Intarget HTTP/XML API

2015

Table of Contents

Version History

Version	Author	Contact	Date	Notes
Draft	Andre Visser	-	11/02/2015	Initial draft version for internal review
V1.0	Andre Visser	-	15/03/2015	Second internal review
V2.0	Andre Visser	-	13/05/2015	Final release version
V2.1	Martin de Jager	martin@intarget.mobi	17/11/2015	Updates CI and review

1. Introduction

The XML/HTTP to Higate Gateway enables an authorized Higate HTTP user to send content - such as SMS messages - to Higate, and to receive content - such as Delivery Notifications and Mobile-Originated messages - from Higate, using XML over HTTP. Higate delivers content to the user asynchronously, by POSTing XML data to a customer-designated URL.

2. Environment

2.1 Setup

To use the HTTP API, the following is needed:

- A correctly configured Higate account, which includes:
- An authorized user name and password;
- A service code
- A Higate URL to which to post your XML requests.

Your Higate account manager will give you the above information (Refer to section 5 for more information). You will also provide your account manager with a URL to which Higate should send content. This content includes mobile-originated (MO) messages, status notifications, and delivery notifications. Your URL must accept the XML content from Higate and return an XML status response to Higate.

2.2 Protocol

The Higate system supports both HTTP 1.0 and HTTP 1.1 with some exceptions. Only the minimum required headers as specified by the RFC is required.

Note that the "Keep Alive" header option is ignored and the default HTTP 1.1 behaviour of keeping the connection open is not supported. The system will always close connection after the response has been send.

HTTPS connections are not supported due to the large overhead of SSL.

3. General Message Formats

3.1 XML Substitutions

When a transaction is submitted in plain text format it is important to substitute the XML reserved characters with XML safe escape codes. The substitutions are listed in Table 1 below:

Table 1 – XML Character Substitutions

Character	Substitute
&	&amp;
'	&apos;
"	&quot;
<	&lt;
>	&gt;

3.2 Text Encoding

The Higate system makes provision for submitting message content in plain text, hexadecimal encoding and base-64 encoding. For example:

The following text:

“INTEGRAT”

can be HEX encoded as:

“494e544547524154”

Or Base-64 Encoded as

“SU5URUdSQVQ=”

Alternate encoding schemes would typically be used when submitting binary content or where the content is not XML safe and the character substitutions are not used for whatever reason. The encode message are decoded by the Higate system before submission to the network operator.

4. Flags

The Higate system makes use of flags to define particular properties of transaction that cannot be achieved through the standard parameters and attributes.

Note: Number prefixed with “*0x*” denotes hexadecimal numbers (base 16).

Flag Name	Value	Description
HIGATE_FLAG_SMS_POPUP	0x00000001	Send SMS as a popup message if supported by operator
HIGATE_FLAG_SMS_FLASH	0x00000002	Send SMS as a flashing popup message if supported by operator
HIGATE_FLAG_SMS_8BIT	0x00000004	SMS contains 8 bit (binary) data note1
HIGATE_FLAG_SMS_UDH	0x00000008	SMS data includes a User Data Header (UDH)
HIGATE_FLAG_SMS_SEGMENT	0x00000010	SMS is a segment of a larger message
HIGATE_FLAG_SMS_DN_MASK	0x001F0000	SMS Delivery Receipt Mask
HIGATE_FLAG_SMS_DN_FINAL	0x00010000	SMS Delivery Receipt requested where final delivery outcome is success or failure
HIGATE_FLAG_SMS_DN_FAILED	0x00020000	SMS Delivery Receipt requested where final delivery outcome is failure
HIGATE_FLAG_SMS_DN_INTERM	0x00100000	SMS Delivery Receipt requested for Intermediate notifications
HIGATE_FLAG_OBS_LINKED	0x00000020	SMS pending OBS authentication
HIGATE_FLAG_OBS_TKT_ADDR	0x00000040	Charge the SMS to the ticket charge address (OBS-linked only)
HIGATE_FLAG_OBS_AUTO_CONFIRM	0x00000200	Auto-confirm once authorized
HIGATE_FLAG_OBS_SUBSCRIBE_ONLY	0x00000400	OBS transaction is a subscription only transaction
HIGATE_FLAG_HEX_ENCODED	0x00000080	The source string is hex encoded text (eg "060BFC')
HIGATE_FLAG_MT_BILLED	0x00000100	MT Billed content
HIGATE_FLAG_USS_EXIT	0x00000001	Terminate this USSD Session

Note1: Although all networks support binary SMS some networks forbid the use of WAP Push messages.

Multiple flags can be combined by bitwise OR (adding) the respective flags.

5. Routing

Example: Routing Configuration Sheet

```
Login Details
-------------
ID                : 1234
Name              : USERNAME
Password          : PASSWORD
Created           : 05/03/17 12:49:21
Status            : Enabled
API Type          : HTTP
Default Service   : SERVICECODE
Default URL       : http://www.yourdomain.co.za/higate.php
Auto-drawdown     : Yes
Drawdown Amount   : 26000 Credits
Final Status Only : False

Service Codes  Vod Service Code  URL
-------------  ----------------  ---
SERVICECODE          INT00123        http://www.yourdomain.co.za/higate.php


Transmit (MT) routings by service code
--------------------------------------
C3T01
  SMS
    Rule: By Network
      MTN
        GateID: 272
        Source Address: 27839300365     PUBLIC   Tags ENABLED  1001
    Rule: By Network
      CellC
        GateID: 92
        Source Address: 27840004683     PUBLIC   Tags ENABLED  1001
Rule: By Network
      Vodacom
        GateID: 209
        Source Address: 27820048062     PUBLIC   Tags ENABLED  1001
  USS
    Rule: Default
        GateID: 299
  OBS
    Rule: By Network
      CellC
        GateID: 203
    Rule: By Network
      Vodacom
        GateID: 211
Rule: By Network
      MTN
        GateID: 254
  VSR
    Rule: Default
        GateID: 316

Receiver (MO) SMS routings by service code
------------------------------------------
SERVICECODE
54321     by Default Number
```

Each account login has a routing sheet (see Listing 1) associated with it, which contains all the necessary information to transact with the Higate system. The configuration in the example listing is used for all examples in this document.

6. Quick Start

Let's take a simple scenario: you want to send a single SMS message and receive status notifications. For the purposes of this discussion, assume that the Higate URL to which you must post the SMS is *http://example.com/sms*, and the URL of your web server that will receive status notifications is *http://mycompany.com/notify*. Let's also say that your user name is USERNAME, your password is PASSWORD, and your service code is SERVICECODE.

To send an SMS, you do the following:

1. **Create an XML SMS request**, that is, a valid XML document containing the request data and SMS message.

2. **Send the XML SMS request** to the system by doing an HTTP POST to the API URL, with content-type set to "text/xml; charset=iso-8859-1".

3. **Receive and check the HTTP response**. On receiving a successful HTTP response (such as HTTP 200), check the status code in the XML returned by system. Assuming a successful status code, extract the fields of interest from the response.

4. **Receive status notifications** sent to your web site (example URL: *http://mycompany.com/notify*) and send acknowledgement XML response to Higate. When a delivery notification arrives at your URL, you will know the corresponding SMS - identified by the sequence number attribute '*seq_no*' - was delivered successfully.

Here are the above steps in more detail:

6.1 Create an XML SMS Request

Create an XML document as shown below. This is the simplest possible XML needed to send an SMS. However, depending on how your account is set up, you may need to provide more fields than this. **Note that keywords (such as SendSMS), and XML element and attribute names (such as ToAddr), are case-sensitive.**

Simple SMS XML Request

```
<?xml version='1.0'?>
<Message>
    <Request Type="SendSMS" RefNo="1">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <SendSMS ToAddr="0829993619" Validity="00020000"
       TimeMask="12583039" Flags="0" DataCoding="0">
          <Reply Tag="1001" />
          <AdultRating>0</AdultRating>
          <Content Type="TEXT">Test message from Higate</Content>
       </SendSMS>
     </Request>
</Message>
```

Please note that a new '*TimeMask*' attribute has been added to the '*OBSRequest*' parameters when submitting these types of messages to the system.
For Example:

```
<SendSMS ToAddr="0829993619"
         Validity="00020000"
         TimeMask="12583039"
```

The TimeMask parameter is a 32-bit value where each of the lower 24 bits represents an hour of the day. If a bit is set then it means that the message will not be submitted to the network during that hour. For example: a TimeMask value of 12583039 (decimal) or 00000000 11000000 00000000 01111111 (binary) means to not send the message in between 22H00 and 23H59:59, and not between 00H00 and 06H59:59. When this attribute is omitted it will have a default value of 0, which indicates that the message can be sent at any time.

6.2 Send the XML SMS Request

Post the XML to the URL:

xhg-lb1.higate.co.za:8888/hg_request

As content-type "text/xml; charset=iso-8859-1".

6.3 Check the HTTP Response

If all goes well, you will receive a successful HTTP response of content-type text/xml. It will contain XML that looks similar to this:

HTTP Response

```
<Response status_code='0' token='#%#TOK_I1JlZjwwLjAuMC4zMTkwPg==='>
    <Data name='msg_generic_rsp'>
        <field name='msg_no' value='4' />
        <field name='seq_no' value='1504089' />
    </Data>
</Response>
```

It is important to note that all XML responses contain a status code in the Response element (*status_code*). A zero (0) status code means success. Any other value means failure.

The fields of interest always appear inside the Data element. In the response to a SendSMS, there's really only one important field (besides status_code, of course): the seq_no field. You can use this information to correlate the corresponding status and delivery notifications that the system sends to your URL. In other words, when you receive a status notification on your Web site, it will contain a seq_no field whose value matches the seq_no value in one of the requests you sent earlier.

Please be aware that a successful XML response (namely, *status_code='0'***) only means that your request was accepted for processing, not that it completed successfully.**

6.4 Receive Status Notifications

You will only know whether or not it completed successfully when you get a status notification on your notification web site. Notifications may be sent at each stage of processing your request, for example, when it is queued, acknowledged, and delivered. On the other hand, you may only receive a delivery notification, but you will get at least one notification of the final status of the SMS. It is only when you receive a delivery notification that you can be sure that the SMS was sent to the recipient. Of course, nobody can guarantee that the recipient actually read the SMS!

6.5 Check Status Notifications

As mentioned earlier, you have a web site to receive status notifications. Higate POSTs these status notifications to that URL when your message changes status, such as when it gets a delivery notification from a network. It POSTs these notifications in XML format to your URL with a content-type of "text/xml; charset=iso-8859-1".

