 16:46:59</DATECREATED>
        </PARCEL>
    </PARCELS>
</REQUEST>
```

SALE DIRECT – Delivery – Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2018-03-21 09:36:10</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110071000000103</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>0</AMOUNT>
    <PARCELS>
        <PARCEL>
            <ID>5ab2278ae4b009f3047741c1</ID>
            <COLLECTIONCODE>819260</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Brittany</FIRSTNAME>
                <LASTNAME>Long</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox96574002</PROVIDERID>
            <STATE>RECEIVED</STATE>
            <DATECREATED>2018-03-21 09:36:10</DATECREATED>
            <DATERECEIVED>2018-03-21 09:36:10</DATERECEIVED>
        </PARCEL>
    </PARCELS>
</RESPONSE>
```

SALE DIRECT – Collection – Request

```
<REQUEST TYPE="SALE" MODE="DIRECT">
    <AUTHORIZATION>
        <USERNAME>JG</USERNAME>
        <PASSWORD>JG</PASSWORD>
    </AUTHORIZATION>
    <TERMINALID>44445555</TERMINALID>
    <PRODUCTID>UKETU12090002</PRODUCTID>
    <LOCALDATETIME>2018-03-15T13:43:00</LOCALDATETIME>
    <AMOUNT>0</AMOUNT>
    <SHOPID>1</SHOPID>
    <TXID>TEST</TXID>
    <RECEIPT />
    <PARCELS>
        <PARCEL>
            <ID>5aa6aac5e138230aa499a129</ID>
            <COLLECTIONCODE>315413</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Alan</FIRSTNAME>
                <LASTNAME>Nguyen</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox95959593</PROVIDERID>
            <STATE>RECEIVED</STATE>
            <DATECREATED>2017-05-21 17:46:59</DATECREATED>
            <DATERECEIVED>2017-05-22 09:10:59</DATERECEIVED>
        </PARCEL>
    </PARCELS>
</REQUEST>
```

SALE DIRECT – Collection – Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<RESPONSE MODE="DIRECT" TYPE="SALE">
    <SERVERDATETIME>2018-03-21 09:37:57</SERVERDATETIME>
    <TERMINALID>44445555</TERMINALID>
    <TXID>TEST</TXID>
    <HOSTTXID>EP9110071000000104</HOSTTXID>
    <TXNERROR>0</TXNERROR>
    <RESULT>0</RESULT>
    <RESULTTEXT>OK</RESULTTEXT>
    <AMOUNT>0</AMOUNT>
    <PARCELS>
        <PARCEL>
            <ID>5aa6aac5e138230aa499a129</ID>
            <COLLECTIONCODE>315413</COLLECTIONCODE>
            <CUSTOMER>
                <FIRSTNAME>Alan</FIRSTNAME>
                <LASTNAME>Nguyen</LASTNAME>
            </CUSTOMER>
            <PROVIDERID>HubBox95959593</PROVIDERID>
            <STATE>COLLECTED</STATE>
            <DATECREATED>2017-05-21 17:46:59</DATECREATED>
            <DATERECEIVED>2017-05-22 09:10:59</DATERECEIVED>
            <DATECOLLECTED>2018-03-21 09:37:57</DATECOLLECTED>
        </PARCEL>
    </PARCELS>
</RESPONSE>
```

6. Basket Data Processing

The following file structures are used to specify how to deliver basket information offline for processing and analysis. Each field must be separated by a TAB (\t) as the delimiter, any quotations will **not** be treated as text identifiers and therefore will become part of the value.

The basket data files are split into two separate files for summary and detail, and a third file for account identifiers such as loyalty cards.

The transfer of basket file should be performed over a secure FTP account which will be supplied to the integrator as part of the integration.

6.1. Summary

The summary file details each basket created at each till, each basket should be unique per store and till.

Name	Mandatory	Data Type	Description
BASKETID	Yes	String	Unique Basket ID assigned, this should be used in the detail file.
TILLID	No	String	Till ID where basket was created
STOREID	Yes	String	Store ID where basket was created
QUANTITY	Yes	Unsigned Integer	Total number of items in the basket
TOTAL	Yes	Signed Integer	Total value of basket in minor currency units
TOTALVAT	No	Signed Integer	Total VAT value of basket in minor currency units
CUSTOMERID	No	String	Unique Customer ID that can be assigned if the customer was identifiable by the integrator
STARTDATE	Yes	Date Time	Date and time at the till when the basket was started.
