tieto.com

Tieto Connection IP Connectivity Service description
Charging API 1.2.3
HTTP Interface

Change History

Version	Date	Description
1.0	1.3.2014	IP Billing interface
1.1	1.4.2014	Layout correction
1.2	1.5.2014	Additional examples
1.2.2	1.4.2015	TransactoinID, Price and three decimals
1.2.3	2.3.2017	Error Code insertion

TABLE OF CONTENTS

Abbreviations

Abbreviation	Specification
API	Application Programming Interface
CAPI	Charging API
CDR	Charging Detail Record
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IPB	IP Billing Proxy
MSISDN	Mobile Subscriber International ISDN Number
RFC	Request For Comments
SSL	Secure Sockets Layer
URL	Uniform Resource Locator
VAT	Value Added Tax

1. Introduction

Charging API provides transaction based charging by using HTTP messages.
Charging API interface is based on HTTP protocol version 1.1
Charging transactions use reserve / commit mechanism.

This document contains the definition of the HTTP interface to Charging API. The purpose of this document is to describe the HTTP interface in such detail it is possible to implement client applications that communicate with Charging API.

1.1   System overview

Overview of Charging API (CAPI) is shown in figure 1.1. CAPI provides a charging interface between a client application and Billing System or Prepaid System. The client connects to CAPI through HTTP connection. CAPI executes the request and returns the status of the request in response.
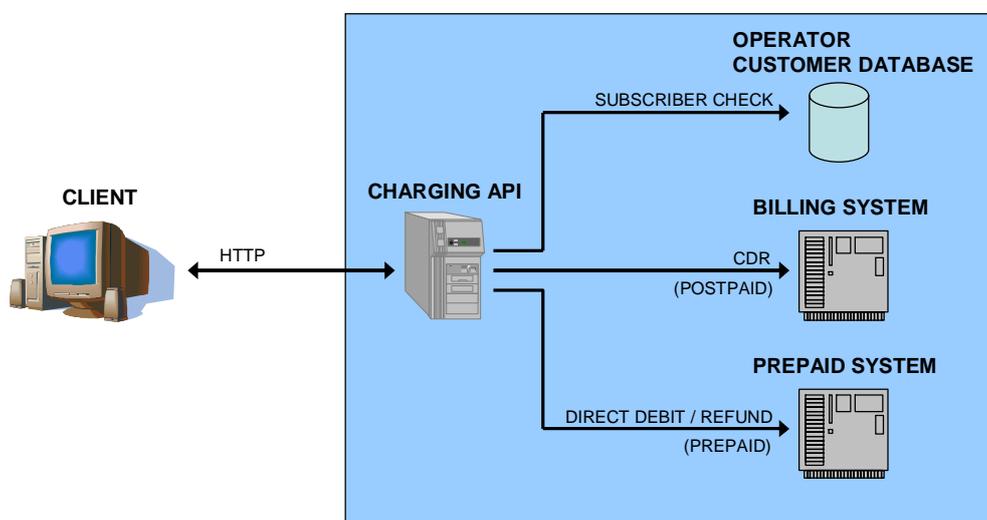

Figure 1.1. Overview of Charging API

Through CAPI interface the client can
- reserve funds from postpaid / prepaid MSISDN.
- charge or cancel reserved payments.
- execute direct debit on postpaid / prepaid MSISDN.

2. Interface Description

Client applications and CAPI communicate through HTTP connection. CAPI supports GET and POST methods for sending messages. Supported message format for GET method is URL query string. POST method supports HTTP headers and HTTP form data. Values of message parameters must be URL encoded. See appendix A for URL encoding and chapter 4 for examples.

CAPI supports both secured HTTP connections with SSL and unsecured plain HTTP connections. SSL connections do not use client certificates. Client is authenticated by using username/password fields in the message provided by client.

Table 2.1 describes accepted HTTP connection parameters for client application.

Table 2.1. Accepted HTTP connection parameters.

Parameter	Accepted values
Protocol	HTTP/1.1 or HTTP/1.0
Method	GET or POST
URL	vascenter.tieto.com/ipb/cpid/serviceid
TCP Port (unsecured HTTP)	8880
URL	vascenter.tieto.com/ipb/capi
TCP Port (secured HTTP)	8443

Figure 2.1 describes a complete Reserve/Commit transaction event. The first step to charge a MSISDN is to reserve funds. To reserve funds the client opens an HTTP connection to CAPI and sends a Reserve message. CAPI handles the request and returns a response. After the funds have been reserved from a MSISDN and the client is ready to complete the transaction, the transaction must be committed. Client commits the transaction by sending a Commit message. CAPI handles the request and returns a response. After the Commit message the transaction is complete.
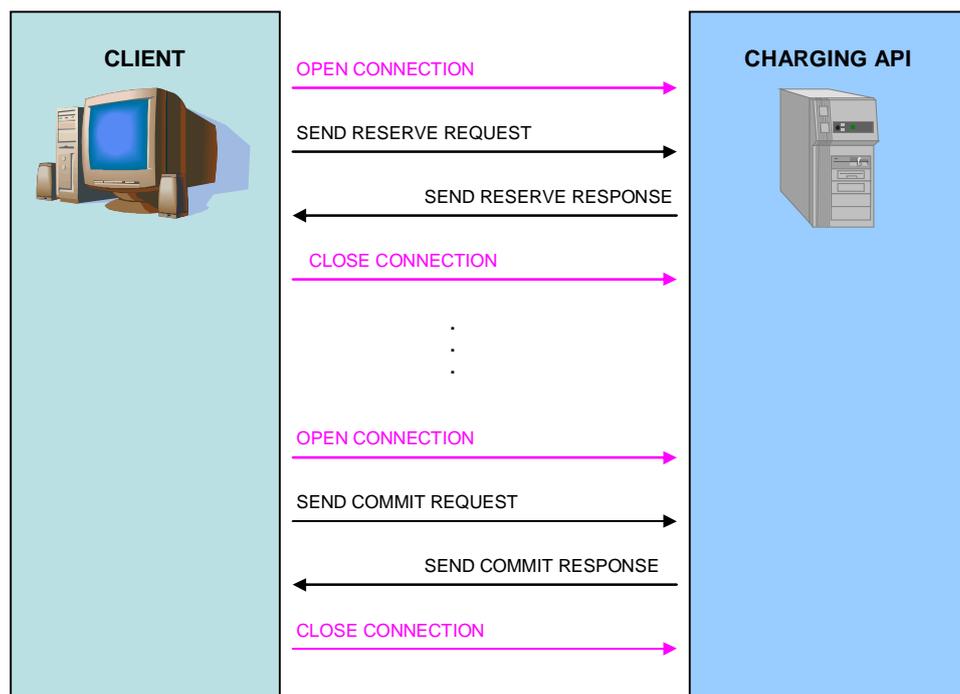
Figure 2.1. A complete Reserve/Commit transaction event using non-persistent connection

CAPI supports both persistent and non-persistent connections. Using persistent connection a client can send several Reserve/Commit messages using the same connection. Using non-persistent connection the client can close the connection after each request/response pair.

CAPI handles Reserve/Commit transactions asynchronously. This means a client can send several Reserve messages before sending Commit messages.

CAPI has a configurable timeout value for handling Reserve/Commit requests. By default the timeout is 3 seconds. The minimum allowed timeout value depends on the external systems. If CAPI is unable to complete the request before timeout, CAPI rejects the request and returns the reason in the response status code.

2.1 Reserve funds

Reserving funds is described in figure 2.2. Client opens an HTTP connection to CAPI and sends a Reserve message. CAPI verifies the client by using username/password and IP authentication. If the message is accepted, CAPI reserves funds from MSISDN. CAPI responds to the request with a response message that tells if the request was accepted or rejected. The HTTP status code of the response is always "200 OK", even if the request has been rejected. If the HTTP status code is not "200 OK", the client should retry the request.
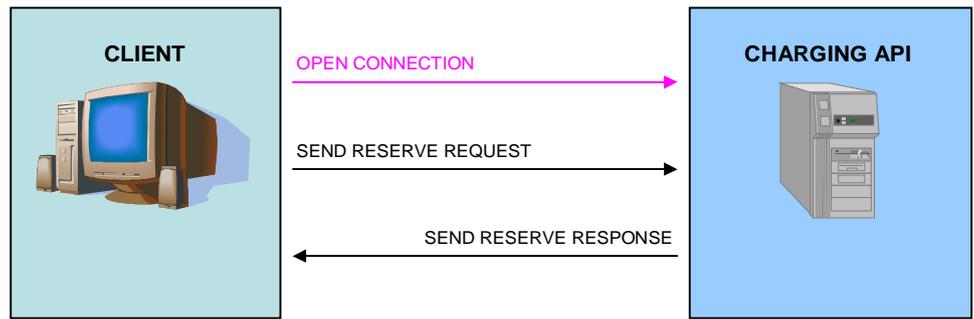
Figure 2.2. Reserve funds

2.1.1 Message flows of successful reservations

CAPI opens a new transaction for each successful fund reservation. The transaction must be closed by client by sending a Commit message. If Commit is not received before the reservation time of transaction, CAPI cancels and closes the transaction.