IMPORTANT NOTE

Your web site must respond to each POSTed status notification with an XML acknowledgement (a response to Higate's POST with a content-type of "**text/xml; charset=iso-8859-1**"). Failure to do so will cause Higate to assume that you did not receive the message

and to reattempt delivery and take other steps to address what it thinks is an error. There are two possible types of response, shown below.

Success Response to Higate

```
<Response status='0'/>
```

Error Response to Higate

```
<Response status='-1'>
   <Data name="error">
      <field name="reason"
      value="Reason for refusing the response" />
   </Data>
</Response>
```

Here's an example of one of the earliest status notifications (known as '*OnResult*' events) that you would receive if you sent the sample SMS shown earlier. It has a status of "*Queued*". Note the matching '*SeqNo*' (1504089). The '*RefNo*' field, which is user-defined, can also be used to correlate responses, but then it's your responsibility to ensure that '*RefNo*' numbers are unique to each request, even across multiple systems you may have running. If this is not the case, you may not be able to correlate these response messages correctly. In this example, no reference number was provided, so it defaulted to zero, as you can see in the response below:

```
<Message>
    <Version Version='1.0' />
    <Response Type='OnResult' TOC='SMS' RefNo='0' SeqNo='1504089'>
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <OnResult Code='1' SubCode='0' Text='Queued' />
    </Response>
</Message>
```

6.6 Response Messages

It's critical to understand that simply receiving a successful HTTP response, like an HTTP 200, is not enough to guarantee that your post succeeded. You must also check the '*status_code*' attribute in the Response XML element. If it's anything other than zero, there was an error posting your request. Details of the error will be provided.

There are four types of response messages all with the same format but indicating different conditions:

6.6.1 Response – XML Error

An XML Error Response will be in the format below:

XML Error Template

```
<Response status_code="-1">
   <Data name="xml_error">
      <field name="error_code" value="<ErrCode>"/>
      <field name="reason" value="{<Param>,&amp;quot;<Reason>&amp;quot;}}"/>
   </Data>
</Response>
```

When submitting a message to the Higate system first checks the format of the XML message and that certain key parameters are included:

Description	<ErrCode>	<Param>	<Reason>
UserID parameter not found	1	xml_missing_tag	UserID
Request parameter not found	1	xml_missing_tag	Request
Message parameter not found	1	xml_missing_tag	Message
Content not specified for SMS	1	xml_missing_tag	Content
SendSMS parameter not specified for SMS	1	xml_missing_tag	SendSMS
USSText parameter not specified for USSD	1	xml_missing_tag	USSText
USSReply parameter not specified for USSD	1	xml_bad_sessionid	USSReply
Ticket parameter not specified for OBS	1	xml_missing_tag	Ticket
OBSRequest parameter not specified for OBS	1	xml_missing_tag	OBSRequest
OBSConfirm parameter not specified for OBS Confirm	1	xml_missing_tag	OBSConfirm

**This list is in not complete and is included for description purposes.*

Note: A missing parameter can also be caused by malformed XML.

6.6.2 Response – Request Error

A Request Error response will be in the format below:

XML Request Error Template

```
<Response status_code="-1">
   <Data name="req_error">
      <field name="error_code" value="<ErrCode>"/>
      <field name="reason" value="{<Param>,&amp;quot;<Reason>&amp;quot;}}"/>
    </Data>
</Response>
```

- Request errors occur when the basic format of the XML is correct but the system is unable to process the message.
- An Error Code of “2” indicates a temporary system level error and the transaction should be retried at a later stage.
- An ErrorCode of “1” indicates that the parameters specified are not supported by the system. This can be due to services no longer available or an incorrect combination of services.

Description	<ErrCode>	<Param>	<Reason>
Higate System Error - Unable to process transaction	2	bind_error	try again
Higate System Error - Transaction processing failed	2	failed_call	try again
The type of request is not supported by the system	1	unknown_request_type	Value of “Type”
The TOC specified does not apply to the Type specified	1	unsuported_request_type	Value of “Type”

**This list is in not complete and is included for description purposes.*

6.6.3 Response – NAK

An NAK Error Response will be in the format below:

XML NAK Error Template

```
<Response token=""status_code="-1">
   <Data name="msg_nak">
      <field name="status" value="<Status>"/>
      <field name="error_text" value="<ErrText>"/>
   </Data>
</Response>
```

After successfully parsing and validating a submitted message the transaction is processed by the Higate system. Although the format is correct there can be a variety of reasons for a transaction to be failed at this point.

The <*Status*> field contains the internal system error code and the <*Err Text*> contains the detailed description of that error.

For example the following reply is received when trying to use an incorrect service code:

XML Error Example

```
<Response token="" status_code="-1">
   <Data name="msg_nak">
      <field name="status" value="258"/>
      <field name="error_text" value="No routing defined for service(XYZ)."/>
   </Data>
</Response>
```

There are a number of reasons for these types of failures, the most common of which are:

- Incorrect routing, i.e. bearer not routed on the service code
- Duplicate reference numbers for OBS and VSR transactions

6.6.4 Response – ACK

An ACK (success) Response will be in the format below:

XML Response Template

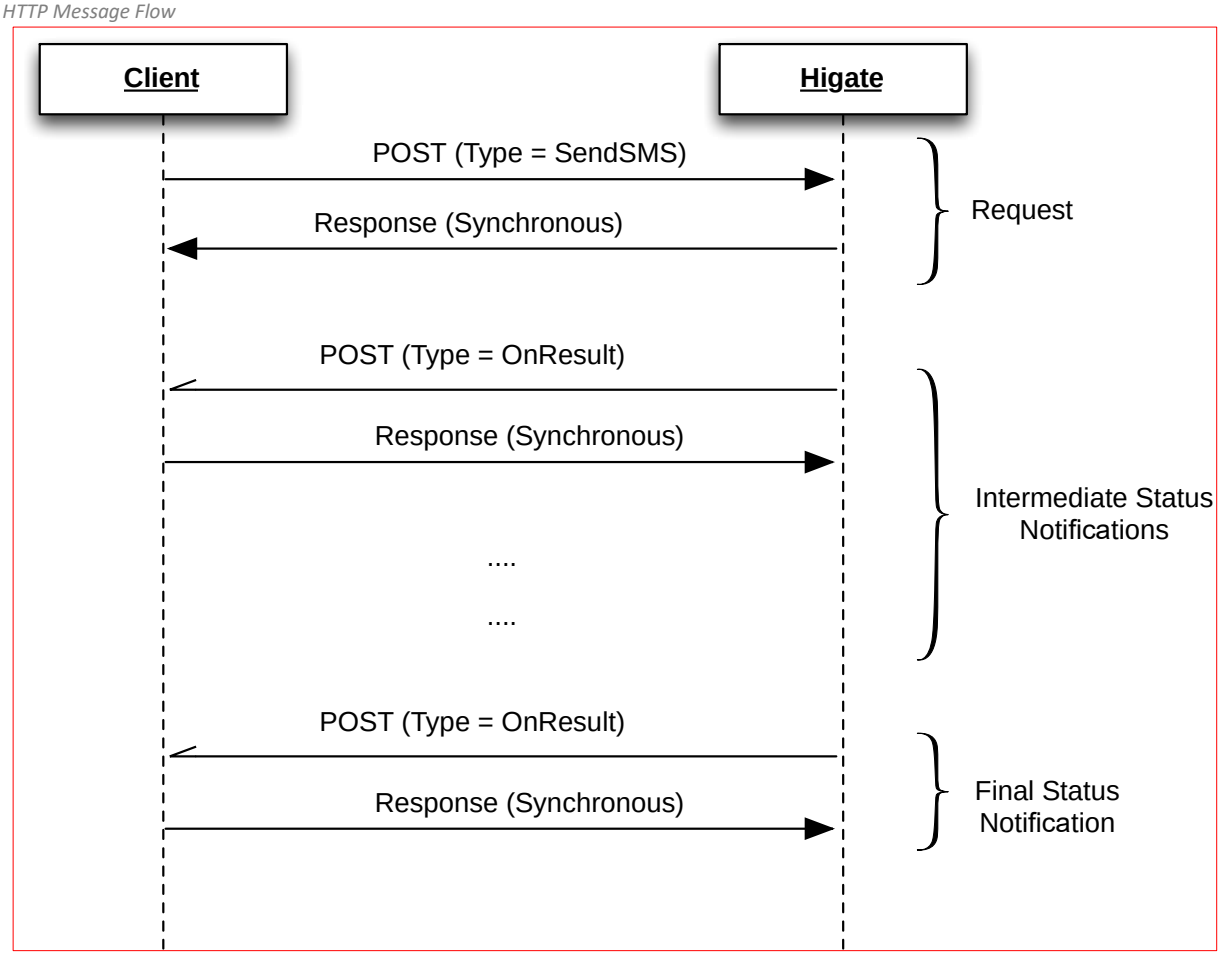
```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="<MsgNo>"/>
      <field name="seq_no" value="<SeqNo>"/>
   </Data>
</Response>
```

An **ACK** response indicates that the message was successfully submitted to the Higate system. The '*SeqNo*' parameter is a unique number that is created for every transaction and included in all call back messages. The '*MsgNo*' is an internal number and can be ignored.

7. SMS MT (TOC_SMS)

7.1 Overview

HTTP Message Flow



7.2 Character Encoding

The content of SMS messages should usually be submitted as plain ASCII (ANSI X3.4) but there are exceptions to this rule when the content contains characters that don't overlap between ASCII and GSM7 (see SMS3.38).

SMPP 3.4 "suggests" that the ESME submits the short message in 8-bit ASCII (including extended characters) to the SMSC and that the SMSC performs the character conversion based on the data-coding scheme specifies.

This is however just a suggestion and not a rule, and mobile operators are inconsistent in applying these rules. In GSM7 for example submitting "**}**" would lead to an escape sequence being sent to the phone as this character is in the extended part of the alphabet table. MTN does this correctly for example but Vodacom does not, it just drops the top bit off the 8-bit ASCII submitted. Some other characters affected are " **{|~ [\]^_** " (all in the extended part).

Another example is *latin-1* encoding which should map 1:1 to ASCII, but here CellC sends this to the phone as GSM7. There can also be differences between different SMSCs at the same operator, some using GSM7 by default and others using CCITT T.50.

When a message contains characters that don't overlap it is important to test the resulting message on a physical device for each network to ensure the message is displayed correctly.

