Mobile Payment Widget

Version 3.1.2

05.05.2017

Table of contents

1 Historie

Date	Changes	Version	Author
15.12.2009	initial version	1.0	Sten Uhlig
05.01.2010	supplementing 2-Phase Payment Changes URLs and Ressources	1.1	Sten Uhlig
27.01.2010	Change length of CallbackURL Correction naming „hash“	1.2	Sten Uhlig
23.02.2010	new Reponsecode 113 new Reponsecode 121 new chapter „specific features in WAP/mobile WEB“ new validation of description	1.3	Sten Uhlig
10.04.2010	added Subscription cleaned Responsecodes	2.0	Sten Uhlig
01.09.2010	correction of parameter lists enhanced Followup Payment Parameter lists	2.1	Sten Uhlig
15.09.2010	added synchron Webservice Requests added One Phase Subscription Setup	2.2	Sten Uhlig
03.02..2011	Correct parameter order in Header Redirect Subscription	2.2.1	Sten Uhlig
23.08.2011	new ReturnCodes	2.2.2	Sten Uhlig
20.12.2011	Double Subscriptions prevention, Bugfixes	2.3	Sten Uhlig
20.08.2012	Correction ResponseCodes	2.3.1	Sten Uhlig
05.11.2012	Correction Refund and Refund Callback	2.3.2	Sten Uhlig
12.11.2012	new Webservice method „status“ new Responseparameter „or“	2.4	Sten Uhlig
15.01.2013	New parameter transaction status method	2.4.1	Sten Uhlig

26.02.2013	new payment systems	2.5	Sten Uhlig
04.07.2013	New parameter transactionid in WS requests	2.6	Sten Uhlig
01.08.2013	new Contentclass 22 Erotik	2.7	Sten Uhlig
27.08.2013	New method subscriptionstatus Added list of subscriptionstatuses	2.8	Sten Uhlig
03.12.2014	Removed TANSMSText and TANID Updated Operators	2.8.1	Sten Uhlig
06.07.2015	Updated transactionstatus, subscriptionstatus	2.8.2	Sten Uhlig
11.09.2015	Added currency parameter	3.0.0	Sten Uhlig
18.01.2017	Removed ansynch mode (Callbacks)	3.1.0	Sten Uhlig
25.01.2017	Edited subscriberid RegExps in redirect callbacks (I_Token)	3.1.1	Sten Uhlig
05.05.2017	Added existing responsecodes 11, 12	3.1.2	Sten Uhlig

2 Overview

mbe4 offers an aggregating platform for various payment processes and systems

- Carrier payment
- Click and Buy
- Giropay
- Credit card
- PayPal
- Direct debit
- mpass
- tbc

Mbe4 is easy to integrate in existing environments and can be easily adapted to mobile and/or big screen scenarios.

2.1 Glossary

- mbe4 = mbe Billing Plattform.
- Subscriber = mobile phone user,
- Client = service provider, shop operator
- APN Access Point Name is the name of a gateway between a GPRS, 3G or 4G mobile network
- Single Payment – single payment transaction (no subscription)
- Subscription/Abo – payment model which consists of an initial payment with user interaction and follow up payments without user interaction

2.2 Order types

Supported are:

- WEB
- WAP (= mobile Web)

2.3 Payment processes 1-Phase Single Payment vs 2-Phase Single Payment vs Subscription)

- 1-Phase Single Payment
  - Transaction consists of one single step
  - Authorization and capture are combined in one single step.
  - That is why the service/ content delivery takes place after capturing
- 2-Phase Single Payment
  - Authorization and capture are two different steps
  - Service delivery takes place after successful authorization and before capturing

- Subscription
    - Subscription consists of multiple payment transactions
    - There are 3 subscription steps
        - Subscription Setup (set up a subscription and perform a first payment)
        - Follow up Payments (cyclic follow up payments without user interaction)
        - Subscription Termination (terminate a subscription – no follow ups allowed after termination)
    - Subscription Setup can be executed as one or two phase transaction
    - Follow up Payments can be executed as one or two phase transaction

2.4 Integrating mbe4

Mbe4 payment widget can be integrated as

- Popup
- IFrame
- Header redirect

Mbe4 takes care of the whole user interaction.

All parameters are submitted as http-GET parameters. All values need to be URL-encoded.

NOTE: some payment systems (like Paypal) do not allow iFrame integration.

3 URLs and resources

URL: http(s)://billing.mbe4.de

For testing purposes, services can be configured as „auto refund“. That means, every successful carrier billing transaction will be refunded automatically.