A successful reservation of funds from postpaid MSISDN is described in figure 2.3.
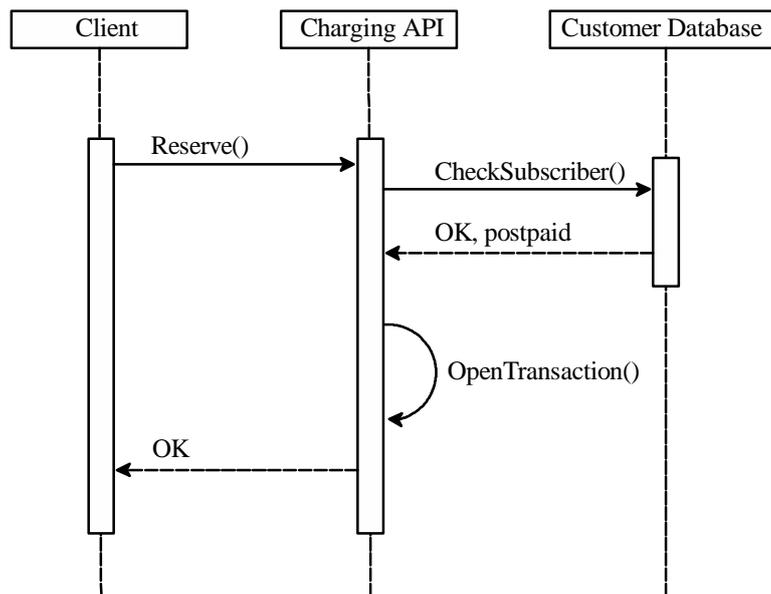
Figure 2.3. Successful reservation of funds from postpaid MSISDN

A successful reservation of funds from prepaid MSISDN is described in figure 2.4.
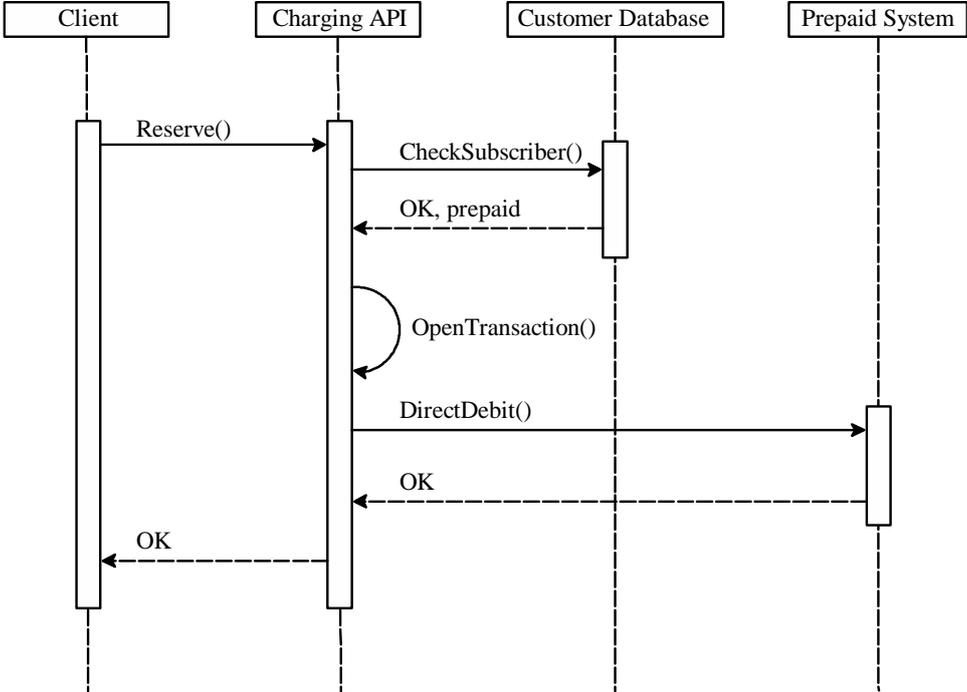
Figure 2.4. Successful reservation of funds from prepaid MSISDN

A successful but cancelled reservation of funds due to reservation timeout is described in figure 2.5 for postpaid MSISDN.
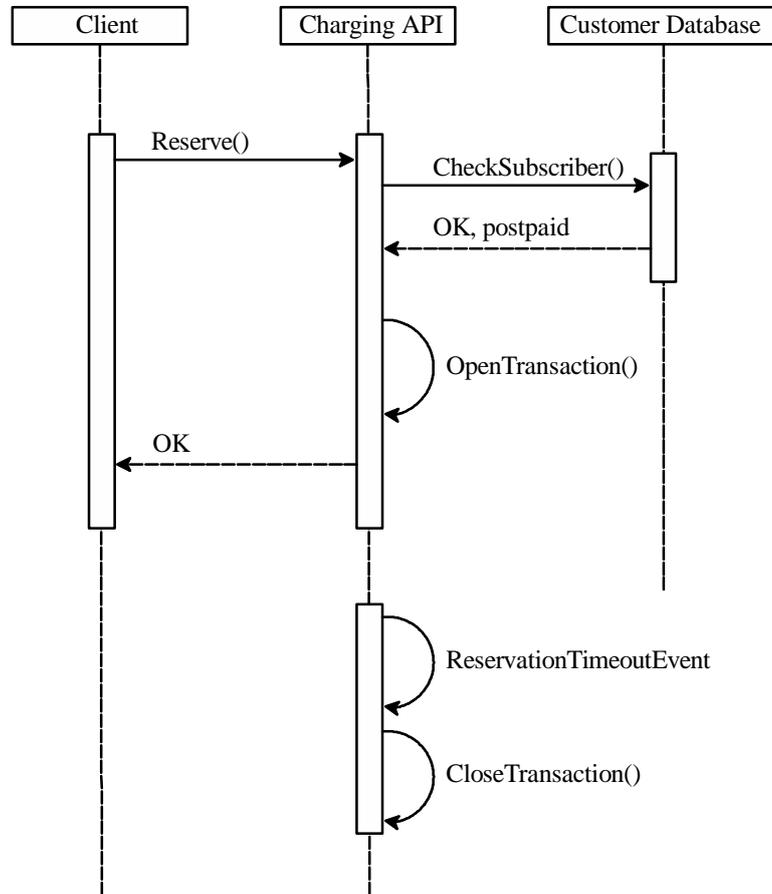
Figure 2.5. Cancelled reservation of funds due to reservation timeout

A successful but cancelled reservation of funds due to reservation timeout is described in figure 2.6 for prepaid MSISDN.
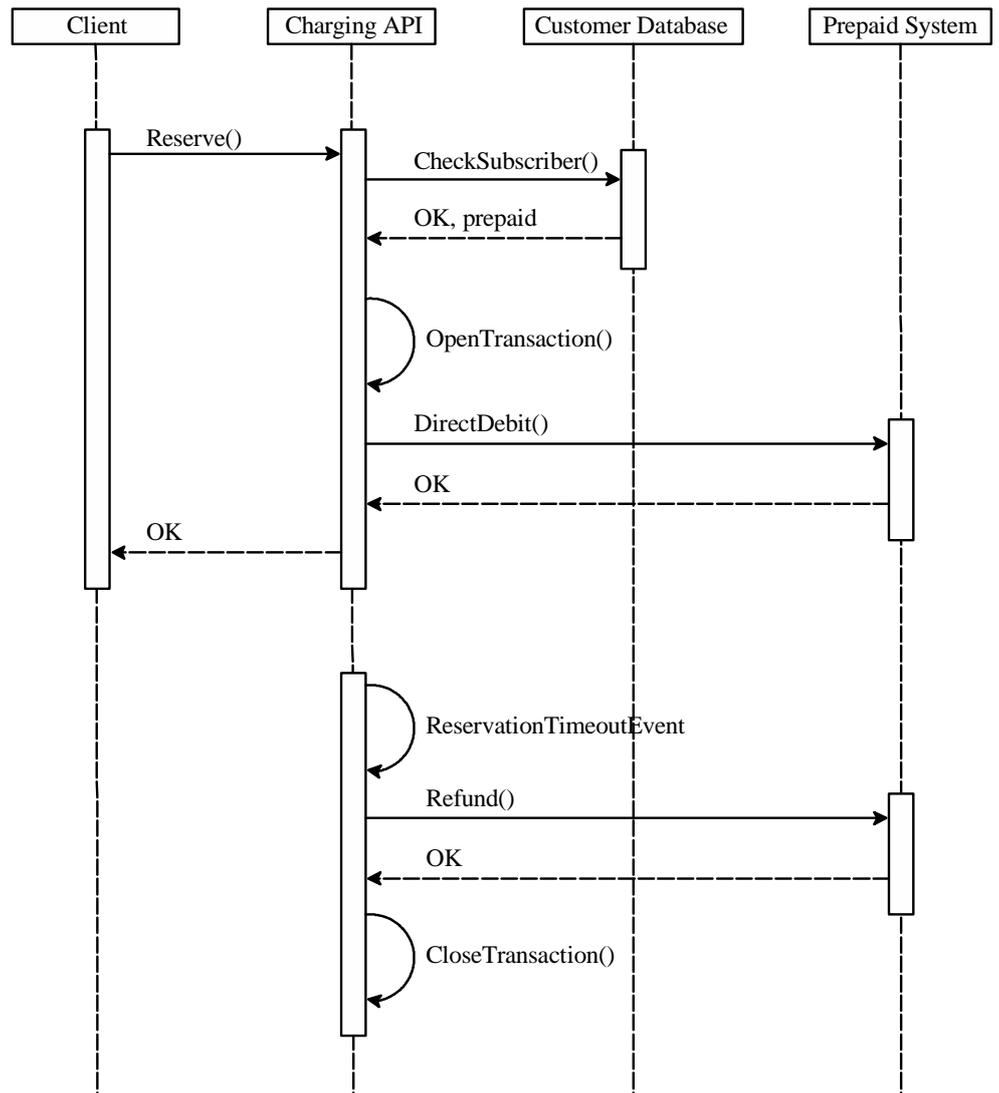
Figure 2.6. Cancelled reservation of funds due to reservation timeout

2.1.2 Message flows of failed reservations

The following figures describe different cases when the reservation of funds can fail.
If the reservation fails, CAPI rejects the request and closes the transaction.