7.3 HTTP Examples

7.3.1 Sending an MT SMS

7.3.1.1 SendSMS Transaction

HTTP POST

```
<?xml version="1.0"encoding="UTF-8"?>
<Message>
    <Request Type="SendSMS" RefNo="1234">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <SendSMS ToAddr="27820000000">
          <Content Type="TEXT">Test message from Higate</Content>
       </SendSMS>
    </Request>
</Message>
```

HTTP Reply (Synchronous)

```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="1"/>
      <field name="seq_no" value="12345678"/>
   </Data>
</Response>
```

HTTP Call back (Asynchronous)
Intermediate Status Notification

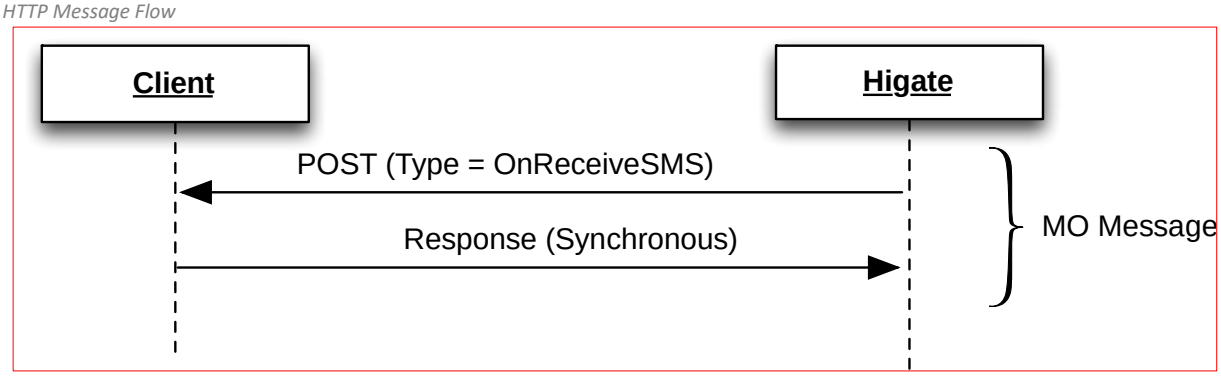
```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="SMS" RefNo="1234" SeqNo="12345678">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001"/>
       <ErrCode>0</ErrCode>
       <ErrText>[E2A0]Receipted</ErrText>
       <OnResult Flags="0" Code="3" SubCode="0" Text="[E2A0]Receipted"/>
    </Response>
</Message>
```

HTTP Call back (Asynchronous)
Final Status Notification

```
<?xml version="1.0"?>
<Message>
    <Version Version="1.0"/>
    <Response Type="OnResult" TOC="SMS" RefNo="1234" SeqNo="12345678">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001"/>
        <ErrCode>0</ErrCode>
        <ErrText>[E2A0]Receipted</ErrText>
        <OnResult Flags="0" Code="4" SubCode="0" Text="[E2A0]Receipted"/>
    </Response>
</Message>
```

8. SMS MO

HTTP Message Flow



8.1 HTTP Examples

8.1.1 Standard MO SMS

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnReceiveSMS">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <Network ID="1" MCC="655" MNC="001"/>
        <OnReceiveSMS SeqNo="95683776" Sent="20150211134923"
                FromAddr="27820000000" ToAddr="27820048062216" ToTag="216"
                Value="0" NetworkID="1" AdultRating="0" DataCoding="1"
                EsmClass="0">
            <Content Type="TEXT">Example reply</Content>
        </OnReceiveSMS>
    </Response>
</Message>
```

8.1.2 Binary MO SMS

The following example shows the format of a SMS with binary content. Note that the Datacoding attribute is set to 8 indicating the content is represented by the UCS2 character set. The Higate system encodes the UCS2 (Big Endian) using Hexadecimal Encoding as indicated by the Content Type attribute (*<Content Type="HEX">*)

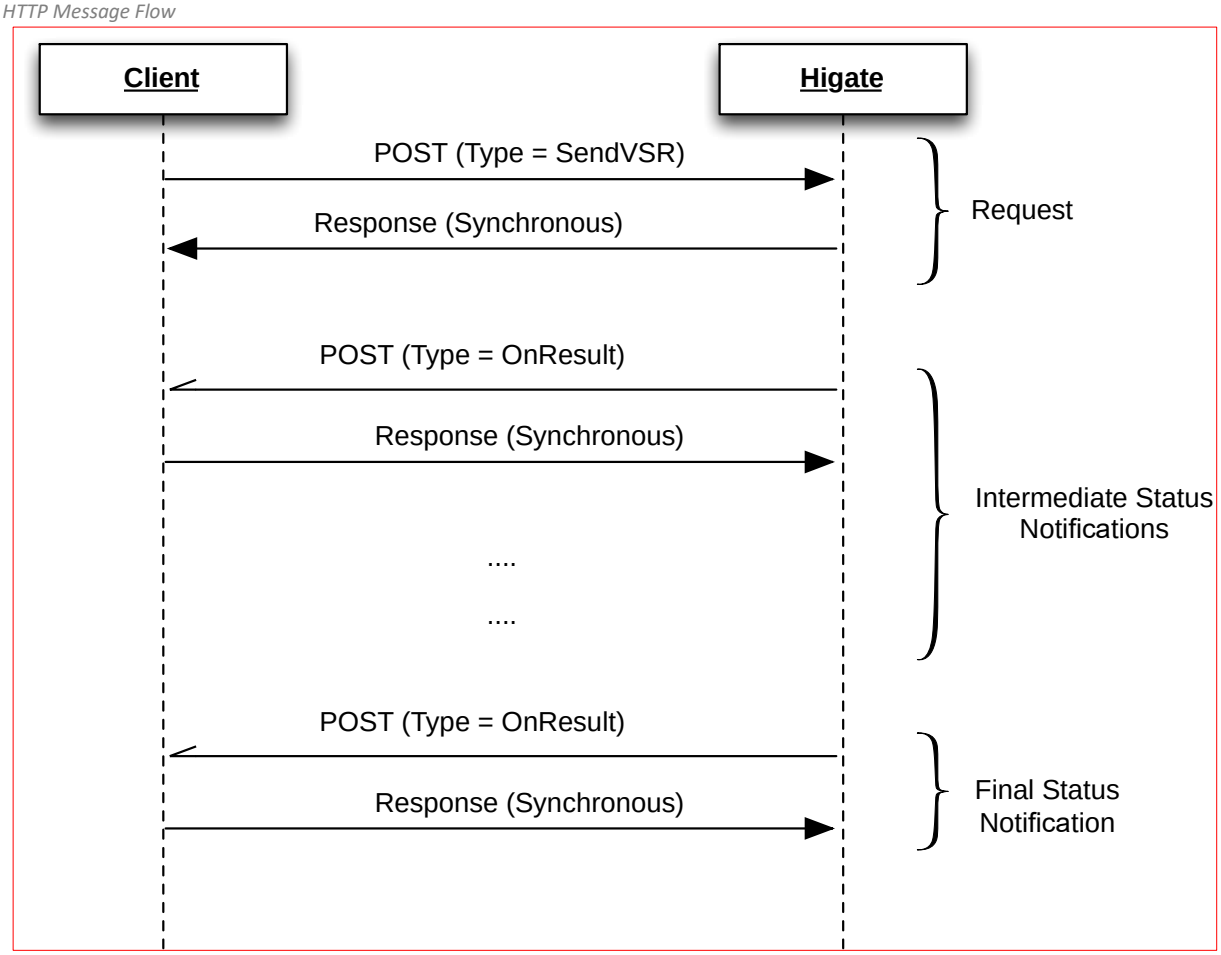
The XML listing below contains the message: **ëxämplë bïnárÿ**

Binary MO SMS

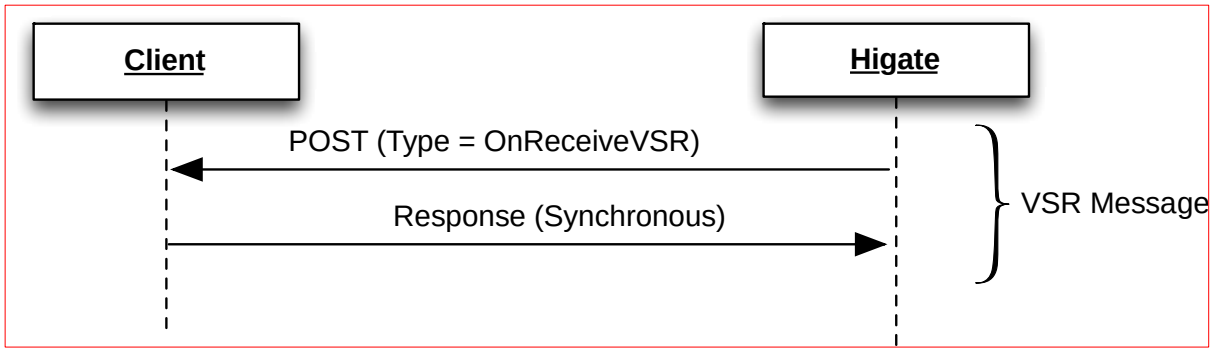
```
<?xml version="1.0"?>
<Message>
    <Version Version="1.0" />
    <Response Type="OnReceiveSMS">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <Network ID="1" MCC="655" MNC="001" />
       <OnReceiveSMS SeqNo="95691869" Sent="20150212063515"
        FromAddr="27767931435" ToAddr="27820048062216" ToTag="216"
        Value="0" NetworkID="1" AdultRating="0" DataCoding="8"
        EsmClass="0">
          <Content Type="HEX">00E800E4006D0070006C00EB0020006200EF006E00E1007200FF</Content>
       </OnReceiveSMS>
     </Response>
</Message>
```

9. VSR (TOC_VSR)

HTTP Message Flow



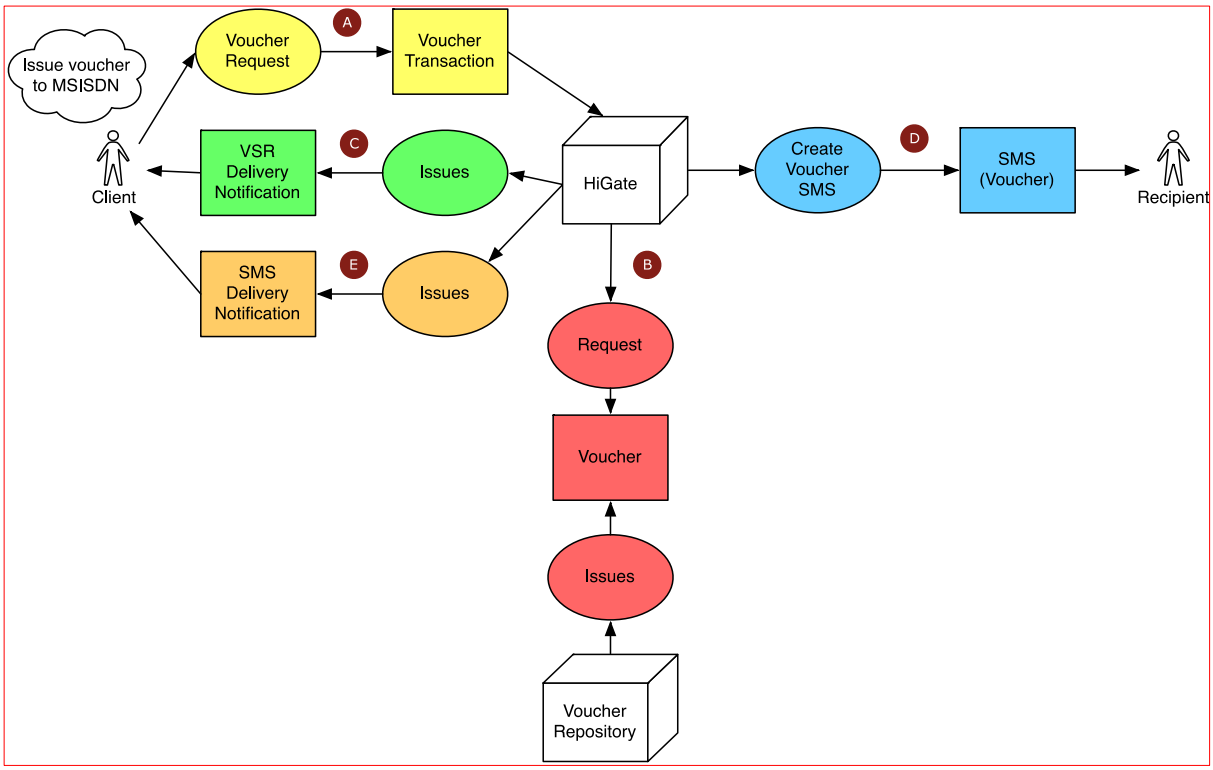
HTTP Message Flow



9.1 System Overview

The content flow diagram below illustrates the steps in the voucher issuing process.

Voucher Transaction Content Flow



A. The client application submits a VSR transaction containing a voucher request XML document to the HiGate system which creates a voucher transaction. If successful it provides the client application with a HiGate sequence number.

B. The Higate system requests a voucher from the voucher repository. If the voucher is network dependant it uses the supplied destination address to determine for which network the voucher should be issued.

C. The Voucher transaction status is returned to the client application.

D. If the voucher was successfully issued an SMS transaction is generated containing the voucher details as specified in the request document.

E. The status of the SMS delivery is returned to the client. Note that this delivery notification will have a different sequence number from the originating VSR transaction but will contain the same reference number as supplied by the originating VSR transaction.

9.2 Services

The following voucher services are currently available:

Higate API:

- Issue voucher with SMS delivery.
- Issue voucher and return to client.

9.3 Setup

Before any vouchers can be issued by an account the account has to be configured for the particular type of vouchers. The setup process is as follows:

1. Your Integrat account manager will arrange access to the voucher repository.
2. Your service code(s) will be configured to accept VSR transactions
3. The application may only utilize pre-authorized Service codes. Depending on how the Voucher Repository has been configured for this Account/Voucher/Network combination.

Note that test vouchers that have no value can be loaded for testing purposes.

9.4 API

There are two methods of issuing a voucher:

1. By submitting a VSR transaction directly to the Higate API which will issue the voucher and send it to the receiver via SMS according to a user customised template consisting of a number of keyword fields.
2. By submitting a VSR transaction directly to the Higate API which will issue the voucher and send it back to the client application which is then responsible for submitting the voucher.

9.4.1 Higate API

Vouchers are issued by submitting a VSR transaction to the Higate system. If issuing via SMS is configured (default) then a SMS transaction will be generated by the system automatically upon successful issuing of the voucher. The status of the SMS shall be send to the client application. The reference number used when requesting the voucher can be used to link the SMS transaction to the originating voucher transaction.

9.4.1.1 XML Template – Submit

Vouchers are requested by submitting an XML document containing the following elements:

- **Action** – (Optional). If this field is omitted the voucher is issued to the recipient via SMS. To return the voucher to the calling application instead includes this field with the value set to "*Get*" instead.
- **Payload** – (Optional). Depends on the nature of the product.
- **Source** – (Optional). This is for audit purposes and may include any meaningful string denoting the source of the original query. For instance it may include the USSD string that was dialled to initiate the request.
- **FaceValue** – The value of the associated mobile product. If this represents a monetary amount then it must be expressed in cents. Note that it may alternatively be a Volume amount depending on the nature of the product. It must lie in the range defined by the values '*ValueMin'* and '*ValueMax'* returned by '*QueryNetworkProducts'*.
- **Instruction** – An SMS template for the message to be sent to the recipient. This may be ignored if the application is expected to deliver the Voucher redemption details (instructions). See 'SMS Templates'.

XML Voucher Submit Document Format