4 character encoding

- all parameters need to be: **UTF-8**
- HTTP GET Parameter need to be URL-encoded

5 Specifics in WAP/mobile WEB

Mbe4 is able to recognize subscriber mobile numbers (MSISDN) automatically when a subscriber uses the carriers´ mobile networks. (only works with carrier payment for German carrier customers)

That means transactions can be processed without a SMS-TAN handshake. “one click payment”

Because mobile number recognition is technically based on a http header enrichment in the carriers´ APNs, it only works with plain http connections. That means the client must not redirect the subscriber to https://billing... if mobile number recognition should be used.

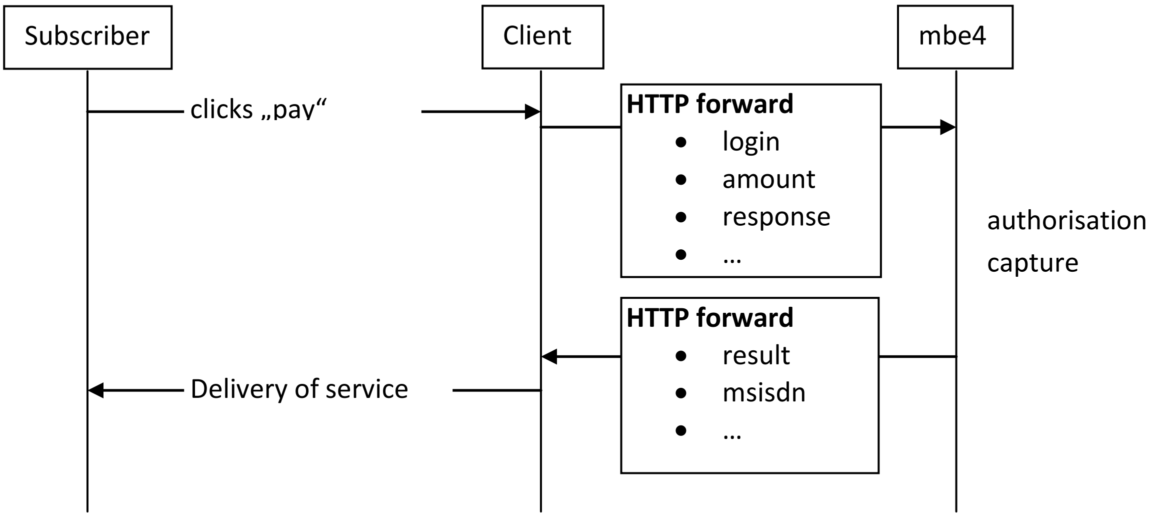
6 Process – mbe4 Mobile Payment Widget

The following charts visualize the different mbe4 processes.

(precondition: Subscriber has chosen an item and chooses mbe4 Mobile Payment Widget to pay)

Note: A mbe4 service is populated for one payment process/payment system at a time. It is not possible to use one service for different payment systems or different payment processes.

6.1 1-Phase Single Payment



1. Client redirects subscriber(-Browsers) to mbe4 Widget URL (Client adds all necessary parameters to the redirect URL).
2. mbe4 executed the whole payment process including authorization and capturing
3. mbe4 redirects the subscriber to a given redirect URL. The redirect URL contains all necessary parameters a client needs.