ENDDATE	Yes	Date Time	Date and time at the till when the basket was completed and paid for.
PAYMENT	Yes	String	The type of payment used for finalizing the basket. CARD CASH LOYALTY COUPON APPLICATION OTHER

6.2. Detail

Each item should be listed as a separate line item, this means that although an item was purchased twice they should be listed separately.

Name	Mandatory	Data Type	Description
BASKETID	Yes	String	Unique Basket ID assigned, this should be used in the summary file.
POSID	Yes	String	Item ID assigned within the basket, this must be unique within the basket only. If there are more than one items and the integrator does not represent them as individual epay suggest using a "–"-ID. **Example** 1-1, 1-2
SKU	Yes	String	SKU of item Also known as a GTIN https://en.wikipedia.org/wiki/Global_Trade_Item_Number
RRP	No	Unsigned Integer	Recommended retail price in minor currency units
UNITCOST	No	Signed Integer	Unit cost at the time of being scanned in minor currency units
ACTUALCOST	Yes	Signed Integer	Actual cost when final basket was totaled after any coupons
PARTOFOFFER	No	Boolean	Was item identified as part of offer
ACTUALVAT	No	Signed Integer	Actual VAT applied to item
SUPPLIERID	No	String	If the item was sent as a part of another request and has a supplier ID reference. Example the item was activated through another supplier.
CATALOGID	No	String	Integrator's own catalog ID for item
MERCHANTCATEGORY	No	String	Category product relates to

6.3. Accounts

If a basket contains account identifiers such as loyalty cards or coupons these should be provided in the accounts field, each reference should be listed as a separate line item.

Name	Mandatory	Data Type	Description
BASKETID	Yes	String	Unique Basket ID assigned, this should be used in the summary file.
POSID	Yes	String	Item ID assigned within the basket, this must be unique within the basket only. If there are more than one items and the integrator does not represent them as individual epay suggest using a "–"-ID. **Example** 1-1, 1-2
TYPE	No	String	Type of account, i.e. coupon or loyalty. This is a flexible field.
REFERENCE	Yes	String	The reference for the account, i.e. the coupon code or loyalty card
DISCOUNT	No	Signed Integer	Value applied as a discount towards the basket. I.e. if the item was a coupon this would be the discount applied as part of the coupon.

7. Glossary

7.1. Terms and Abbreviations

Term/Abbreviation	Definition
Integrator	An integrator can be classified as a EPOS, POS, online retailer, device or switch. This is the system that will integrate with the API
Merchant Account	A merchant account represents an account on the transaction platform that all requests are billed again. The merchant account is identified using the `TERMINALID` and the `AUTHORIZATION` details.
POX	POX stands for Plain Old XML
Json	JavaScript object notation
SOAP	Simple Object Access Protocol
API	Application Programming Interface
User	Represents a user of the integrator system, also known as a cashier, customer service representative.
Coupon	In this document where the term 'coupon' is used it refers to the term Digital Coupon as defined below