```
<Voucher>
   <Action></Action>
    <Name></Name>
    <FaceValue></FaceValue>
    <Instruction></Instruction>
    <Source></Source>
    <Payload></Payload>
</Voucher>
```

9.4.1.2 XML Template – Response

When vouchers are requested with the "Get" action the voucher is returned to the application in the form of a VSR transaction with the voucher details in an XML structure.

XML Voucher Response Document Format

```
<Txn>
   <RefNo></RefNo>
   <Instruction></Instruction>
</Txn>
```

- **RefNo** – The reference number of the VSR transaction that requested the voucher.
- **Instruction** – The voucher details formatted according to the SMS Template rules. Note that this means that the client application needs to specify the template for the format of the reply message.

9.4.1.3 Instruction SMS Template

SMS templates are text strings defining the format of voucher redemption instructions to the subscriber. A typical example of such a template is as follows...

Please dial {**RDCODE** } to redeem your {**FACEVALUE**} airtime. Ref({**V_REF**}).

Note that text enclosed by braces {} represent place-holder text that will be replace by the Voucher Repository once the associated values are known. Note that the onus is on the application to ensure that valid parameters are used. If the Voucher Repository fails to identify a parameter, then it will remain in the message 'as is'.

Currently defined parameters include...

- **{RDCODE}** – The network specific USSD redeem code
- **{PIN}** – the assigned redemption PIN for this voucher
- **{FACEVALUE}** – the Face Value of the voucher in format :<currency symbol><rands>.<cents> where rands uses the comma currency format to denote thousands, millions etc. (E.g. R 15.00)
- **{V_REF}** – the unique Voucher Repository Reference number for this Voucher/Transaction
- **{A_REF}** – the client application specified Reference Number.

9.4.1.4 Higate HTTP/XML

The examples below show the minimum required elements needed to request a voucher via the HTTP interface. The voucher request content can be sent in plain text, hex encoded or base 64 encoded format.

Voucher / HEX Encoded Example

```
<?xml version="1.0"?>
<Message>
    <Request Type="SendVSR" RefNo="2">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <SendVSR ToAddr="0821234567">
         <Content Type="HEX">
3c54786e3e3c566f75636865723e3c4e616d653e41697274696d65546573743c2f4e616d653e3c4661636556616c75653e35303
03c2f4661636556616c75653e3c496e737472756374696f6e3e5468616e6b7320666f72207573696e67205768697a7a2120546f
2072656465656d20796f7572207b4641434556414c55457d2061697274696d6520766f7563686572206469616c207b5244434f
44457d3c2f496e737472756374696f6e3e3c536f757263652f3e3c5061796c6f61642f3e3c2f566f75636865723e3c2f54786e333
</Content>
       </SendVSR>
    </Request>
</Message>
```

When a VSR transaction is submitted in plain text format it is important to substitute the XML reserved characters with XML safe escape codes. The substitutions are listed in Table 1.
Text Encoding Example:

*Note the character substitutions.

HTTP Plain Text Example

```
<?xml version="1.0"?>
<Message>
    <Version Version="1.0" />
    <Request Type="SendVSR" RefNo="1">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <SendVSR ToAddr="0821234567">
         <Content Type="TEXT">
&lt;Txn&gt;&lt;Voucher&gt;&lt;Name&gt;AirtimeTest&lt;/Name&gt;&lt;FaceValue&gt;500&lt;/FaceValue&gt;&lt;Instruction&
gt;Thanks for using Integrat! To redeem your {FACEVALUE} airtime voucher dial
    {RDCODE}&lt;/Instruction&gt;&lt;Source/&gt;&lt;Payload/&gt;
    &lt;/Voucher&gt;&lt;/Txn&gt;</Content>
       </SendVSR>
     </Request>
</Message>
```

Successful Reply Example

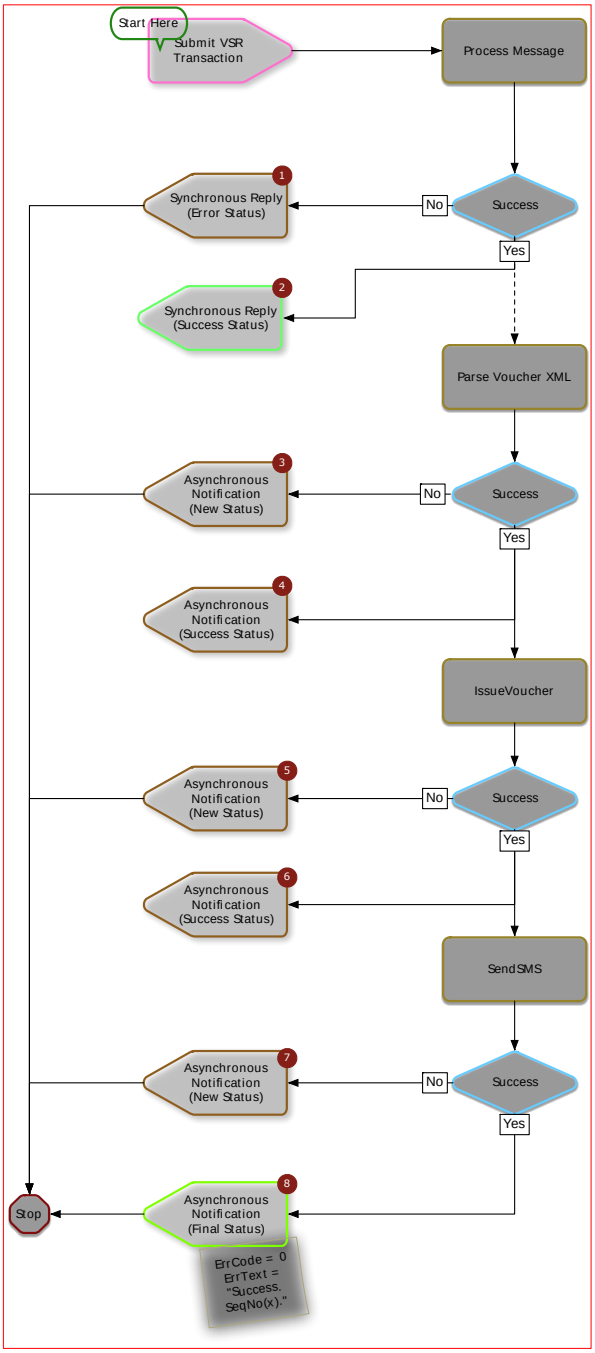
```
<?xml version="1.0"?>
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="1" />
      <field name="seq_no" value="12345678" />
   </Data>
</Response>
```

Failure Reply Example