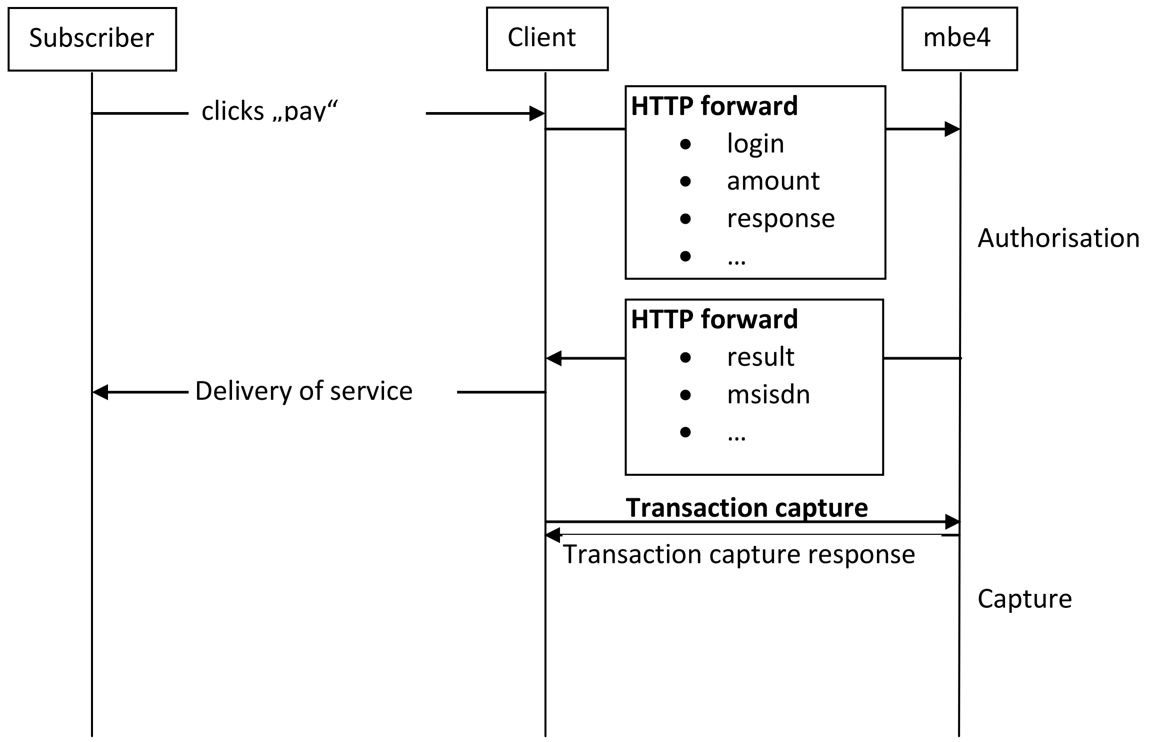
Positive result

4. Client will perform the delivery of service

Negative result

4. Client informs the subscriber that the transaction failed (and offers different payment methods) NOTE: mbe4 usually does not communicate the result of a payment transaction to the subscriber!

6.2 2-Phase Single Payment



1. Client redirects subscriber (-Browsers) to mbe4 Widget URL (Client adds all necessary parameters to the redirect URL).
2. mbe4 executed the whole payment process including authorization (but no capturing will take place).
3. mbe4 redirects the subscriber to a given redirect URL. The redirect URL contains all necessary parameters a client needs.

Positive result

4. Client will perform the delivery of service
5. Client sends a „Transaction capture“ request to mbe4 with a capture result.