Paper Coupon	A paper coupon is a physical ticket or document that can be exchanged for a financial discount or for loyalty points when making a purchase
Digital Coupon	A Digital Coupon is an electronic presentation, that is distributed and presented without manifesting as a "paper" or in other hard-copy form, and that can be exchanged for a financial discount or for loyalty points when making a purchase
Mobile Coupon	Mobile coupons are a type of digital coupon received and/or presented via consumer mobile devices
Digital Coupon Instance	A digital coupon instance is an individual digital coupon issued to offer user
Merchant account	A merchant account a transaction is billed against
Terminal ID	A unique identifier relating to a transaction terminal on the merchant account

8. Appendix

8.1. Result Codes

RESULT	RESULTTEXT	CANCELLATION REQUIRED
1	FORMAT	No
	DUPLICATE LINE	
	DUPLICATE NUMBER	
	EXCEEDED LINES	
	INACTIVE LOYALTY ACCOUNT	
	INACTIVE OLD LOYALTY ACCOUNT	
	INVALID BATCH	
	INVALID CLASS	
	INVALID COMMAND	
	INVALID CURRENCY	
	INVALID DATETIME	
	INVALID DISTRIBUTOR ID	
	INVALID END DATE	
	INVALID EXPIRY DATE	
	INVALID LINES	
	INVALID LOYALTY ACCOUNT	
	INVALID LOYALTY PAN	
	INVALID LOYALTY PAN LUHN	
	INVALID LOYALTY REFERENCE	
	INVALID LOYALTY SCHEME	
	INVALID LPN	
	INVALID MERCHANT ID	
	INVALID MULTIBANK	
	INVALID NUMBER SELECTION	
	INVALID OLD LOYALTY ACCOUNT	
	INVALID OLD LOYALTY PAN	
	INVALID OLD LOYALTY PAN LUHN	
	INVALID OLD LOYALTY REFERENCE	
	INVALID OLD LOYALTY SCHEME	
	INVALID PAN	
	INVALID PAN LENGTH	
	INVALID PAN LUHN	

	INVALID POSTCODE	
	INVALID PRODUCT	
	INVALID PURPOSE OF USE	
	INVALID QML PRODUCT	
	INVALID REFERENCE	
	INVALID REFUND TRANSACTION ID	
	INVALID START DATE	
	INVALID STATE	
	INVALID STORE ID	
	INVALID SUPPLIER ID	
	INVALID TFL PRODUCT	
	INVALID TRANSACTION ID	
	INVALID TRIP	
	INVALID VRM	
	INVALID WEEKS	
	LOYALTY ACCOUNT ALREADY REGISTERED	
	LOYALTY REGISTRATION MISMATCH	
	PIN REQUIRED	
	ORDER ID/REFERENCE NOT FOUND	
	INVALID ORDER ID/REFERENCE	
	PRODUCT PICKED NOT FOR ORDER	
2	**CARD NOT FOUND**	**No**
	CARD INVALID	
	CARD NOT RECOGNISED	
	INVALID MOBILE NUMBER	
	TICKET NOT FOUND	
3	**ACTIVATION ERROR**	**No**
	CARD ALREADY ACTIVATED	
	PRODUCT ALREADY ACTIVATED	
4	**EXPIRED**	**No**
	ADD VALUE EXPIRED	
	CARD EXPIRED	
	PRODUCT EXPIRED	
5	**CARD ALREADY DEACTIVATED**	**No**
	GIFT CARD VOID	
	REFUND ALREADY PROCESSED	
	TRANSACTION ALREADY REFUNDED	

	TRANSACTION ALREADY VOID	
	VOID ALREADY PROCESSED	
6	**CARD NOT ACTIVE**	**No**
	BATCH DISABLED	
	CARD DISABLED	
	CARD INACTIVE	
	CARD NOT ACTIVATED	
	CARD NOT REGISTERED	
	MOBILE NOT ACTIVE	
	PRODUCT DISABLED	
	PRODUCT NOT PERMITTED	
7	**TXREF_MISSING**	**No**
8	**TXNOTFOUND**	**No**
	TRANSACTION NOT FOUND	
9	**INVALID AMOUNT**	**No**
	AMOUNT MISMATCH	
	INVALID AMOUNT	
	INVALID VALUE	
10	**NO ACCOUNT FOUND**	**No**
	NO ACCOUNT FOUND	
11	**ILLEGAL VALUES**	**No**
	ILLEGAL NEW ACCOUNT VALUES	
12	**CANCEL MSG TYPE NOT ALLOWED**	**No**
	ILLEGAL CANCEL MSG TYPE	
13	**TYPE NOT ALLOWED**	**No**
	GAME CLOSED	
	LOTTERY SOLD OUT	
14	**CARD USED**	**No**
	PRODUCT ALREADY USED	
	WINNINGS ALREADY CLAIMED	