```
<?xml version="1.0"?>
<Response token="" status_code="-1">
   <Data name="msg_nak">
      <field name="status" value="1" />
      <field name="error_text" value="&lt;error message&gt;" />
   </Data>
</Response>
```

9.5 Error Management

The flow chart in the figure below shows the steps in a typical voucher transactions.



There are four possible points of failure in a VSR transaction:

- Failure on submit (Refer to 1 in Figure).
- Failure on voucher XML parsing (Refer to 3 in Figure).
- Failure on voucher issuing (Refer to 5 in Figure).
- Failure on voucher delivery (Refer to 7 in Figure).

9.5.1 Failure on Submit

In this case the failure code is returned directly in reply to submit transaction. The transaction is never accepted or recorded by the Higate system.

9.5.2 Failure on Voucher XML Parsing

In this case the submission was successful but the voucher XML could not be parsed. The error response will be in the form of a call back (HTTP interface) or delivery notification (SMPP interface)

9.5.3 Failure on Voucher Issuing

In this case the XML parsing was successful but the voucher could not be issued. The error response will be in the form of a callback (HTTP interface) or delivery notification (SMPP interface)

9.5.4 Failure on Voucher Delivery

In this case the voucher was successfully issued but delivery via SMS failed.

9.6 HTTP Example

9.6 Requesting a Voucher with Automatic SMS Delivery

9.6.1.1 Submit VSR Transaction

The example below shows a typical HTTP POST Message to request a voucher with automated delivery. Note the escaped XML in the Content field.

HTTP POST:

```
<Version Version="1.0" />
<Message>
    <Request Type="SendVSR" RefNo="9">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <SendVSR ToAddr="0820000000" Validity="00020000" Flags="0">
          <Content Type="TEXT">
&lt;Txn&gt;&lt;Voucher&gt;&lt;Name&gt;AirtimeTest&lt;/Name&gt;&lt;FaceValue&gt;500&lt;/FaceValue&gt;&lt;Instruction&
gt;Thanks for using Integrat! To redeem your {FACEVALUE} airtime voucher dial
{RDCODE}&lt;/Instruction&gt;&lt;Source/&gt;&lt;Payload/&gt;&lt;/Voucher&gt;&lt;/Txn&gt;&lt;/Content&gt;</Content>
          </SendVSR>
    </Request>
</Message>
```

A successful submit will result in a synchronous response from the server providing the assigned unique sequence number. This sequence number is included in all related call back messages relating to this VSR Transaction.

HTTP Reply:

```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="1" />
      <field name="seq_no" value="2620247709" />
   </Data>
</Response>
```

9.6.1.2 VSR Call Back

A number of call backs will be received for a successfully submitted VSR transaction. The type of response will be '*OnResult*' with a TOC of "*VSR*". The "Code" attribute of the '*OnResult*' parameter identifies the status of the transaction. Refer to the online Higate documentation for the meaning of the different result codes.

The below call back indicates the "*Receipted*" status of the VSR transaction.

HTTP Call Back (Receipted)

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="9" SeqNo="2620247709">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <ErrCode>0</ErrCode>
       <ErrText>Success. SMS SeqNo(2620247727)</ErrText>
       <OnResult Flags="0" Code="4" SubCode="0"
       Text="Success. SMS SeqNo(2620247727)" />
    </Response>
</Message>
```

The below call back indicates the "Submitted" status of the SMS transaction containing the voucher details. Note that this transaction has a different sequence number from the originating VSR transaction but has the same reference number, which can be used to link the transactions together.

HTTP Call back (Submitted)

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="SMS" RefNo="9" SeqNo="2620247727">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <ErrCode>0</ErrCode>
       <ErrText></ErrText>
       <OnResult Flags="0" Code="2" SubCode="0" Text="Submitted" />
  </Response>
</Message>
```

The below call back indicates the "*Acknowledged*" status of the SMS transaction

HTTP Call back (Acknowledged)

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="SMS" RefNo="9" SeqNo="2620247727">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText></ErrText>
        <OnResult Flags="0" Code="3" SubCode="0" Text="Acknowledged" />
    </Response>
</Message>
```

The below call back indicates the final "*Receipted*" status of the SMS transaction.

HTTP Call back (Receipted)

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="SMS" RefNo="9" SeqNo="2620247727">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText>Receipted</ErrText>
        <OnResult Flags="0" Code="4" SubCode="0" Text="Receipted" />
    </Response>
</Message>
```

*** Note:** The reception of intermediate status notifications is not guaranteed. The system always transmits the latest status of a transaction and might skip over intermediate notifications should the status change quickly. The only notification that is guaranteed is the final status notifications.

9.6.2 Requesting a Voucher with Contents Returned to Application

9.6.2.1 Submit VSR Transaction

The example below shows a typical HTTP POST Message to request a voucher with delivery back to the WASP system.

HTTP POST:

```
<Version Version="1.0" />
<Message>
    <Request Type="SendVSR" RefNo="20">
        <UserID>USERNAME</UserID>
        <Password>SERVICECODE</Password>
        <SendVSR ToAddr="0820000000" Validity="00020000" Flags="0">
            <Ticket Type="Mobile" Service="serv" />
            <Content Type="TEXT">&lt;Txn&gt;&lt;Action&gt;Get&lt;/Action&gt;
&lt;Voucher&gt;&lt;Name&gt;AirtimeTest&lt;/Name&gt;&lt;FaceValue&gt;500&lt;/FaceValue&gt;&lt;Instruction&gt;Thanks
for using Integrat! To redeem your {FACEVALUE} airtime voucher dial
{RDCODE}&lt;/Instruction&gt;&lt;Source/&gt;&lt;Payload/&gt;&lt;/Voucher&gt;&lt;/Txn&gt;</Content>
        </SendVSR>
    </Request>
</Message>
```

HTTP POST Reply:

```
<Response token="" status_code="0">
    <Data name="msg_generic_rsp">
        <field name="msg_no" value="1" />
        <field name="seq_no" value="2620830216" />
    </Data>
</Response>
```

9.6.2.2 VSR Call Backs

A number of call backs will be received for a successfully submitted VSR transaction. The type of response will be '*OnResult*' with a TOC of "*VSR*". The "*Code*" attribute of the '*OnResult*' parameter identifies the status of the transaction. Refer to the online Higate documentation for the meaning of the different result codes.

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="20" SeqNo="2620830216">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText>Success</ErrText>
        <OnResult Flags="0" Code="4" SubCode="0" Text="Success" />
    </Response>
</Message>
```

For this type of request the voucher detail is returned to the application as a VSR transaction originating from the Higate system. The type of message shall be "*OnReceiveVSR*" and the content shall contain the voucher data in XML

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnReceiveVSR">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <Network ID="1" MCC="655" MNC="001" />
        <OnReceiveVSR SeqNo="88961047" Sent="20140509110641"
          FromAddr="27000000000" ToAddr="27820048062" ToTag="216"
          Value="0" NetworkID="1" AdultRating="0" DataCoding="0"
          EsmClass="0">
        <Content Type="TEXT">&lt;Txn&gt;&lt;RefNo&gt;20&lt;/RefNo&gt;&lt;Instruction&gt;Thanks for using Integrat! To
redeem your R5.00 airtime voucher dial *100*1234567890110214#&lt;/Instruction&gt;&lt;/Txn&gt;</Content>
        </OnReceiveVSR>
     </Response>
</Message>
```

In this example the escaped content:

*<Txn><RefNo>20</RefNo><Instruction>Thanks for using Integrat! To redeem your R5.00 airtime voucher dial *100*1234567890110214#</Instruction></Txn>*

Translates to the following XML:

```
<Txn>
   <RefNo>20</RefNo>
    <Instruction>Thanks for using Integrat! To redeem your R5.00 airtime
      voucher dial *100*1234567890110214#</Instruction>
</Txn>
```

9.6.3 Common Error Messages

The following section lists some of the common error message that can be expected when submitting a VSR transaction.

9.6.3.1 Duplicate Reference Number

Accounts are setup by default not to allow vouchers to be issued with duplicate VSR transaction reference numbers. This safety mechanism prevents duplicate vouchers to be submitted by accident. In the event that a transaction is submitted that contains a reference number that was recently used - the system will return an error response as listed below.

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="9" SeqNo="2620290613">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <ErrCode>0</ErrCode>
       <ErrText>Duplicate RefNo</ErrText>
       <OnResult Flags="0" Code="6" SubCode="0"
        Text="Duplicate RefNo" />
    </Response>
</Message>
```

9.6.3.2 Wrong Service Code (Could not be routed)

VSR transactions is usually linked to a certain service code. Should the service code used not be configured to send VSR transactions then the following error message can be expected as a response to the original POST transaction.