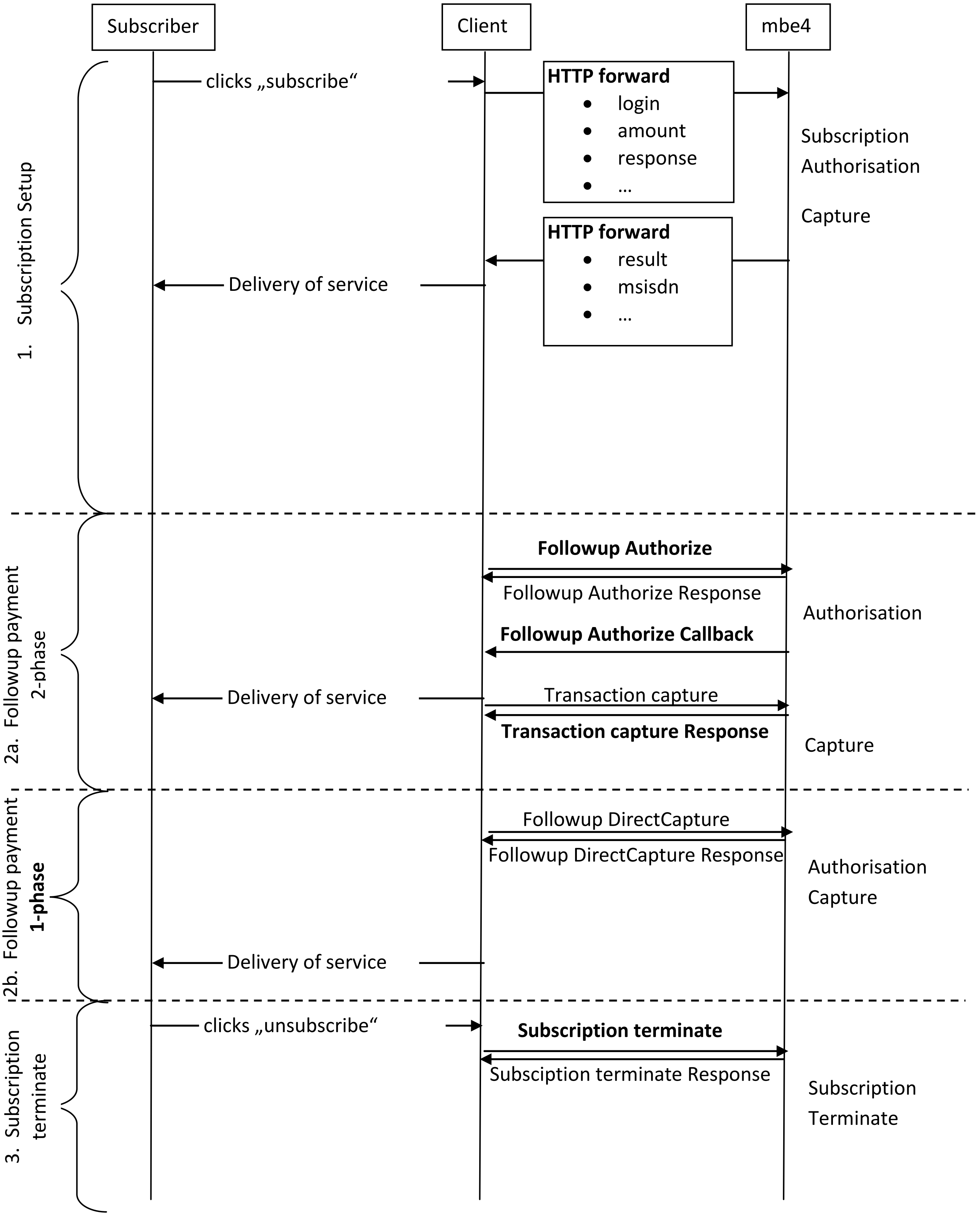
Negative result

5. Client informs the subscriber that the transaction failed (and offers different payment methods) NOTE: mbe4 usually does not communicate the result of a payment transaction to the subscriber!

NOTE2:

Some carriers do not fully support „2Phase Payments”. Therefore some 2 Phase transactions can be already captured when the subscriber is redirected to the client. Nevertheless the capture request can be executed. The answer will be positive!

6.3 1 –Phase Subscription Setup



A Subscription consists of 3 steps (step two can be executed multiple times)

1. **Subscription Setup – including a first payment**

    1. Client redirects Subscriber(-Browsers) to mbe4 Widget URL (Client adds all necessary parameters to the redirect URL).
    2. mbe4 executed the whole payment process including authorization and capturing and creates a subscription.
    3. mbe4 redirects the subscriber to a given redirect URL. The redirect URL contains all necessary parameters a client needs.

       Positive result

    5. Client will perform the delivery of service

       Negative result

    6. Client informs the subscriber that the transaction failed (and offers different payment methods) NOTE: mbe4 usually does not communicate the result of a payment transaction to the subscriber!

NOTE: a subscriber is only allowed to have one subscription per service! See StatusCode=10

2. **FollowUp Payments – (collecting fees) 1-Phase or 2-Phase Transaction**

    a. **2-Phase**

    1. Client sends "followupauthorize" request.
    2. mbe4 executes authorization and answers with the auth-response

       Positive result

    3. Client will perform the delivery of service
    4. Client sends „Transaction capture" request. Mbe4 answers with a capture result.

       Negative result

    4. Client must persist information about this event. The client can either terminate the subscription or try to send the follow up again later.

    b. **1-Phase**

    1. Client sends "followupdirectcapture" request .
    2. mbe4 executes the Directcapture process. The payment will be authorized and captured in one step.
    3. mbe4 answers with a direct capture response.