Figure 2.6 describes a case when CAPI cannot obtain MSISDN information from the
Customer Database.
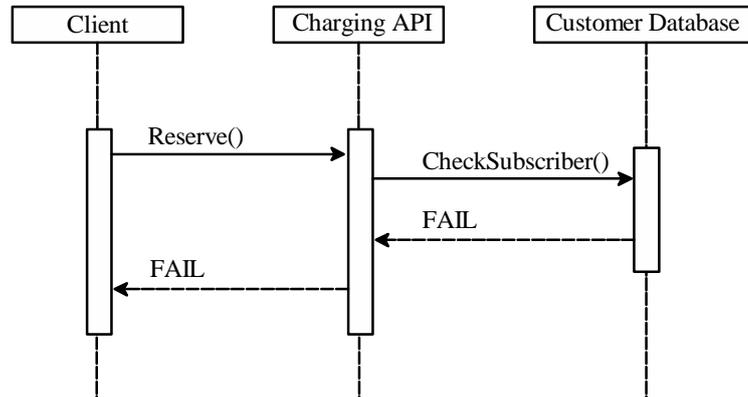
Figure 2.6. Rejected reservation of funds due to subscriber check failure

Figure 2.7 describes a case when CAPI cannot do direct debit for prepaid fund
reservation.
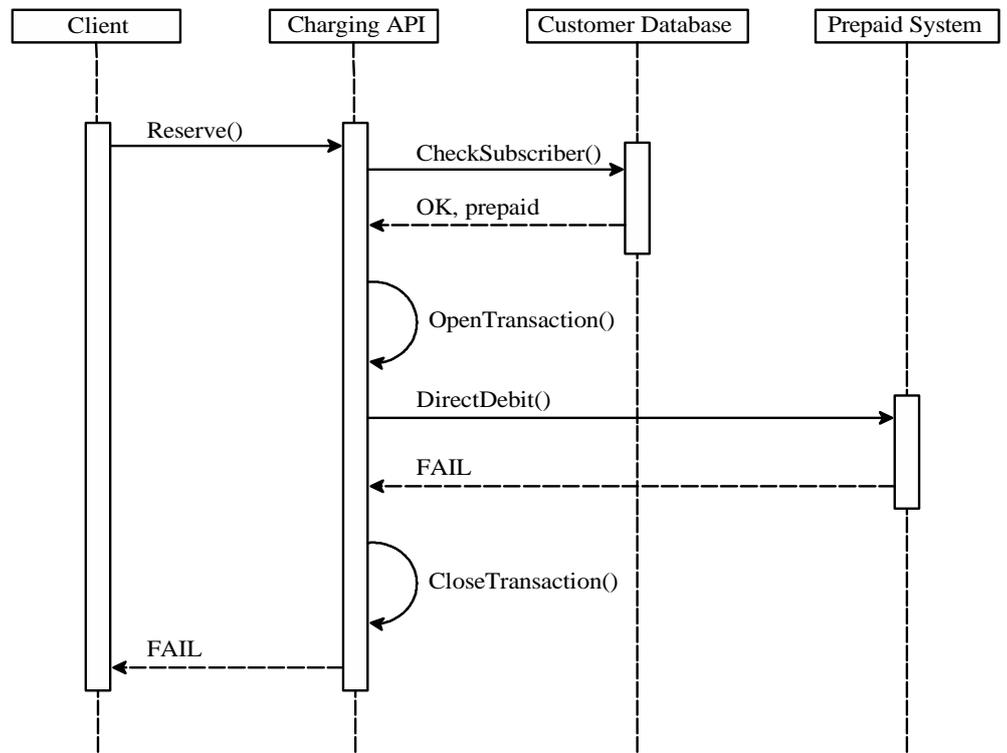
Figure 2.7. Rejected reservation of funds due to direct debit failure

2.1.3 Reserve request

Table 2.2 describes the parameters the client can supply when reserving funds. M/O column describes whether the parameter is mandatory or optional.

Table 2.2. Reserve request message.

| Parameter	M / O	Accepted values	Max length	Description
username	M	A-z, 0-9	32 chars	Username
password	M	A-z, 0-9	32 chars	Password
action	M	Reserve	7 chars	Type of the message
transactionid	M	A-z, 0-9	16 chars	Transaction ID Received from IP billing message.
serviceid	M	0-9	32 bit	Service ID
price	M	0-9, .	0-999.999	Price (without VAT) Amount of funds to be reserved.
vatclass	M	0-9	0-9999	Value added tax class Values described in Appendix B
servicegroupid	M	0-9	32 bit	Service group ID [1] Can be used for service specific denialing. Values described in Appendix C.
servicedescid	O	0-9	32 bit	Service description ID [1] Can be stored in CDR.
reservationtime	O	0-9	32 bit	Reservation time in seconds Default: 900 (15 minutes) If Commit message is not received before reservation time expires, reservation is cancelled.

[1] Allowed values depend on the service configured by the operator.

Below is an example of Reserve request using POST HTTP form data.

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-form-data
Content-Length: 144

username=user&password=pass&action=Reserve&transactionid=I2147549141&serv
iceid=31010&price=1.45&vatclass=1&servicegroupid=3&reservationtime=3600
```

2.1.4 Reserve response

Table 2.3 describes the parameters of response to Reserve funds request. Response statuscodes are described in table 2.4.

Table 2.3. Reserve response message.

Parameter	M / O	Accepted values	Max length	Description
status	M	ok, fail	4 chars	Status of the request
statuscode	M	0-9	32 bit	Status code
transactionid	M	A-z, 0-9	16 chars	Transaction ID

Below is an example of Reserve response.

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/http-form-data
Content-Length: 48
X-CAPI-Status: ok
X-CAPI-Status-Code: 0
X-CAPI-Transaction-Id: I2147549141

status=ok&statuscode=0&transactionid=I2147549141
```

Table 2.4. Response statuscodes.

Statuscode	Description
0	Reserve succeeded
1000	Authentication failed to username/password check
1001	Authentication failed to IP check
1002	Internal error
1100	Missing parameter 'username'
1101	Missing parameter 'password'
1102	Missing parameter 'action'
1103	Missing parameter 'msisdn' or 'imsi'
1104	Missing parameter 'price'
1105	Missing parameter 'serviceid'
1106	Missing parameter 'vatclass'
1500	Invalid parameter 'username'
1501	Invalid parameter 'password'
1502	Invalid parameter 'action'
1503	Invalid parameter 'msisdn'
1505	Invalid parameter 'serviceid'
1506	Invalid parameter 'reservationtime'
1507	Invalid parameter 'imsi'
1508	Invalid parameter 'servicegroupid'
1509	Invalid parameter 'servicedescid'
1510	Invalid parameter 'price'
1511	Invalid parameter 'vatclass'
1512	Invalid parameter 'transactionid'
1600	Invalid Content-Type
1601	Invalid message
2000	Subscriber check failed to connection error
2001	Subscriber check failed to unknown MSISDN
2002	Subscriber check failed to response timeout
2003	Subscriber check failed to barred MSISDN
2004	Subscriber check failed to unknown IMSI
3000	Prepaid charging failed to connection error
3001	Prepaid charging failed to insufficient balance
3002	Prepaid charging failed to response timeout
3003	Prepaid charging failed to Prepaid system error
5000	Insufficient balance

2.2   Commit payment

Committing payment is described in figure 2.8. Client commits the payment by opening an HTTP connection to CAPI and sending a Commit message. CAPI verifies the client by using username/password and IP authentication. CAPI also verifies that the provided transaction ID is still valid. If the message is accepted CAPI commits and closes the transaction. CAPI responds to the request with a response message that tells if the request was accepted or rejected. The HTTP statuscode of the response is always "200 OK", even if the request has been rejected. If the HTTP statuscode is not "200 OK", the client should retry the request.
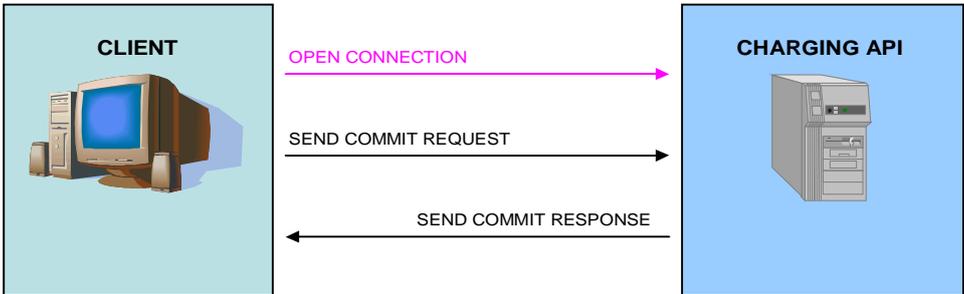
Figure 2.8. Commit payment

2.2.1   Message flows of successful commitations

Successful fund reservations can be charged or cancelled by sending a Commit message. After successful commitation the transaction is closed.

A successful charging of postpaid payment is described in figure 2.9.
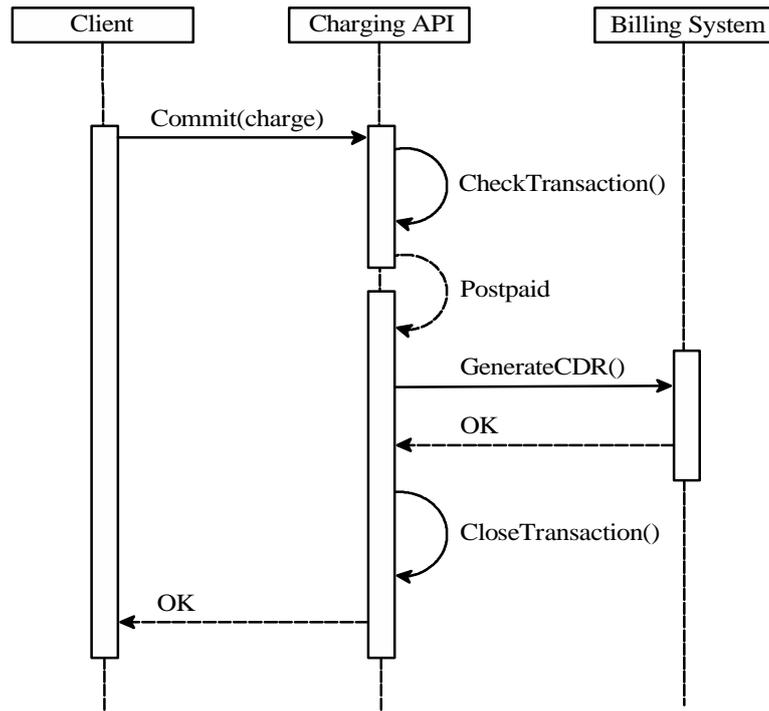
Figure 2.9. Successful charging of postpaid payment

A successful charging of prepaid payment is described in figure 2.10.
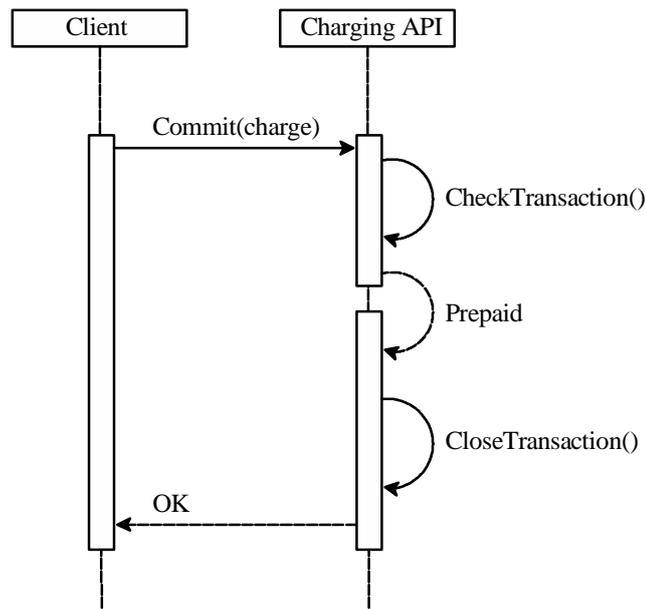
Figure 2.10. Successful charging of prepaid payment