HTTP Reply:

```
<Response token="" status_code="-1">
   <Data name="msg_nak">
      <field name="status" value="260" />
      <field name="error_text"
       value="No routing defined for service(...) and TOC(9)." />
  </Data>
</Response>
```

9.6.3.3 Invalid Voucher Name

Each service has a number of voucher names that the service can use. Should the name specified not exist then the following message is returned:

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="10" SeqNo="2620636884">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <ErrCode>0</ErrCode>
       <ErrText>Unknown or disabled voucher</ErrText>
       <OnResult Flags="0" Code="6" SubCode="0"
         Text="Unknown or disabled voucher" />
    </Response>
</Message>
```

9.6.3.4 Invalid XML

The voucher XML included in the content field is only validated after the VSR transaction has been submitted. If the XML is identified as invalid at that point the following message is returned:

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="13" SeqNo="2620756869">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText></ErrText>
        <OnResult Flags="0" Code="6" SubCode="0" Text="Failed" />
    </Response>
</Message>
```

9.6.3.5 No Stock Available

Should there be no stock available of the requested voucher type or denomination then the following message will be returned:

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="16" SeqNo="2620785926">
        <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText>No stock available</ErrText>
        <OnResult Flags="0" Code="6" SubCode="0"
          Text="No stock available" />
     </Response>
</Message>
```

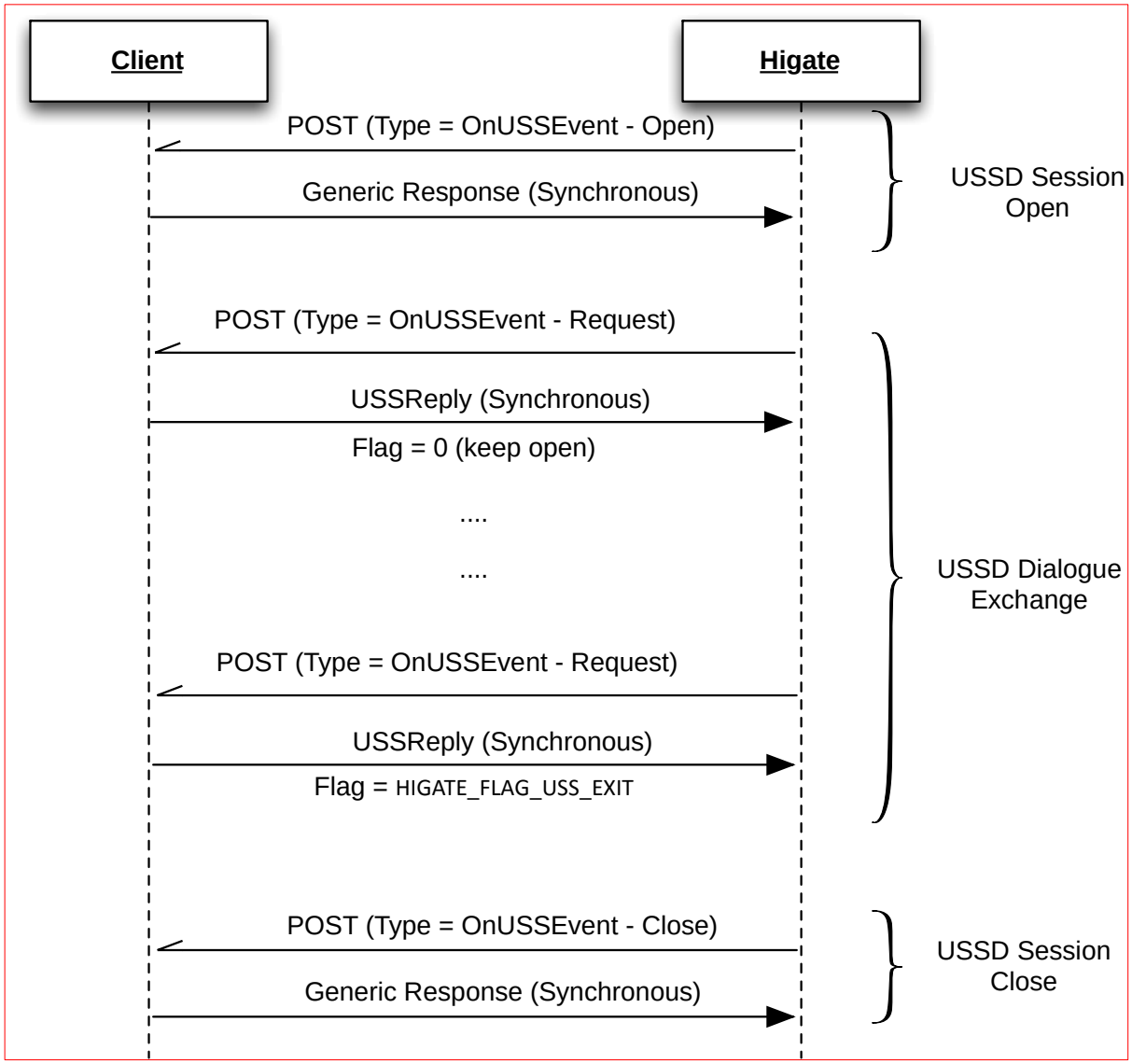
9.6.3.6 Invalid Action

The action field can be used to specify certain actions to be performed, such as returning the voucher to the calling application. If this action is specified incorrectly the following error message will be returned:

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnResult" TOC="VSR" RefNo="18" SeqNo="2620823829">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <ErrCode>0</ErrCode>
        <ErrText>Invalid Action</ErrText>
        <OnResult Flags="0" Code="6" SubCode="0"
          Text="Invalid Action" />
    </Response>
</Message>
```

10. USSD (TOC_USS)



USSD (Mobile Originating) is a real time service that requires instant responses to dialogue requests. It is highly recommended that the USSD service be configured on a dedicated login to separate it from any other bearers like SMS, OBS and VSR. The call back interface has limited capacity and the delivery of real time USSD requests can be influenced by for example bulk SMS messaging. The resulting delays can cause the USSD session to expire before the reply is received.

Because the call back interface has dedicated channels for each login that will prevent call back traffic for other bearers delaying USSD related traffic.

Note that you can only respond on a "*Request*" and not on "*Open*" and "*Close*" events. Also note that the reply can be synchronous (directly on the call-back) or asynchronous. Asynchronous requests should be used if there is a delay between receiving the call back and having the response message available.

10.1 HTTP Example

10.1.1 Standard USSD Session Example

10.1.1.1 USSD Open Transaction

USSD Open:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnUSSEvent">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <Network ID="1" MCC="" MNC="" />
        <OnUSSEvent Type="Open">
            <USSContext SessionID="12345678" NetworkSID="2576443669"
           MSISDN="27820000000" Script=""
           ConnStr="*120*99*123#"/>
           <USSText Type="TEXT"></USSText>
       </OnUSSEvent>
    </Response>
</Message>
```

10.1.1.2 USSD Request / Response Transaction

USSD Request:

```
<Version Version="1.0" />
<Message>
    <Response Type="OnUSSEvent">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <Network ID="1" MCC="655" MNC="001" />
        <OnUSSEvent Type="Request">
           <USSContext SessionID="12345678" NetworkSID="2576443669"
           MSISDN="27820000000" Script=""
           ConnStr="*120*99*123#" />
           <USSText Type="TEXT">REQ</USSText>
        </OnUSSEvent>
     </Response>
</Message>
```

USSD Response:

```
<Version Version="1.0" />
<Message>
    <Request Type="USSReply" SessionID="12345678" Flags="0">
       <UserID Orientation="TR">USERNAME</UserID>
       <Password>PASSWORD</Password>
       <USSText Type="TEXT">Welcome the this USSD session</USSText>
     </Request>
</Message>
```

10.1.1.3 USSD Close Session

USSD Close:

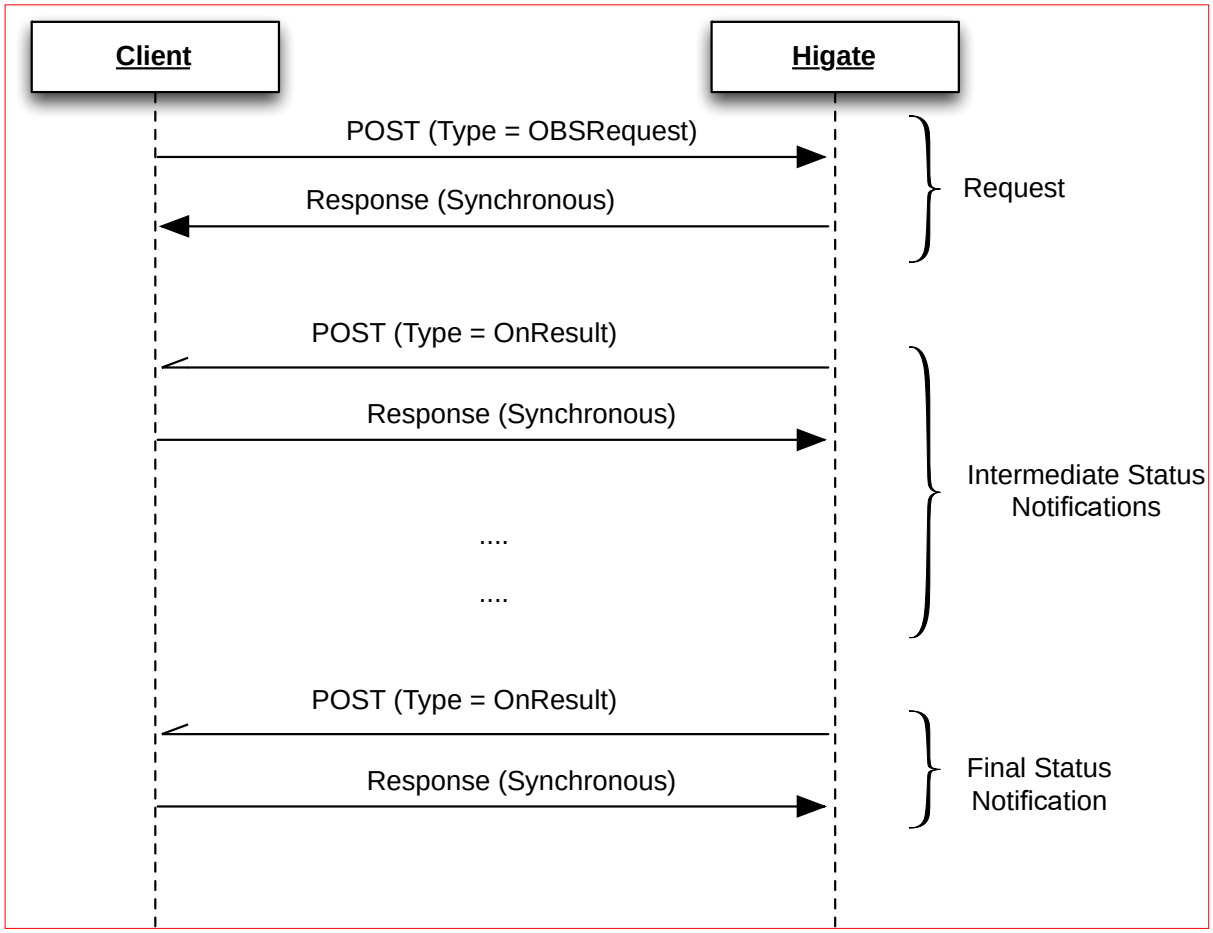
```
<?xml version="1.0"?>
<Message>
    <Response Type="OnUSSEvent">
      <SystemID>Higate</SystemID>
      <UserID>USERNAME</UserID>
      <Service>SERVICECODE</Service>
      <Network ID="1" MCC="" MNC="" />
      <OnUSSEvent Type="Close">
        <USSContext SessionID="12345678" NetworkSID="2576443669"
        MSISDN="27820000000" Script=""
        ConnStr="*120*99*123#"/>
        <USSText Type="TEXT"></USSText>
       </OnUSSEvent>
    </Response>
</Message>
```

A USSD session can be closed for various reasons. The CLOSE event is used to notify the client that a session has been closed. The timeout between requests and responses from both the application and handset sides should never exceed 10 seconds or the session will time out.

The following are typical scenarios where a session has been closed:

- **Terminated by subscriber** when the clients selects Exit instead of Reply on the handset
- **Handset took too long to respond**
- **Forced closed by application** by setting the HIGATE_FLAG_USS_EXIT flag in the reply.
- **Network (Time To Live) TTL expiry**. A USSD session has a limited time to live. This value is network specific but is typically 120 seconds.
- **Application is too slow in responding** – if no reply is received on a request the Higate system will force the session closed with a generic error message. This is to prevent the network operator from forcing the session closed with a cryptic error message.

11. OBS (TOC_OBS)



Mobile operators provides for billing users for once off and subscription services. Both billing models require user consent to being billed via a double-opt-in (DOI) system. DOI is described in detail in Integrat **Higate Concepts Manual** found on the Intarget Support page**:**

http://integrat.freshdesk.com/solution/articles/4000053161-higate-concepts

OBS essentially consist of three parts:

1. Product Subscription or Ado – where the user has to opt-in to being billed for a service.
2. Recurring billing – periodic billing transactions for subscription services.
3. Termination of subscriptions.

All three steps are performed by submitting an OBS Request transaction.

Special case – Subscription Only

Normally the OBS transaction that triggers the double-opt-in (subscription start) would also perform a billing transaction on successful opt-in. In some cases this might not be desirable, for example where a user is opt in on a "per event" service where no event has yet occurred that requires billing. In this case the HIGATE_FLAG_OBS_SUBSCRIBE_ONLY flag can be set which will prevent the user by the subscription start OBS request transaction.

Special case – Termination

Note that terminating a subscription will not result in a billing transaction, even if a value has been submitted. The value of successful terminating transactions is always set to 0 cents, although the transaction will be recorded as a successful OBS request. It is therefore important for the client to take this into account when calculating revenues.

11.1 Subscription Start

Before a subscription can be started a corresponding product must exists in the system. New product can be created on the product portal (Refer to **Higate Concepts Manual** for more information).

A new subscription is triggered by specifying the "**FIRST**" keyword in the "*Started*" attribute of the "*Subscr*" parameter for the OBS Request.

The name of the product being subscribed to is contained in the "*Category*" attribute.

OBS Subscription Start