       Positive result

    5. Client will perform the delivery of service

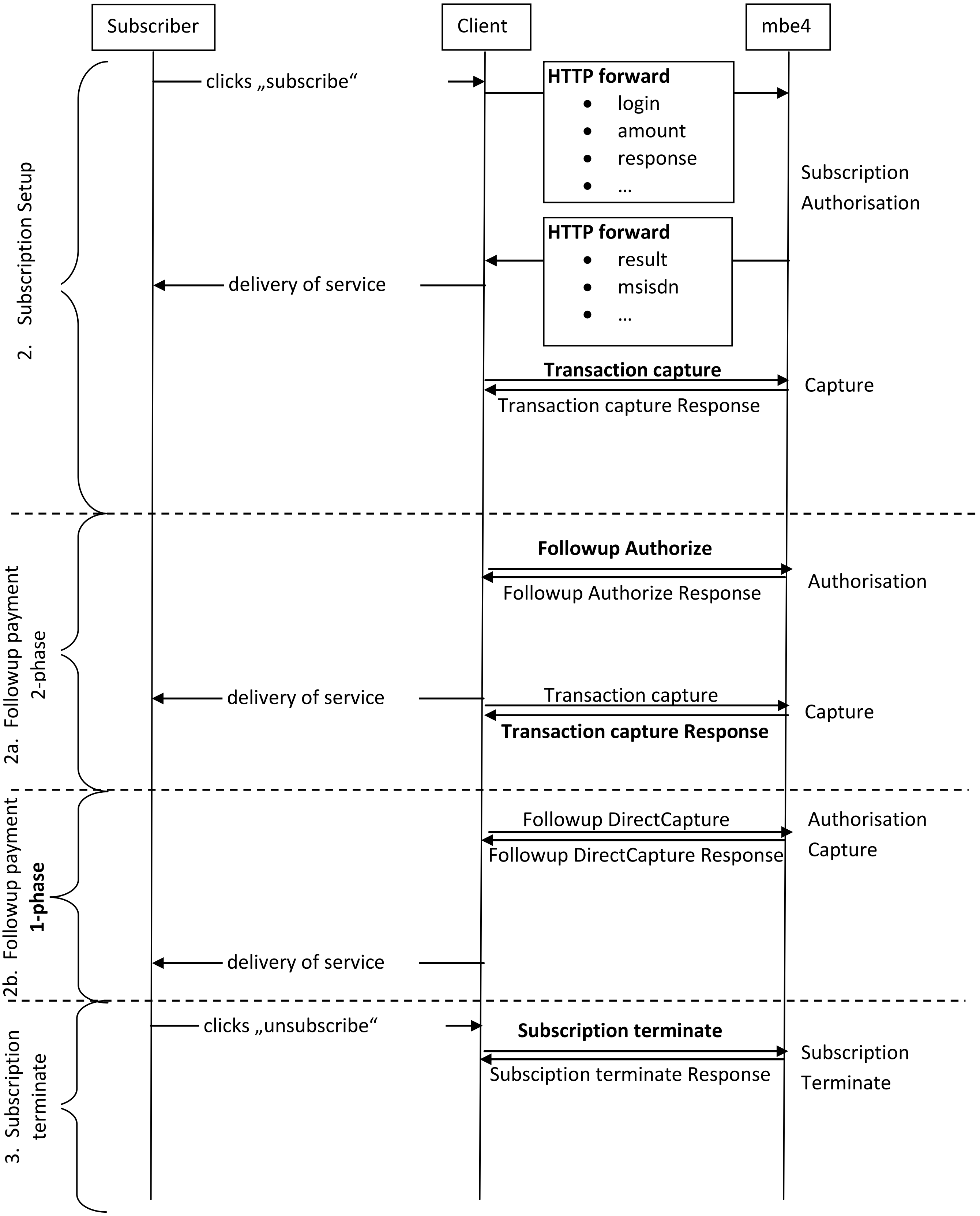
       Negative result

5. Client must persist information about this event. The client can either terminate the subscription or try to send the follow up again later..

3. **Subscription terminate – (terminating a subscription)**
   1. Client sends “subscriptionterminate” request.
   2. mbe4 executes subscription termination.
   3. mbe4 answers with a subscriptionterminate response.

NOTE: mbe4 platform does **NOT** trigger followup payments automatically!

6.4 2 –Phase Subscription Setup



A Subscription consists of 3 steps (step two can be executed multiple times)

4. Subscription Setup – including first payment

4. Client redirects subscriber(-Browsers) to mbe4 Widget URL (Client adds all necessary parameters to the redirect URL).
5. mbe4 executed the whole payment process. The transaction will be authorized and a subscription will be created.
6. mbe4 redirects the subscriber to a given redirect URL. The redirect URL contains all necessary parameters a client needs.

Positive result

6. Client will perform the delivery of service
7. Client sends a „transaction capture“ request. mbe4 answers with a capture result.

Negative result

7. Client informs the subscriber that the transaction failed (and offers different payment methods) NOTE: mbe4 usually does not communicate the result of a payment transaction to the subscriber!

NOTE: a subscriber is only allowed to have one subscription per service! See StatusCode=10

5. **FollowUp Payments**

see 1-Phase Subscription Setup – FollowUp Payments!

6. Subscription terminate

see 1-Phase Subscription Setup – Subscription terminate!