A successful cancelling of postpaid payment is described in figure 2.11.
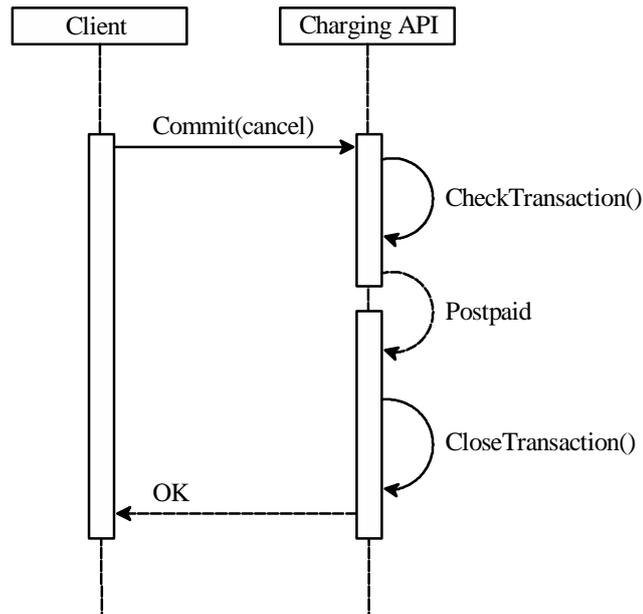
Figure 2.11. Successful cancelling of postpaid payment

A successful cancelling of prepaid payment is described in figure 2.12.
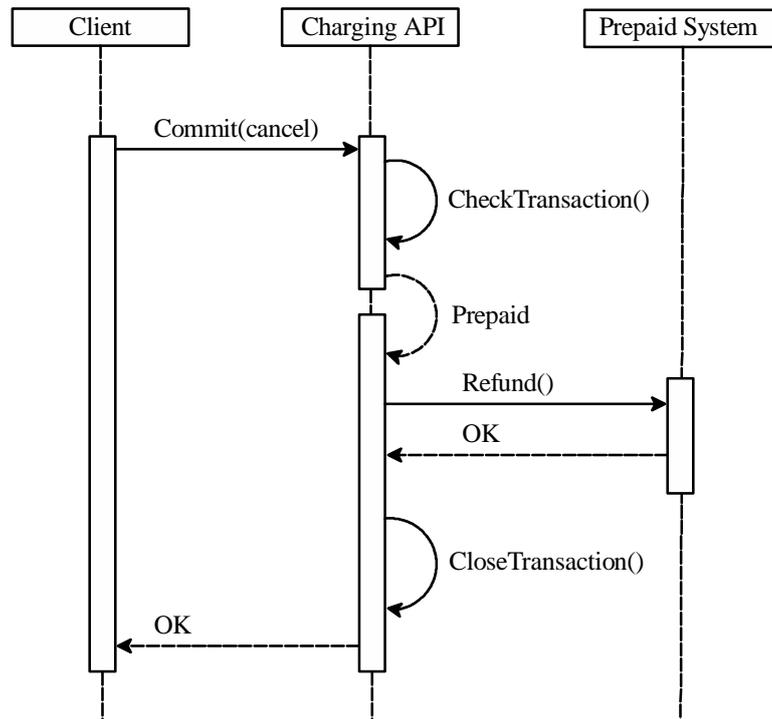
Figure 2.12. Successful cancelling of prepaid payment

2.2.2 Message flows of failed commitations

If commitation fails due to Billing System of Prepaid System error, CAPI leaves the transaction open. If successful commitation is not made before the reservation time of the transaction, CAPI cancels and closes the transaction.

Figure 2.13 describes a case when CAPI cannot find a valid transaction ID.
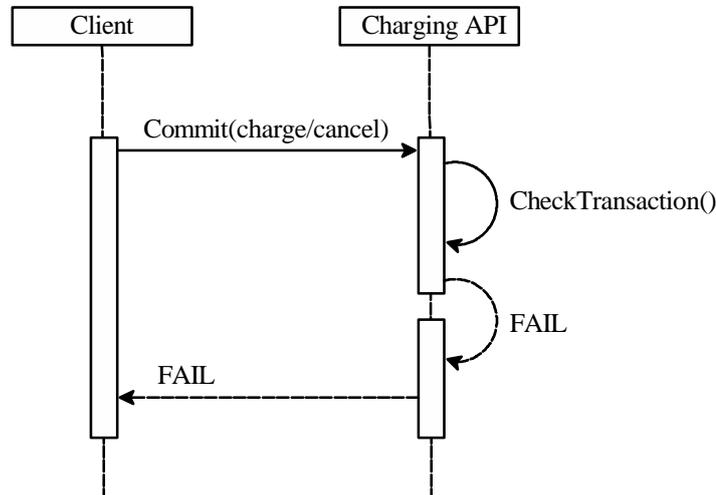
Figure 2.13. Rejected committing of payment due to invalid transaction ID

Figure 2.14 describes a case when CAPI cannot generate a CDR for postpaid charging of payment.
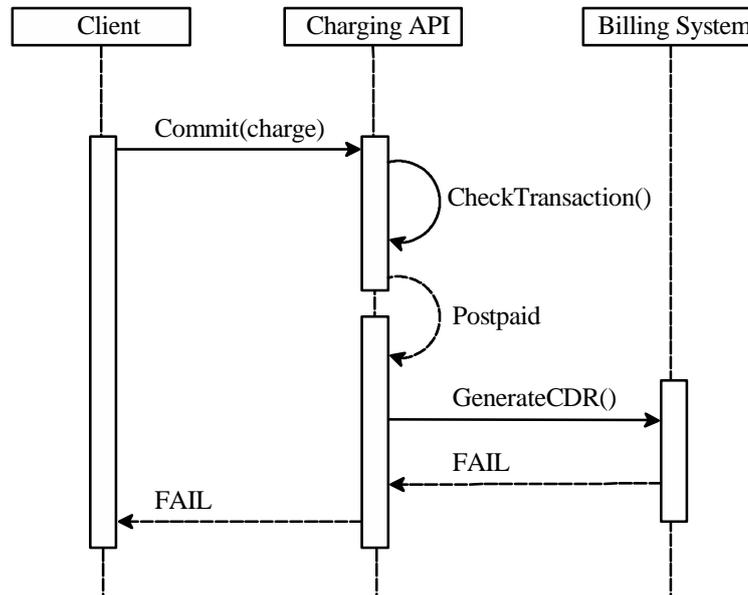
Figure 2.14. Rejected charging of payment due to CDR generation failure

Figure 2.15 describes a case when CAPI cannot refund for prepaid payment cancellation.
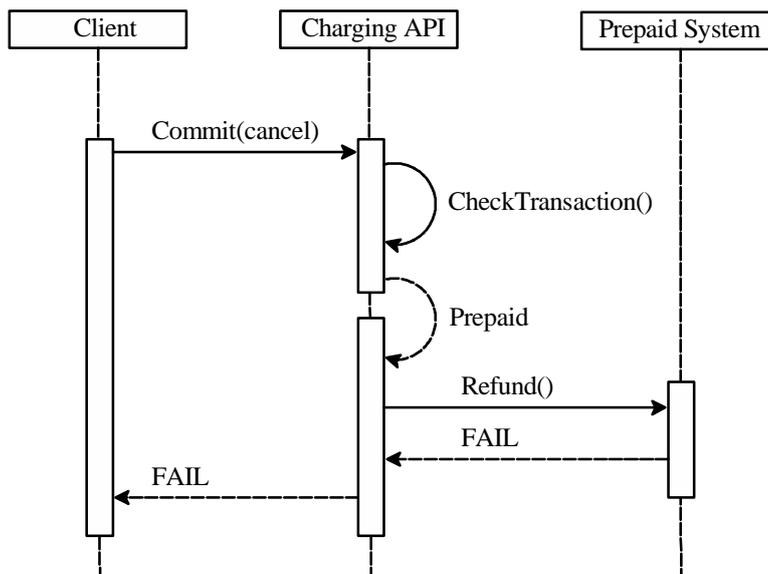
Figure 2.15. Rejected cancelling of payment due to refund failure

2.2.3 Commit request

Table 2.5 describes the parameters client can supply when committing payment. M/O column describes whether the parameter is mandatory or optional.

Table 2.5. Commit request message.

Parameter	M / O	Accepted values	Max length	Description
username	M	A-z, 0-9	32 chars	Username
password	M	A-z, 0-9	32 chars	Password
action	M	Commit	6 chars	Type of the message
transactionid	M	A-z, 0-9	16 chars	Transaction ID Same ID as in the Reserve response message.
method	M	charge, cancel	6 chars	Commit method charge = charge funds cancel = cancel reservation

Below is an example of Commit request using POST HTTP form data.

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-form-data
Content-Length: 81

username=user&password=pass&action=Commit&transactionid=I2147549141&metho
d=charge
```

2.2.4 Commit response

Table 2.6 describes the parameters of response to Commit request. Response statuscodes are described in table 2.7.

Table 2.6. Commit response message.

Parameter	M / O	Accepted values	Max length	Description
status	M	ok, fail	4 chars	Status of the request
statuscode	M	0-9	32 bit	Status code
transactionid	M	A-z, 0-9	16 chars	Transaction ID

Below is an example of Commit response.

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/http-form-data
Content-Length: 48
X-CAPI-Status: ok
X-CAPI-Status-Code: 0
X-CAPI-Transaction-Id: I2147549141

status=ok&statuscode=0&transactionid=I2147549141
```

Table 2.7. Response statuscodes.