15	**DUPLICATE ERROR**	**No**
	DUPLICATE TRANSACTION ID	
	SEQUENCE ERROR	
16	**LIMIT EXCEEDED**	**No**
	ACCOUNT LIMIT REACHED	
	CREDIT LIMIT EXCEEDED	
	DAILY LIMIT EXCEEDED	

17	**CUSTOMER LOCKED**	**No**
	CUSTOMER ACCOUNT LOCKED	
18	**BALANCE INVALID**	**No**
	BALANCE NOT ZERO	
19	**REGION ERROR**	**No**
	WRONG AREA	
20	**NO VALID CARD PROFILE**	**No**
	CARD AND MOBILE MISMATCH	
	MOBILE NUMBER NOT PREPAY	
21	**CARD PROFILE ISSUE**	**No**
	CARD CREATE ERROR	
22	**PAYMENT INVALID**	**No**
	PAYMENT ALREADY MADE	
23	**PIN ERROR**	**No**
	INCORRECT PIN	
29	**FUNDS ERROR**	**No**
	INSUFFICIENT FUNDS	
	WON MORE THAN FIFTY	
30	**PENDING TRANSACTION**	**No**
	REFUND PENDING	
	TRANSACTION IN PROCESS	
	VOID PENDING	
31	**STATE INVALID**	**No**
	INVALID NEW STATE	
32	**STOCK ERROR**	**No**
	OUT OF STOCK	
33	**CARD BLOCKED**	**No**
	ACCOUNT BLOCKED	
	CARD IS BARRED	
	CARD IS STOLEN	
	MOBILE IS BARRED	
	MOBILE IS STOLEN	
	PIN BLOCKED	
46	**5 MINUTE LIMIT ERROR**	**No**
47	**24 HOUR LIMIT ERROR**	**No**
48	**ALREADY DEACTIVATED**	**No**
50	**PRODUCT NOT ALLOWED**	**No**

51	**BATCH ERROR**	**No**
86	**LOGIN BLOCKED**	**No**
87	**LOGIN NOT ACTIVATED**	**No**
4000	**MERCHANT UNKNOWN**	**No**
	RETAILER UNKNOWN	
	TERMINAL UNKNOWN	
4001	**MERCHANT BLOCKED**	**No**
	RETAILER BLOCKED	
	TERMINAL BLOCKED	
5000	**COUPON ERROR**	**No**
	COUPON EXPIRED	
	COUPON STOLEN	
	COUPON TIMING INVALID	
	PROMOTION ENDED	
	COUPON ALREADY REDEEMED	
	NOT VALID FOR STORE	
	EMPTY BASKET ITEM	
	COUPON CANNOT BE UNREDEEMED	
	OFFER RULES VIOLATED	
	COUPON NOT ACTIVE	
	INVALID GS1 GCN	
	CUSTOMER RULES VIOLATED	
	INCORRECT REDEMPTION PIN	
	BASKET ID NOT PROVIDED	
	BASKET QUANTITY NOT PROVIDED	
	BASKET TOTAL NOT PROVIDED	
6000	**PROMPT**	**No**
6001	**PARTIAL REJECTION, AMOUNT REQUIRED**	**No**
9988	**NOT SUPPORTED**	**No**
	ACCOUNT CORRUPT	
	CAPTURE ALREADY PROCESSED	
	CAPTURE PENDING	
	CAPTURE PERIOD EXCEEDED	
	CARD MUST BE SWIPED	
	CONTACT CUSTOMER SERVICE	
	DISTRIBUTORACC NOT TRADING	
	DRAW PENDING	

	INACTIVE PRODUCT	
	INACTIVE PRODUCT GROUP	
	INACTIVE QML PRODUCT	
	INACTIVE TFL PRODUCT	
	LOT NOT CONFIRMED	
	NO VALID ITEMS	
	NO WINNING NUMBERS	
	NOT A WINNER	
	POSTCODE FULL	
	PRIZES LOCKED	
	REFUND DISABLED	
	REFUND PERIOD EXCEEDED	
	REWARD FAILED	
	SOME INVALID ITEMS	
	SUPPLIERACC NOT TRADING	
	TOO MANY ITEMS	
	TRANSACTION DECLINED	
	TRANSACTION REJECTED	
	UPGRADE ALREADY TAKEN	
	UPGRADE EXPIRED	
	UPGRADE NOT AVAILABLE	
	UPGRADE NOT YET VALID	
	VOID DISABLED	
	VOID PERIOD EXCEEDED	
9994	**MAC ISSUE**	**No**
9995	**SERVICE UNAVAILABLE**	**Yes**
	NO REMOTE CONNECTION	
	REMOTE AUTH FAILED	
	REMOTE AUTH TIMEOUT	
	REMOTE COMMAND FAILED	
	REMOTE COMMAND TIMEOUT	
	REMOTE CONNECT FAILED	
	REMOTE CONNECT TIMEOUT	
	REMOTE CONNECTIONS AT MAXIMUM	
	REMOTE RESPONSE INVALID	
	REMOTE RESPONSE TIMEOUT	
	REMOTE SETUP FAILED	

	REMOTE SETUP TIMEOUT	
	SERVICE DOWN	
	SERVICE UNAVAILABLE	
9996	**STRUCTURE ERROR**	**Yes**
9997	**AUTHENTICATION ERROR**	**No**
	AUTHENTICATION FAILED	
	INVALID SOURCE ADDRESS	
	INVALID TERMINAL	
9998	**INTERNAL ERROR**	**Yes**
	INTERNAL ERROR	
	INVALID PRODUCT CONFIG	
9999	**UNDEFINED**	**Yes**