7 Client Authentication

7.1 Widget HTTP forward

A client must send username and hash code with every redirect. The hash code is used sign all parameter values.

7.2 Creating the Hash code

The hash is created using the MD5 algorithm md5(Password+Parametervalue1+Parametervalue2+Parametervalue3…)

+ (plus) stands for the concatenation of the parameter values encoded in UTF-8 (NOT URL-encoded)

The client receives the needed password from mbe.

NOTE: The order of the values is important and has to follow the order how the parameters are specified in this document.

7.3 Methods capture, terminate, refund, followupauthorize, followupdirectcapture, subscriptionterminate, status

These methods are SOA methods.

The communication between client and mbe4 is based on IP-safe https connections.

- The client must send all needed IPs to mbe4
- mbe4 excepts request only from these enabled IPs.
- These requests must be send via SSL (https)

8 Single Payment HTTP Forward Client->mbe4

Subscriber clicks „pay“ Client redirects subscriber to mbe4 Payment Widget SinglePayment URL.
Preconditions: • Subscriber wants to pay • clienttransactionid is unique
Folgeschritte: • wait for redirected client browser on callbackURL

HTTP forward:
http(s)://billing.mbe4.de/widget?
username={username}
&clientid={clientid}
&serviceid={serviceid}
&contentclass={contentclass}
&description={description}
&clienttransactionid={clienttransactionid}
&amount={amount}
&currency={currency}

&callbackurl={callbackurl}
&timestamp={timestamp}
&hash={md5hashcode}				

Parameter				
IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]**{5}**$	clientid provided by mbe4
IN	serviceid	mandatory	^[0-9]**{5}**$	serviceid provided by mbe4
IN	contentclass	mandatory	^[0-9]**{1,2}**$	See list
IN	description	mandatory	^.{1,100}$	Description of Contents
IN	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
IN	amount	mandatory	^[1-9]{1}[0-9]{0,4}$	amountin EUR Cent
IN	currency	optional	^[A-Z]{3}$	currency in ISO 4217 3char code. If currency is not provided mbe4 uses “EUR” as default. Availability of currencies depends on payment methods
IN	callbackurl	mandatory	^http.{12,150}$	Subscriber will be redirected to this URL after a payment process
IN	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-	Timestamp

			9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	
IN	hash	mandatory	^[a-zA-Z0-9]**{32}**$	See „client authentication“

8.1 Single Payment HTTP forward mbe4 -> Client

After a payment process a subscriber will be redirected to the clients CallbackURL
Preconditions: • http forward client -> mbe4 • final status of transaction is reached
Possible following steps: • delivery of service

HTTP forward:
https://{callbackURL}?
transactionid={transactionid}
&clienttransactionid={clienttransactionid}
&responsecode={responsecode}
&description={description}
&subscriberid={subscriberid}
&operatorid={operatorid}
&timestamp={timestamp}
&hash={md5hashcode}

Parameter

Request/ Response	Name	Presence	Format	Description
Request	transactionid		^[0-9]{1,10}$	mbe transactionid
Request	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
Request	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (siehe Liste Responsecodes)
Request	description	mandatory	^.{1,100}$	Responsecode Description
Request	subscriberid	mandatory	(^491[5-7]{1}[0-9]{8,9}$)\|(^I_Token$)	Subscribers mobile number or „I_Token“ for anonymous token based transactions
Request	operatorid		^[-0-9a-zA-Z]{1-20}$	Operator see List
Request	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
Request	hash	mandatory	^[a-zA-Z0-9]**{32}**$	See „client authentication“

9 Transaction capture Request

Transaction capture

Transaction capture finalizes a 2-Phase transaction process and captures the transaction .

Preconditions:

- Two phase service
- Responsecode=0 from Authorization
- Delivery of service was successful

Possible following steps:

- If result is “not final” wait for callback.