Statuscode	Description
0	Commit succeeded
1000	Authentication failed to username/password check
1001	Authentication failed to IP check
1002	Internal error
1100	Missing parameter 'username'
1101	Missing parameter 'password'
1102	Missing parameter 'action'
1103	Missing parameter 'transactionid'
1104	Missing parameter 'method'
1500	Invalid parameter 'username'
1501	Invalid parameter 'password'
1502	Invalid parameter 'action'
1503	Invalid parameter 'transactionid'
1504	Invalid parameter 'method'
1505	Invalid parameter 'validityperiod'
1506	Invalid parameter 'price'
1507	Invalid parameter 'vatclass'
1600	Invalid Content-Type
1601	Invalid message
2000	Unknown transaction ID
2001	Expired transaction ID
3000	Prepaid refund failed to connection error
3001	Prepaid refund failed to response timeout
3002	Prepaid refund failed to Prepaid system error
3003	Prepaid charging failed to insufficient balance
5000	Insufficient balance

2.3 Direct Debit

Direct debit is described in figure 2.16. Client opens an HTTP connection to CAPI and sends a Direct Debit message. CAPI verifies the client by using username/password and IP authentication. If the message is accepted, CAPI executes a direct debit on the MSISDN. CAPI responds to the request with a response message that tells if the request was accepted or rejected. The HTTP statuscode of the response is always "200 OK", even if the request has been rejected. If the HTTP statuscode is not "200 OK", the client should retry the request.
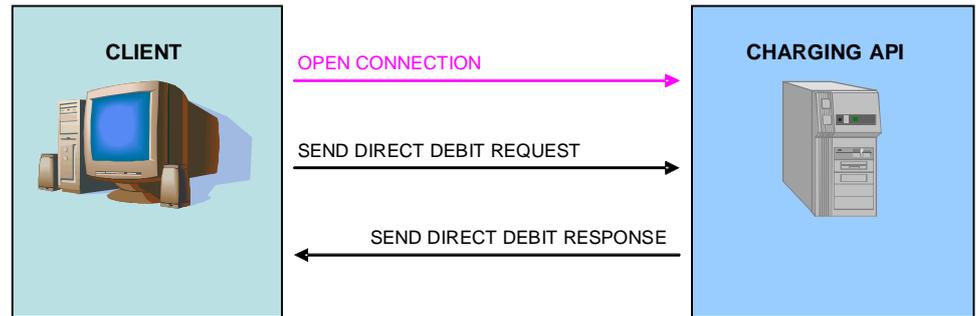
Figure 2.16. Direct Debit

2.3.1 Message flows of successful direct debits

This chapter describes successful direct debit transactions for both postpaid and prepaid MSISDNs.

A successful direct debit of postpaid MSISDN is described in figure 2.17.
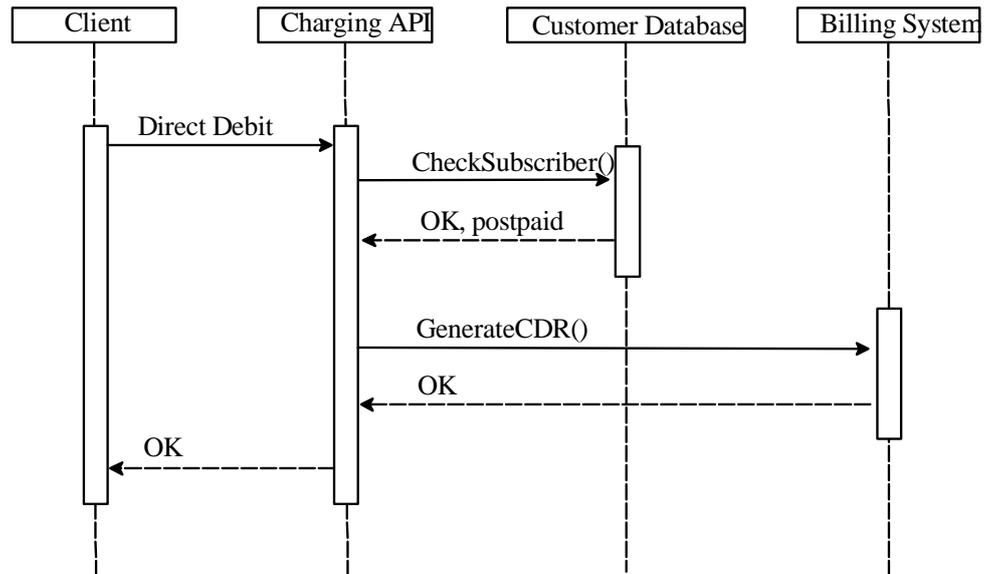
Figure 2.17. Successful Direct Debit of postpaid MSISDN

A successful direct debit of prepaid MSISDN is described in figure 2.18.
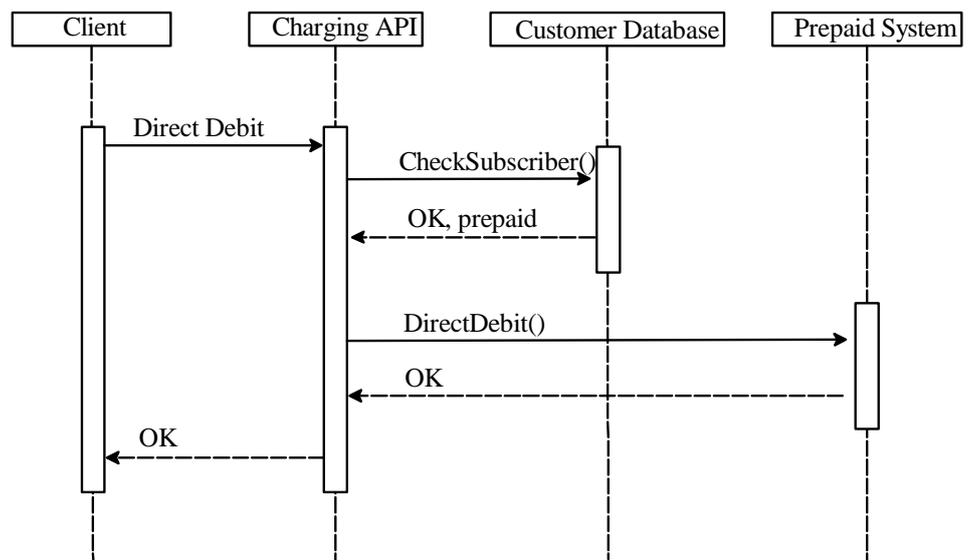
Figure 2.18. Successful Direct Debit of prepaid MSISDN

2.3.2 Message flows of failed direct debits

This chapter describes failed direct debit transactions.

Figure 2.19 describes a case when CAPI cannot obtain MSISDN information from the Customer Database.
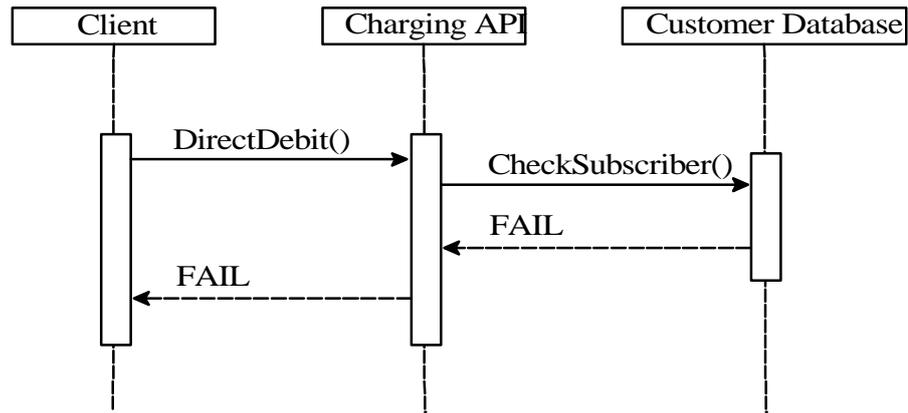
Figure 2.19. Rejected Direct Debit due to subscriber check failure

Figure 2.20 describes a case when CAPI cannot generate a CDR for postpaid charging of payment.
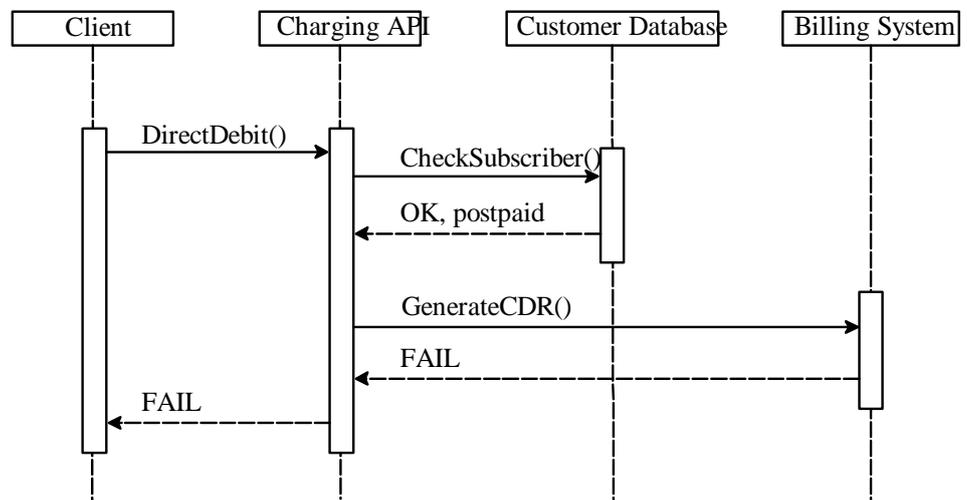
Figure 2.20. Rejected Direct Debit due to CDR generation failure

Figure 2.21 describes a case when CAPI cannot do a direct debit for prepaid charging of payment.
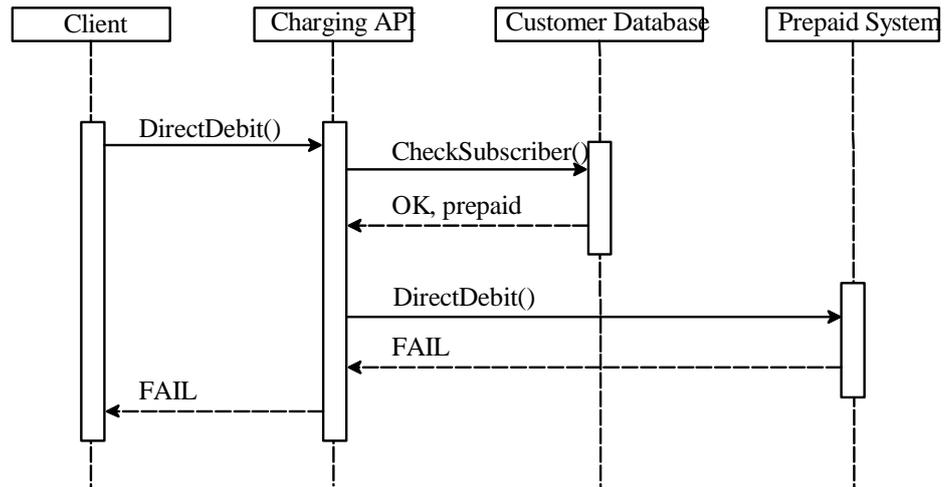
Figure 2.21. Rejected Direct Debit due to prepaid system failure

2.2.5 Direct Debit request

Table 2.8 describes the parameters client can supply when executing a direct debit.
M/O column describes whether the parameter is mandatory or optional.