```
<Message>
   <Request Type="OBSRequest" RefNo="1">
      <OBSRequest Validity="00020000" Flags="0">
         <Subscr Started="2015-03-12 15:59:52 FIRST" Category="ExampleWeekly" />
      </OBSRequest>
    </Request>
</Message>
```

11.2 Recurring Billing

After successful opt-in from a subscriber recurring billing requests should be submitted without the "*FIRST*" keyword. It is very important to not include the "*FIRST*" keyword as all transactions containing this keyword will be submitted to the DOI system which will slow down the billing transaction considerably.

OBS Recurring Billing

```
<Message>
   <Request Type="OBSRequest" RefNo="1">
      <OBSRequest Validity="00020000" Flags="0">
         <Subscr Started="2015-03-12 15:59:52" Category="ExampleWeekly" />
      </OBSRequest>
    </Request>
</Message>
```

11.3 Subscription Termination

An existing subscription is terminated by submitting a transaction containing the "**STOP**" keyword in the Started attribute of the '*Subscr*' parameter

OBS Subscription Termination

```
<Message>
   <Request Type="OBSRequest" RefNo="1">
      <OBSRequest Validity="00020000" Flags="0">
         <Subscr Started="2015-03-12 15:59:52 STOP" Category="ExampleWeekly" />
      </OBSRequest>
    </Request>
</Message>
```

11.4 HTTP Examples

When performing OBS transactions the transaction can traverse through a number of intermediate states before reaching a final state. It is therefore important to always inspect the '*ResultCode*' parameter to determine the state of the transaction and not to rely on '*ErrCode*', '*ErrorText*' and '*ResultText*'. These parameters only have context if used together with the '*ResultCode*'.

11.4.1 Standard Billing Request

HTTP POST:

```
<Message>
    <Version Version="1.0" />
    <Request Type="OBSRequest" RefNo="1">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <OBSRequest Validity="00020000" Flags="0">
          <Ticket Type="Mobile" Service="SERVICECODE" ChargeAddr="0829034444"
          Value="500" />
          <Subscr Started="2015-03-12 15:59:52"
          Category="ExampleWeekly" />
       </OBSRequest>
     </Request>
</Message>
```

HTTP Response:

```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="1" />
      <field name="seq_no" value="2888195129" />
   </Data>
</Response>
```

11.4.2 Subscription Start Request

The following example shows a subscription start request. Note the inclusion of the "FIRST" keyword.

Subscription Start Request

```
<Message>
    <Version Version="1.0" />
    <Request Type="OBSRequest" RefNo="6">
       <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <OBSRequest Validity="00020000" Flags="512">
          <Ticket Type="Mobile" Service="SERVICECODE" ChargeAddr="0767931435"
          Value="500" />
          <Subscr Started="2015-03-12 15:59:52 FIRST"
          Category="ExampleWeekly" />
       </OBSRequest>
    </Request>
</Message>
```

HTTP Response:

```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="4" />
      <field name="seq_no" value="2888198596" />
   </Data>
</Response>
```

In this example, the opt-in was declined:

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnOBSResponse" RefNo="6" SeqNo="2888198596">
       <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <Flags>512</Flags>
       <ResultCode>6</ResultCode>
       <ResultText>[E501A0]Double Opt In : Declined</ResultText>
       <ErrCode>2147483915</ErrCode>
       <ErrText>[E501A0]Double Opt In : Declined</ErrText>
       <OnOBSResponse Type="XML">
          <OBS>
             <Action>AUTHORIZE</Action>
             <Param>
                <BillingCode></BillingCode>
                <AdultRating>0</AdultRating>
                <Category>ExampleWeekly</Category>
                <Descr></Descr>
                <ItemID></ItemID>
                <RxSeqNo>0</RxSeqNo>
                <Other>
                   <Vodacom>
                      <Service>INT00017</Service>
                      <Subscr>
                         <Started>2015-03-12 15:59:52 FIRST</Started>
                      </Subscr>
                      <Product Name='ExampleWeekly' ID='315'
                      BillingFreq='Week' ServiceID='INT00315'
                      Activation='2011-11-09' />
                   </Vodacom>
                </Other>
             </Param>
             <Subscr Backbill='' Category='ExampleWeekly'
             Started='2015-03-12 15:59:52 FIRST' Trigger='' />
          </OBS>
       </OnOBSResponse>
    </Response>
</Message>
```

In this example the opt-In was accepted. Note that the result text would indicate the status if the OBS transaction that was part of the subscription transaction. It is still possible that the transaction can fail with an “Insufficient funds” result which would still imply successful opt-in.

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
   <Response Type="OnOBSResponse" RefNo="7" SeqNo="2888199140">
      <SystemID>Higate</SystemID>
       <UserID>USERNAME</UserID>
       <Service>SERVICECODE</Service>
       <NetworkID>1</NetworkID>
       <Network ID="1" MCC="655" MNC="001" />
       <Flags>512</Flags>
       <ResultCode>4</ResultCode>
       <ResultText>[E1A0]Success</ResultText>
       <ErrCode>260</ErrCode>
      <ErrText>[E1A0]Success</ErrText>
      <OnOBSResponse Type="XML">
         <OBS>
            <Action>CONFIRMED</Action>
            <Result>0</Result>
            <ResultText>Success</ResultText>
            <Param>
               <BillingCode />
                <AdultRating>0</AdultRating>
                <Category>ExampleWeekly</Category>
                <Descr />
                <ItemID />
                <RxSeqNo>0</RxSeqNo>
                <Other>
                   <Vodacom>
                      <Service>INT00017</Service>
                      <Subscr>
                         <Started>2015-03-12 15:59:52 FIRST</Started>
                      </Subscr>
                      <Product Name="ExampleWeekly" ID="315"
                      BillingFreq="Week" ServiceID="INT00315"
                      Activation="2011-11-09" />
                      <Hash>DONE</Hash>
                   </Vodacom>
                </Other>
            </Param>
            <Subscr Backbill="" Category="ExampleWeekly"
            Started="2015-03-12 15:59:52 FIRST" Trigger="" />
         </OBS>
      </OnOBSResponse>
   </Response>
</Message>
```

11.4.3 Subscription Recurring Billing

Once a subscriber has been opted in, subsequent billing requests should not include the "**FIRST**" keyword. The "*Started*" date should be date when the opt-in was triggered.

HTTP POST:

```
<Message>
<Version Version="1.0" />
   <Request Type="OBSRequest" RefNo="8">
      <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <OBSRequest Validity="00020000" Flags="512">
          <Ticket Type="Mobile" Service="SERVICECODE" ChargeAddr="0767931435"
          Value="1" />
          <Subscr Started="2015-03-12 15:59:52"
          Category="ExampleWeekly" />
       </OBSRequest>
   </Request>
</Message>
```

HTTP Response:

```
<Response token="" status_code="0">
   <Data name="msg_generic_rsp">
      <field name="msg_no" value="4" />
      <field name="seq_no" value="2888214477"/>
   </Data>
</Response>
```

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnOBSResponse" RefNo="7" SeqNo="2888214477">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <Flags>512</Flags>
        <ResultCode>4</ResultCode>
        <ResultText>[E1A0]Success</ResultText>
        <ErrCode>260</ErrCode>
       <ErrText>[E1A0]Success</ErrText>
       <OnOBSResponse Type="XML">
          <OBS>
             <Action>CONFIRMED</Action>
             <Result>0</Result>
             <ResultText>Success</ResultText>
             <Param>
                <BillingCode />
                 <AdultRating>0</AdultRating>
                 <Category>ExampleWeekly</Category>
                 <Descr />
                 <ItemID />
                 <RxSeqNo>0</RxSeqNo>
                 <Other>
                    <Vodacom>
                       <Service>INT00017</Service>
                       <Subscr>
                          <Started>2015-03-12 15:59:52 FIRST</Started>
                       </Subscr>
                       <Product Name="ExampleWeekly" ID="315"
                       BillingFreq="Week" ServiceID="INT00315"
                       Activation="2011-11-09" />
                       <Hash>DONE</Hash>
                    </Vodacom>
                 </Other>
              </Param>
              <Subscr Backbill="" Category="ExampleWeekly"
              Started="2015-03-12 15:59:52 FIRST" Trigger="" />
           </OBS>
       </OnOBSResponse>
    </Response>
</Message>
```

11.4.4 Subscription Terminate Request

When a subscription termination request is received and the network operator requires manual termination of the subscription a "**STOP**" request is issued.