Table 2.8. Direct Debit request message.

Parameter	M / O	Accepted values	Max length	Description
username	M	A-z, 0-9	32 chars	Username
password	M	A-z, 0-9	32 chars	Password
action	M	DirectDebit	11 chars	Type of the message
transactionid	M	A-z, 0-9	16 chars	Transaction ID. Received from IP billing message.
serviceid	M	0-9	32 bit	Service ID
price	M	0-9, .	0-999.999	Price (without VAT). Amount of funds to be debited.
vatclass	M	0-9	0-9999	Value added tax class. Values described in Appendix B.
servicegroupid	M	0-9	32 bit	Service group ID [1] Can be used for service specific denial. Values described in Appendix C.
servicedescid	O	0-9	32 bit	Service description ID [1] Can be stored in CDR.

[1] Allowed values depend on the service configured by the operator.

Below is an example of Direct Debit request using POST HTTP form data.

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-form-data
Content-Length: 127

username=user&password=pass&action=DirectDebit&transactionid=I2147549141&
serviceid=31010&price=1.45&vatclass=1&servicegroupid=3
```

2.2.6 Direct Debit response

Table 2.9 describes the parameters of response to Direct Debit request. Response statuscodes are described in table 2.10.

Table 2.9. Direct Debit response message.

Parameter	M / O	Accepted values	Max length	Description
status	M	ok, fail	4 chars	Status of the request
statuscode	M	0-9	32 bit	Status code
transactionid	M	A-z, 0-9	16 chars	Transaction ID

Below is an example of Direct Debit response.

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/http-form-data
Content-Length: 48
X-CAPI-Status: ok
X-CAPI-Status-Code: 0
X-CAPI-Transaction-Id: I2147549141

status=ok&statuscode=0&transactionid=I2147549141
```

Table 2.10. Response statuscodes.

Statuscode	Description
0	Direct Debit succeeded
1000	Authentication failed to username/password check
1001	Authentication failed to IP check
1002	Internal error
1003	Subscriber barred
1100	Missing parameter 'username'
1101	Missing parameter 'password'
1102	Missing parameter 'action'
1103	Missing parameter 'transactionid'
1104	Missing parameter 'msisdn' or 'imsi'
1105	Missing parameter 'price' or 'vatclass'
1106	Missing parameter 'serviceid'
1107	Missing parameter 'vatclass'
1500	Invalid parameter 'username'
1501	Invalid parameter 'password'
1502	Invalid parameter 'action'
1503	Invalid parameter 'transactionid'
1504	Invalid parameter 'msisdn'
1506	Invalid parameter 'serviceid'
1508	Invalid parameter 'servicegroupid'
1509	Invalid parameter 'servicedescid'
1510	Invalid parameter 'price'
1511	Invalid parameter 'vatclass'
1512	Invalid parameter 'vatclass'
1600	Invalid Content-Type

1601	Invalid message
2000	Unknown transaction ID
2001	Expired transaction ID
3000	Subscriber check failed to connection error
3001	Subscriber check failed to unknown MSISDN
3002	Subscriber check failed to response timeout
3003	Subscriber check failed to barred MSISDN
3004	Subscriber check failed to unknown IMSI
4000	Prepaid charging failed to connection error
4001	Prepaid charging failed to insufficient balance
4002	Prepaid charging failed to response timeout
4003	Prepaid charging failed to Prepaid system error
5000	Insufficient balance

3. IP Billing Proxy

This chapter describes the IP Billing Proxy (IBP). IPB provides a charging interface between the MSISDN service operator and content providers. Service operator's GGSN servers and IPB communicate through HTTPS connection. IPB supports GET and POST methods for sending messages. Supported message format for GET method is URL query string or HTTP headers. POST method supports HTTP headers and HTTP form data. Values of message parameters must be URL encoded. See appendix A for URL encoding and chapter 4 for examples.

IPB supports only secured HTTP connections with SSL. SSL connections do not use client certificates.

Table 3.1 describes accepted HTTPS connection parameters.

Table 3.1. Accepted HTTPS connection parameters.

Parameter	Accepted values
Protocol	HTTP/1.1 or HTTP/1.0
Method	GET or POST
URL	https://vascenter.tieto.com/ipb/capi/
TCP Port (secured HTTP)	8443

IPB overview is described in figure 3.1. IP billing is initiated by an MSISDN subscriber when it is requesting a downloadable content from the GGSN. The GGSN redirects the request to IPB which validates the request and forwards it to the content provider. The content provider charges the MSISDN by sending a Direct Debit request to CAPI using the charging parameters received from IPB. The content provider returns the requested content or an error content to IPB which returns the response as-is to the subscriber.
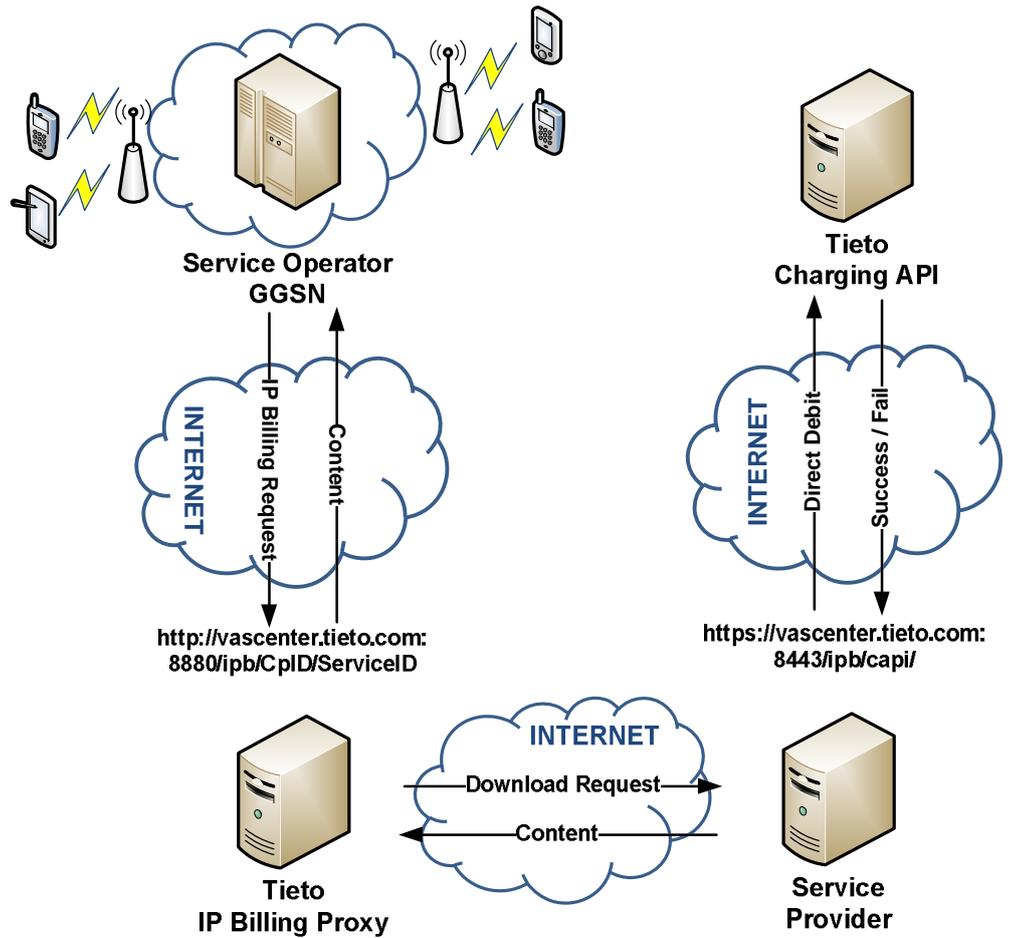
Figure 3.1. IP Billing Proxy overview

Figure 3.2 describes an IP billing transaction. Each transaction consists of a single request/response pair.
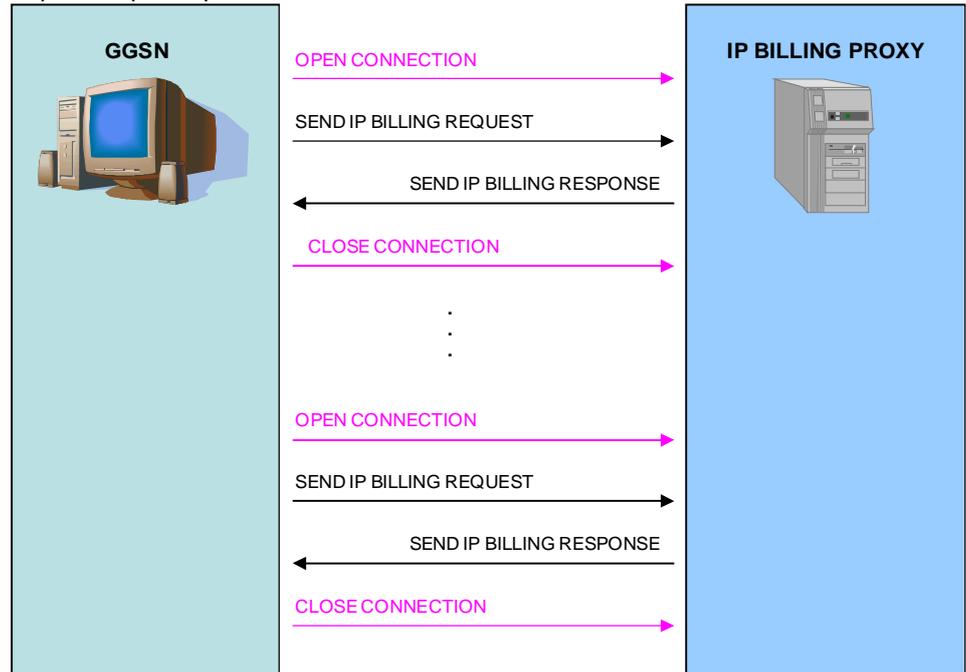
Figure 3.2. IP billing transaction using non-persistent connection

IPB supports both persistent and non-persistent connections. Using persistent connection the GGSN can send several IP billing requests using the same connection. Using non-persistent connection the GGSN can close the connection after each request/response pair.

3.1.1 Message flows of successful IP billing

A successful IP Billing request is described in figure 3.3.
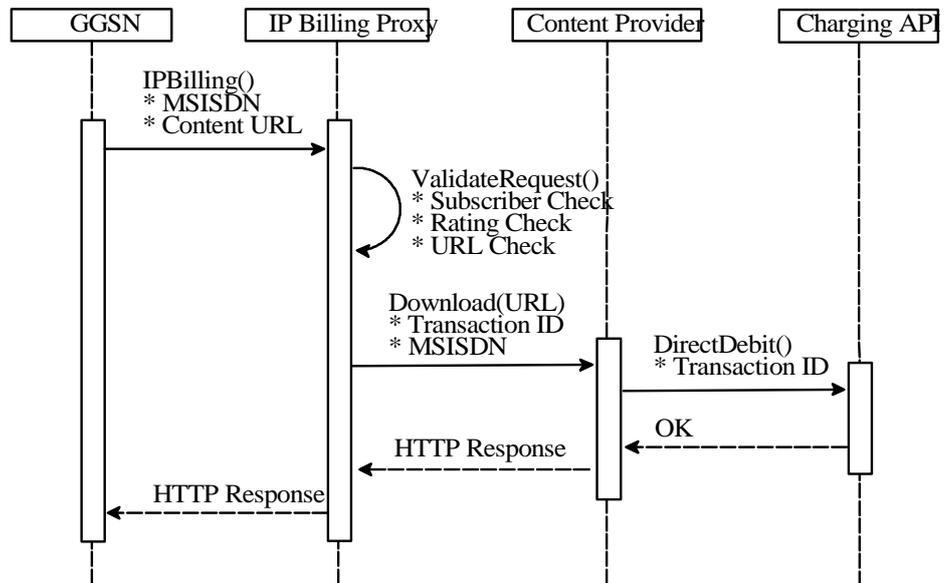
Figure 3.3. Successful IP Billing request

The GGSN sends an IP Billing request to IPB with MSISDN and optional content URL parameter. IPB validates the request by performing a subscriber check to the Customer Database, executing a rating check on the subscriber and by identifying the content provider. If the subscriber is found and the content provider identified, IPB generates an unique transaction ID for the request and forwards the request to the content provider along with the transaction ID and MSISDN parameters. The request also contains the status of rating and URL checks. The content provider sends a direct debit request to CAPI with the transaction ID received from IPB. After the direct debit, the content provider returns the requested content (or an error content) back to IPB which forwards the response as-is back to the GGSN.

3.1.2 Message flows of failed IP billing

A failed IP Billing request due to unknown subscriber is described in figure 3.4.
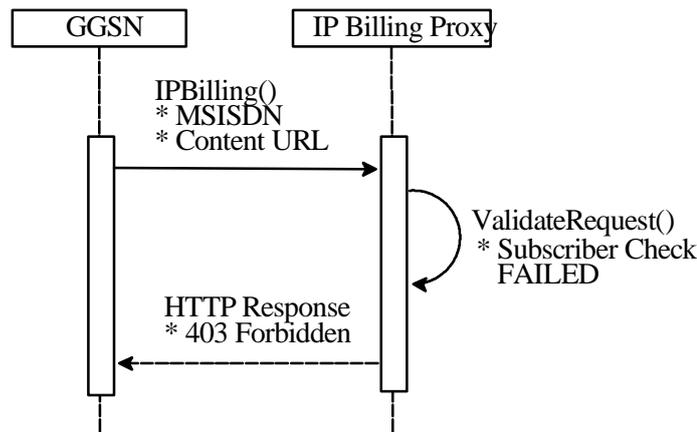
Figure 3.4. Rejected IP Billing request due to unknown subscriber

A failed IP Billing request due to subscriber charging failure is described in figure 3.5. The error content generated by the content provider is returned to the GGSN.
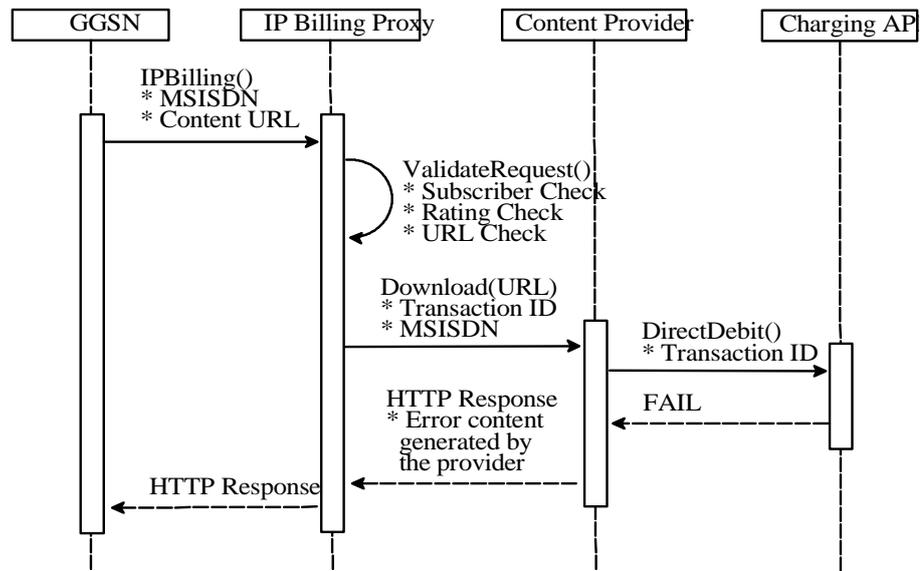
Figure 3.5. Rejected IP Billing request due to failed subscriber charging

3.1.3 IP Billing request from MSISDN

All HTTP request headers except the X-IPB-Subscriber and X-IPB-URL are passed through as-is to the content provider. Also X-IPB-Transaction-Id, X-IPB-Msisdn, X-IPB-Rating and X-IPB-Barring headers generated by IPB are sent to the content provider. X-IPB-Rating describes the status code of the subscriber rating check and X-IPB-Barring the status code of the URL barring check.

Table 3.4. X-IPB-Rating status codes.

Status code	Description
0	Rating check OK
1	Subscriber is out of balance
2	Rating check failed

Table 3.5. X-IPB-Barring status codes.

Status code	Description
0	OK, not barred
1	URL is barred

If the request does not contain an X-IPB-URL parameter, the default content URL from the content provider configuration is used. See chapter 4 for examples.

3.1.4 IP Billing response

The HTTP response generated by the content provider is returned as-is in the IP Billing response. It is the content provider's responsibility to generate an HTTP error response the subscriber's device understands in case the requested content is unavailable for some reason.

The only exception when IPB generates the response is when the request is rejected due to unknown subscriber or content provider. In this case an HTTP error response of "403 Forbidden" is returned.

4. Examples

This chapter describes a couple of charging and IP billing examples. In the first example a client reserves funds from a MSISDN. In the second example the client commits the payment. In the third example the client executes a direct debit. The last example describes IP billing transactions. All examples describe successful and failed responses.

4.1 Reserving funds

Client reserves funds by sending a following message described in figure 4.1.

CLIENT   **CHARGING API**

SEND RESERVE REQUEST

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-form-data
Content-Length: 144

username=user&password=pass&action=Reserve&transactionid=I2147549141&servi
ceid=31010&price=1.45&vatclass=1&servicegroupid=3&reservationtime=3600
```

OR

```
GET /ipb/capi?username=user&password=pass&action=Reserve&transactionid=
I2147549141&serviceid=31010&price=1.45&vatclass=1&servicegroupid=3&reservati
ontime=3600 HTTP/1.1
Host: 127.0.0.1:25000
```

OR

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-headers
Content-Length: 0
X-CAPI-Username: user
X-CAPI-Password: pass
X-CAPI-Action: Reserve
X-CAPI-Transaction-Id: I2147549141
X-CAPI-Service-Id: 31010
X-CAPI-Price: 1.45
X-CAPI-Vat-Class: 1
X-CAPI-Service-Group-Id: 3
X-CAPI-Reservation-Time: 3600
```

Figure 4.1. Reserve funds request

CAPI responds to the request by sending a following message described in figure 4.2. If the request has failed the status code describes the error.

CLIENT   **CHARGING API**

```
                                                              SEND RESERVE RESPONSE
◄─────────────────────────────────────────────────────────────────────────

  HTTP/1.1 200 OK
  Date: Wed, 19 Feb 2014 10:11:12 GMT
  Content-Type: application/http-form-data
  Content-Length: 48
  X-CAPI-Status: ok
  X-CAPI-Status-Code: 0
  X-CAPI-Transaction-Id: I2147549141

  status=ok&statuscode=0&transactionid=I2147549141

  OR

  HTTP/1.1 200 OK
  Date: Wed, 19 Feb 2014 10:11:12 GMT
  Content-Type: application/http-form-data
  Content-Length: 53
  X-CAPI-Status: fail
  X-CAPI-Status-Code: 2001
  X-CAPI-Transaction-Id: I2147549141

  status=fail&statuscode=2001&transactionid=I2147549141
```

Figure 4.2. Reserve funds response

4.2   Committing payment

Client charges the payment by sending a following message described in figure 4.3.