HTTP POST:

```
<Message>
<Version Version="1.0" />
   <Request Type="OBSRequest" RefNo="8">
      <UserID>USERNAME</UserID>
       <Password>PASSWORD</Password>
       <OBSRequest Validity="00020000" Flags="512">
          <Ticket Type="Mobile" Service="SERVICECODE" ChargeAddr="0767931435"
          Value="1" />
          <Subscr Started="2015-03-12 15:59:52 STOP"
          Category="ExampleWeekly" />
       </OBSRequest>
   </Request>
</Message>
```

HTTP Response:

```
<Response token="" status_code="0">
    <Data name="msg_generic_rsp">
       <field name="msg_no" value="2" />
       <field name="seq_no" value="2888197818"/>
    </Data>
</Response>
```

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnOBSResponse" RefNo="4" SeqNo="2888197818">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <Flags>512</Flags>
        <ResultCode>4</ResultCode>
        <ResultText>Successful Termination</ResultText>
        <ErrCode>0</ErrCode>
       <ErrText>Successful Termination</ErrText>
       <OnOBSResponse Type="XML">
          <OBS>
             <Action>AUTHORIZE</Action>
             <Result>0</Result>
             <ResultText>Success</ResultText>
             <Param>
                <BillingCode />
                 <AdultRating>0</AdultRating>
                 <Category>ExampleWeekly</Category>
                 <Descr />
                 <ItemID />
                 <RxSeqNo>0</RxSeqNo>
                 <Other>
                    <Vodacom>
                       <Service>INT00017</Service>
                       <Subscr>
                          <Started>2015-03-12 15:59:52 FIRST</Started>
                       </Subscr>
                       <Product Name="ExampleWeekly" ID="315"
                       BillingFreq="Week" ServiceID="INT00315"
                       Activation="2011-11-09" />
                    </Vodacom>
                 </Other>
             </Param>
             <Subscr Backbill="" Category="ExampleWeekly"
             Started="2015-03-12 15:59:52 STOP" Trigger="" />
          </OBS>
       </OnOBSResponse>
    </Response>
</Message>
```

11.4.5 Common Error Messages

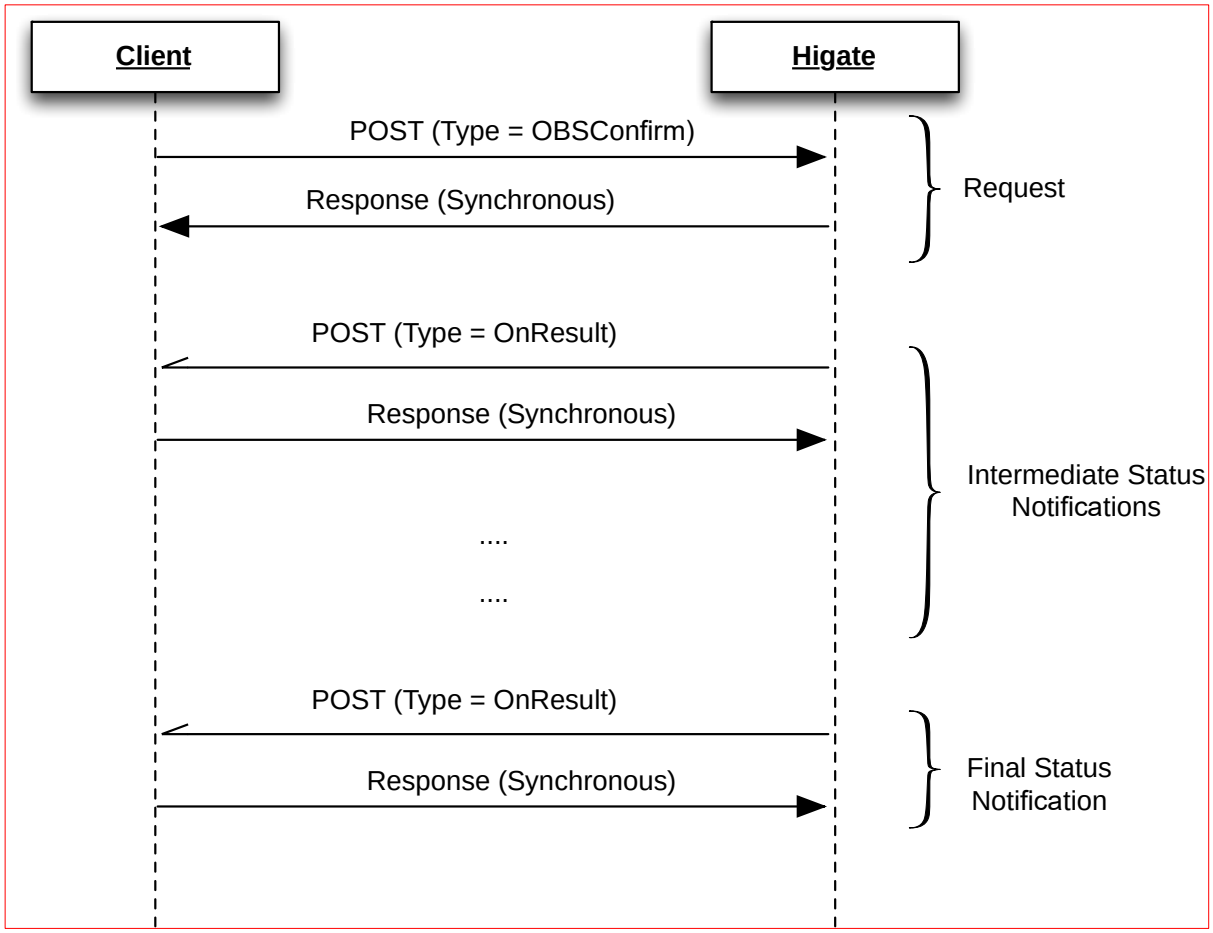
The following section lists some of the common error message that can be expected when submitting an OBS transaction.

11.4.5.1 Subscriber Not Subscribed

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnOBSResponse" RefNo="1" SeqNo="2888195129">
       <SystemID>Higate</SystemID>
        <UserID>USERNAME</UserID>
        <Service>SERVICECODE</Service>
        <NetworkID>1</NetworkID>
        <Network ID="1" MCC="655" MNC="001" />
        <Flags>0</Flags>
        <ResultCode>6</ResultCode>
        <ResultText>[E504A0]Double Opt In : Subscriber is not subscribed</ResultText>
        <ErrCode>3221229472</ErrCode>
        <ErrText>[E504A0]Double Opt In : Subscriber is not subscribed</ErrText>
        <OnOBSResponse Type="XML">
           <OBS>
              <Action>Failed</Action>
              <Result>4000</Result>
              <ResultText>Subscriber is not subscribed to this service</ResultText>
              <Param>
                 <BillingCode></BillingCode>
                 <AdultRating>0</AdultRating>
                 <Category>ExampleWeekly</Category>
                 <Descr></Descr>
                 <ItemID></ItemID>
                 <RxSeqNo>0</RxSeqNo>
                 <Other>
                    <Vodacom>
                       <Service>INT00017</Service>
                       <Subscr>
                          <Started>2015-03-12 15:59:52</Started>
                       </Subscr>
                       <Product Name='ExampleWeekly' ID='315'
                       BillingFreq='Week' ServiceID='INT00315'
                       Activation='2011-11-09' />
                    </Vodacom>
                 </Other>
              </Param>
              <Subscr Backbill='' Category='ExampleWeekly'
              Started='2015-03-12 15:59:52' Trigger='' />
           </OBS>
        </OnOBSResponse>
    </Response>
</Message>
```

11.5 OBS Confirm



The Vodacom Telco requires a confirmation transaction to be issued in addition to the normal billing transaction. The purpose of this method is to allow billing transactions to only be confirmed after the service billing transaction is linked to deliver. If the confirmation step is not performed and automatic confirmation is not enabled the billing transaction will expire after a few hours.

The Higate system automatically manages this confirmation step if the **HIGATE_FLAG_OBS_AUTO_CONFIRM** (512) flag is set when submitting the transaction. Should this flag not be set the transaction will remain in the "Approved" state until a corresponding "Confirm" transaction is issued. Failure to submit this will result in the billing transaction being cancelled after a few hours. It is always recommended to use the HIGATE_FLAG_OBS_AUTO_CONFIRM option instead of manually confirming the transactions.

11.6 HTTP Examples

11.6.1 Confirming an OBS Transaction

OBS Confirm

```
<Message>
<Version Version="1.0"/>
   <Request Type="OBSConfirm">
      <UserID>USERNAME</UserID>
      <Password>PASSWORD</Password>
      <OBSConfirm SeqNo="2866342797" Done="Y"/>
    </Request>
</Message>
```

HTTP Call Back:

```
<?xml version="1.0"?>
<Message>
    <Response Type="OnOBSResponse" RefNo="11" SeqNo="2866342797">
      <SystemID>Higate</SystemID>
      <UserID>USERNAME</UserID>
      <Service>SERVICECODE</Service>
      <NetworkID>1</NetworkID>
      <Network ID="1" MCC="655" MNC="001" />
      <Flags>0</Flags>
      <ResultCode>1</ResultCode>
      <ResultText>[E1A0]Success</ResultText>
      <ErrCode>257</ErrCode>
      <ErrText>[E1A0]Success</ErrText>
      <OnOBSResponse Type="XML">
        <OBS>
          <Action>CONFIRM</Action>
          <Param>
            <BillingCode />
            <AdultRating>0</AdultRating>
            <Category>ExampleWeekly</Category>
            <Descr />
            <ItemID />
            <RxSeqNo>0</RxSeqNo>
            <Other>
              <Vodacom>
                <Service>INT00017</Service>
                <Subscr>
                  <Started>2015-02-10 15:59:52 FIRST</Started>
                </Subscr>
                <Product Name="ExampleWeekly" ID="315"
                BillingFreq="Week" ServiceID="INT00315"
                Activation="2011-11-09" />
                <Hash>DONE</Hash>
              </Vodacom>
            </Other>
          </Param>
          <Subscr Backbill="" Category="ExampleWeekly"
          Started="2015-02-10 15:59:52 FIRST" Trigger="" />
        </OBS>
      </OnOBSResponse>
    </Response>
</Message>
```