CLIENT   **CHARGING API**

```
  SEND COMMIT REQUEST
───────────────────────────────────────────────────────────────────────►

  POST /ipb/capi HTTP/1.1
  Host: 127.0.0.1:25000
  Content-Type: application/http-form-data
  Content-Length: 81

  username=user&password=pass&action=Commit&transactionid=I2147549141&meth
  od=charge

  OR

  GET /ipb/capi?username=user&password=pass&action=Commit&transactionid=
  I2147549141&method=charge HTTP/1.1
  Host: 127.0.0.1:25000

  OR

  POST /ipb/capi HTTP/1.1
  Host: 127.0.0.1:25000
  Content-Type: application/http-headers
  Content-Length: 0
  X-CAPI-Username: user
  X-CAPI-Password: pass
  X-CAPI-Action: Commit
  X-CAPI-Transaction-Id: I2147549141
  X-CAPI-Method: charge
```

Figure 4.3. Charge payment request

CAPI responds to the request by sending a following message described in figure 4.4. If the request has failed the status code describes the error.
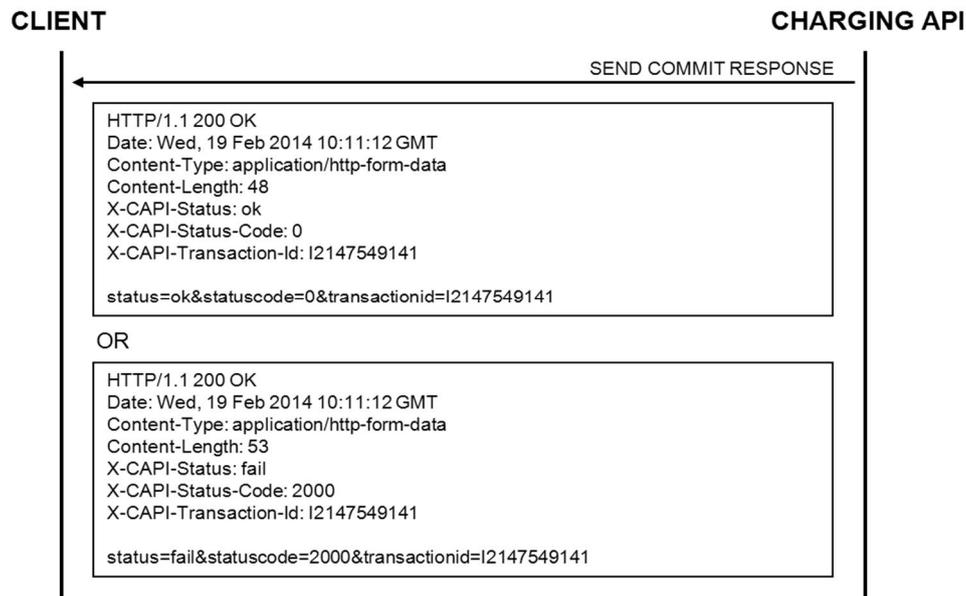
Figure 4.4. Charge payment response

4.3 Direct Debiting

Client executes a direct debit by sending a following message described in figure 4.5.
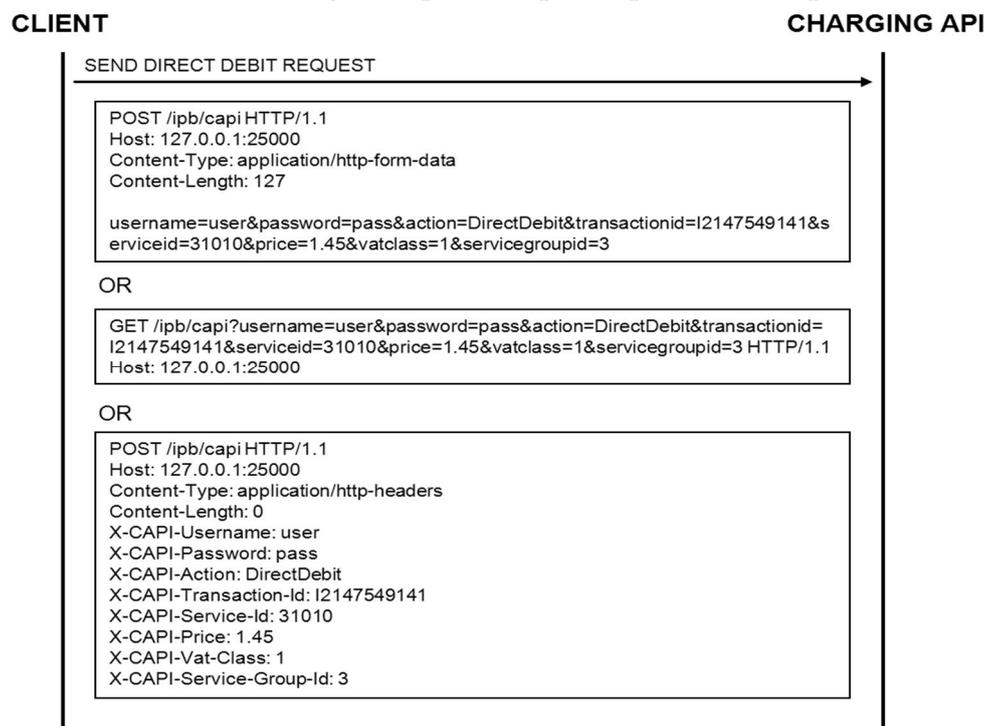

Figure 4.5. Direct Debit request

CAPI responds to the request by sending a following message described in figure 4.6. If the request has failed the statuscode describes the error.

CLIENT   **CHARGING API**

SEND DIRECT DEBIT RESPONSE

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/http-form-data
Content-Length: 48
X-CAPI-Status: ok
X-CAPI-Status-Code: 0
X-CAPI-Transaction-Id: I2147549141

status=ok&statuscode=0&transactionid=I2147549141
```

OR

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/http-form-data
Content-Length: 53
X-CAPI-Status: fail
X-CAPI-Status-Code: 2001
X-CAPI-Transaction-Id: I2147549141

status=fail&statuscode=2001&transactionid=I2147549141
```

Figure 4.6. Direct Debit response

4.4   IP Billing

When IPB receives request from MSISDN it identifies both content provider and service and forwards the request as described in figure 4.8.

IP BILLING PROXY   **CONTENT PROVIDER**

SEND REQUEST

```
POST /game.app HTTP/1.1
Host: app.download.com
Content-Type: application/http-form-data
Content-Length: 0
X-IPB-MSISDN: 358441234567
X-IPB-Transaction-Id: I2147549141
X-IPB-Rating: 0
X-IPB-Barring: 0
```

OR

```
GET /game.app HTTP/1.1
Host: app.download.com
X-IPB-MSISDN: 358441234567
X-IPB-Transaction-Id: I2147549141
X-IPB-Rating: 0
X-IPB-Barring: 0
```

Figure 4.8. IP Billing request forwarded to content provider

Figure 4.11 describes the IP Billing response. If the request was successful, IP Billing response contains the content received from the content provider. If the request was rejected by IPB, a "403 Forbidden" error response is returned.

GGSN   **IP BILLING PROXY**

SEND IP BILLING RESPONSE

```
HTTP/1.1 200 OK
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: application/octet-stream
Content-Length: 65432

<binary content>
```

OR

```
HTTP/1.1 403 Forbidden
Date: Wed, 19 Feb 2014 10:11:12 GMT
Content-Type: text/html
Content-Length: 35

<html><body>Forbidden</body></html>
```
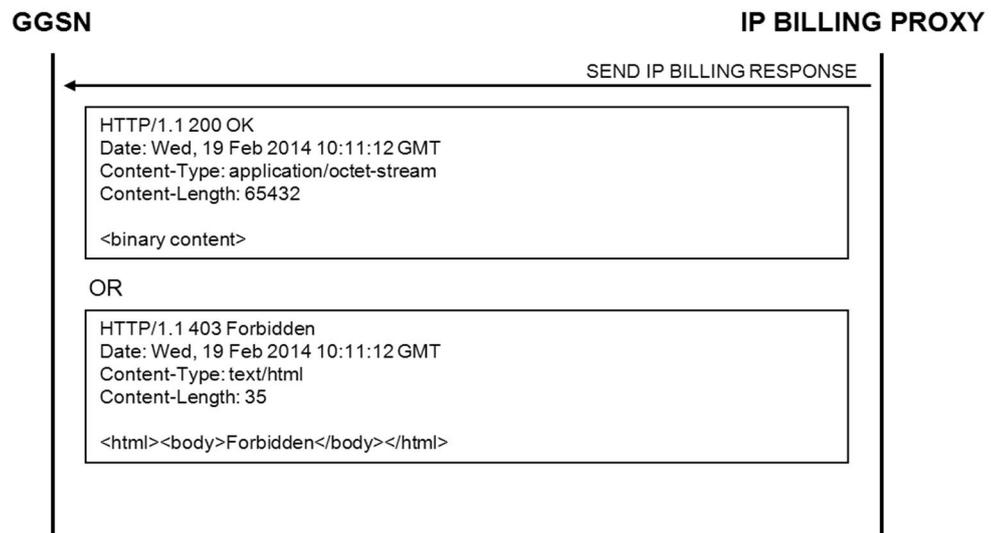
Figure 4.11. IP Billing response

Appendix A. URL encoding

Values of message parameters must be URL encoded as described in RFC 1738. URL encoding of a character consists of a '%' symbol followed by two-digit hexadecimal value of the character in ISO-Latin set.

For example the following message content

```
servicedesc=Alpha & Beta&cpdesc=Test provider&tc=2
```

is URL encoded as:

```
servicedesc=Alpha%20%26%20Beta&cpdesc=Test%20provider&tc=2
```

Appendix B. VAT class values

Class 0	0%
Class 1	24%
Class 2	14%
Class 3	10%

Appendix C. Service groups

Content provider is responsible for categorizing all premium rate IP content according to the service categories as defined by Finnish Communications Regulatory Authority [3]. Service groups are as follows:

Service group 1	General services
Service group 2	Consulting and ordering services
Service group 3	Entertainment services
Service group 4	Adult services

Example of Direct Debit request with service group "*Consulting and ordering services*".

```
POST /ipb/capi HTTP/1.1
Host: 127.0.0.1:25000
Content-Type: application/http-form-data
Content-Length: 113

username=user&password=pass&action=DirectDebit&I2147549141&serviceid=
31010&price=1.45&vatclass=1&servicegroupid=2
```

Content provider places services physically to different URLs based on which category the service falls. The URL of the service must contain the appropriate service group identifier. The identifiers are as follows:

Service group 1   sg1
Service group 2   sg2
Service group 3   sg3
Service group 4   sg4

More detailed information of Regulation 35 P/2011 on barring categories in telecommunications. www.viestintavirasto.fi

Example of IP billing request (from MSISDN) URL with service group "*Consulting and ordering services*".

```
http://vascenter.tieto.com:8880/ipb/987/31010/sg2?par=billme
```

Appendix D. IPB request HTTP headers

x-ipb-url

Appendix E. IPB request forward HTTP headers

x-ipb-transaction-id
x-ipb-msisdn
x-ipb-barring
x-ipb-rating

Appendix F. CAPI request/response HTTP headers

x-capi-username
x-capi-password
x-capi-action
x-capi-transaction-id
x-capi-price
x-capi-vat-class
x-capi-service-id
x-capi-service-desc-id
x-capi-service-group-id
x-capi-reservation-time
x-capi-method

x-capi-status   only in response
x-capi-status-code   only in response