Transaction capture request(HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
transactionid={transactionid}
&username={username}
&clientid={clientid}
&password={md5(password)}
&do=capture
&callbackurl={callbackurl}
&timestamp={timestamp}
```

Transaction capture response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]**{5}**$	clientid provided by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]**{32}**$	md5 encoded Password

IN	transactionid		^[0-9]{1,10}$	mbe transactionid
IN	do	mandatory	^capture$	String “capture”
IN	callbackurl	mandatory	^http.{12,150}$	URL for callback requests
IN	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list Responsecodes)
OUT	description	mandatory	^.{1,100}$	Responsecode description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	timestamp

10 Transaction terminate Request

Transaction terminate

Transaction terminate terminates a not captured (authorized) transaction.

Preconditions:

- Two phase service
- authorized Transaction

Possible following steps:

Transaction terminate request (HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
transactionid={transactionid}
&username={username}
&clientid={clientid}
&password={md5(password)}
&do=terminate
&reason={reason}
&callbackurl={callbackurl}
&timestamp={timestamp}
```

Transaction terminate response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]**{5}**$	clientid provided by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]**{32}**$	md5 encrypted Password
IN	transactionid		^[0-9]{1,10}$	mbe transactionid

IN	do	mandatory	^terminate$	String “terminate”
IN	reason	optional	^[-_*+()&!?:.;,&öäüÄÖÜß\sa-zA-Z0-9]{1,100}$	Reason for termination
IN	callbackurl	mandatory	^http.{12,150}$	URL for callback requests
IN	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (siehe Liste Responsecodes)
OUT	description	mandatory	^.{1,100}$	Responsecode Description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid

11 Transaction refund Request

Transaction refund Refund roles back a captured transaction.
Preconditions: • Transaction is captured
Possible following steps:

Transaction refund request (HTTP GET/POST):
 	
https://billing.mbe4.de/http/transaction?
transactionid={transactionid}
&username={username}

```
&clientid={clientid}
&password={md5(password)}
&do=refund
&reason={reason}
&callbackurl={callbackurl}
&timestamp={timestamp}
```

Transaction refund response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted Password
IN	transactionid		^[0-9]{1,10}$	mbe transactionid
IN	do	mandatory	^refund$	String “refund”
IN	reason	optional	^[-_*+()&!?:.;,&öäüÄÖÜß\sa-zA-Z0-9]{1,100}$	reason for this refund
IN	callbackurl	mandatory	^http.{12,150}$	URL for callback requests
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel:	Timestamp

			2009-01-01T10:00:00.000Z	
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list Responsecodes)
OUT	description	mandatory	^.{1,100}$	responsecode description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp

12 Transaction Status

Transaction Status

Transaction status can be used to retrieve the current status of an existing transaction.

Preconditions:

- Transaction exists

Transaction Status request (HTTP GET/POST):

https://billing.mbe4.de/http/transaction?
transactionid={transactionid}
&clienttransactionid={clienttransactionid}
&serviced={serviceid}
&username={username}
&clientid={clientid}
&password={md5(password)}
&do=status
&timestamp={timestamp}

Transaction status response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&status={transactionstatus}
&description={description}
&operatorid={operatorid}
&transactionid={ transactionid }
&subscriberid={subscriberid}
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted Password
IN	transactionid	mandatory wenn clienttransactionid and serviceid nicht vorhanden sind	^[0-9]{1,10}$	mbe transactionid
IN	clienttransactionid	mandatory wenn transactionid nicht vorhanden ist	^[-a-zA-Z0-9_]{1,95}$	mbe clienttransactionid
IN	serviceid	mandatory wenn transactionid nicht vorhanden ist	^[0-9]{5}$	mbe serviceid
IN	do	mandatory	^refund$	String “status”
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-	Timestamp

			9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list Responsecodes)
OUT	status	mandatory	^[0-9]{1,4}$	Current transaction status
OUT	description	mandatory	^.{1,100}$	Status Description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	subscriberid	mandatory	^491[5-7]{1}[0-9]{8,9}$	subscribers mobile number
OUT	operatorid	mandatory	^[-0-9a-zA-Z]{1-20}$	Operator see List
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp

13 Subscription Setup HTTP Forward Client->mbe4

Subscriber clicks „subscribe“

Client redirects subscriber to mbe4

Preconditions:

- Subscriber clicks „subscribe“
- clienttransactionid is unique
- subscriptionid is unique

Possible following steps:

HTTP forward:

```
http(s)://billing.mbe4.de/widget?
username={username}
&clientid={clientid}
&serviceid={serviceid}
&contentclass={contentclass}
&description={description}
&clienttransactionid={clienttransactionid}
&amount={amount}
&callbackurl={callbackurl}
&subscriptionid={subscriptionid}
&subscriptiondescription={subscriptiondescription}
&subscriptioninterval={subscriptioninterval}
&timestamp={timestamp}
&hash={md5hashcode}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Clients username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4
IN	serviceid	mandatory	^[0-9]{5}$	serviceid provided by mbe4
IN	contentclass	mandatory	^[0-9]{1,2}$	contentclass (see

				list)
IN	description	mandatory	^.{1,100}$	Description of content
IN	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
IN	amount	mandatory	^[1-9]{1}[0-9]{0,4}$	amount in EUR Cent
IN	callbackurl	mandatory	^http.{12,150}$	URL mbe4 uses to redirect the subscriber to after payment process is finished
IN	subscriptionId	mandatory	^[a-zA-Z0-9]{1,32}$	unique subscriptionId created by the client
IN	subscription description	mandatory	^[a-zA-Z0-9 \.,!?\-]{1,20}$	Description of this subscription
IN	subscriptioninterval	mandatory	^[0-9]{1,3}$	Interval between two (planned) followup payments in days
IN	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
IN	hash	mandatory	^[a-zA-Z0-9]**{32}**$	See „client authentication“

13.1 Subscription Setup HTTP forward mbe4 -> Client

After a payment process a subscriber will be redirected to the clients CallbackURL

Preconditions:

- http forward client -> mbe4
- final status of transaction is reached

Possible following steps:

- delivery of service
- if 2Phase : „capture“ transaction

HTTP forward:

```
https://{callbackurl}?
 transactionid={transactionid}
&clienttransactionid={clienttransactionid}
&responsecode={responsecode}
&description={description}
&subscriberid={subscriberid}
&operatorid={operatorid}
&timestamp={timestamp}
&subscriptionid={subscriptionid}
&hash={md5hashcode}
```

Parameter

Request/ Response	Name	Presence	Format	Description
Request	transactionid	mandatory	^[0-9]{1,10}$	mbe transactionid
Request	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
Request	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (siehe Liste Responsecodes)
Request	description	mandatory	^.{1,100}$	Responsecode Description
Request	subscriberid	mandatory	(^491[5-7]{1}[0-	Subscribers mobile number or „I_Token“

			9]{8,9}$)\|(^I_Token$)	for anonymous token based transactions
Request	operatorid	mandatory	^[-0-9a-zA-Z]{1-20}$	operator see list
Request	timestamp	mandatory	^[2]**{1}**[0]**{1}**[0-2]**{1}**[0-9]**{1}**[-]**{1}**(([0]**{1}**[0-9]**{1}**)|([1]**{1}**[0-2]**{1}**))[-]**{1}**[0-3]**{1}**[0-9]**{1}**[T]**{1}**[0-2]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[:]**{1}**[0-5]**{1}**[0-9]**{1}**[.]**{1}**[0-9]**{3}**[Z]**{1}**$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
Request	subscriptionid	mandatory	^[a-zA-Z0-9]{1,32}$	Unique subscription id created by the client
Request	hash	mandatory	^[a-zA-Z0-9]**{32}**$	See „client authentication“

14 Subscription followupauthorize Request

Subscription followupauthorize Followupauthorize creates a 2-Phase followup transaction within an existing subscription.
Preconditions: • Active Subscription
Possible following steps:
Subscription followupauthorize request(HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
subscriptionid ={subscriptionid}
&subscriptiondescription={subscriptiondescription}
&subscriptioninterval={subscriptioninterval}
&username={username}
&clientid={clientid}
&password={md5(password)}
&serviceid={serviceid}
&contentclass={contentclass}
&description={description}
&clienttransactionid={clienttransactionid}
&amount={amount}
&subscriberid={subscriberid}
&do=followupauthorize
&callbackurl={callbackurl}
&ordertype={ordertype}
&timestamp={timestamp}
```

Subscription followupauthorize response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
&or={operatorresponse}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	subscriptionid	mandatory	^[a-zA-Z0-9]{1,32}$	subscriptionid of subscription
IN	subscriptiondescription	mandatory	^[a-zA-Z0-9 \.,!?\-]{1,20}$	Subscriptions description
IN	subscriptioninterval	mandatory	^[0-9]{1,3}$	Interval between 2 followups
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Username des Client
IN	clientid	mandatory	^[0-9]{5}$	clientid provided

				by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted password
IN	serviceid	mandatory	^[0-9]{5}$	serviceid provided by mbe4
IN	contentclass	mandatory	^[0-9]{1,2}$	Contentclass see list
IN	description	mandatory	^.{1,100}$	Content description
IN	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
IN	amount	mandatory	^[1-9]{1}[0-9]{0,4}$	Amount in cent (must be not higher than the amount of the subscription setup transaction)
IN	subscriberid	mandatory	^491[5-7]{1}[0-9]{8,9}$	Subscribers mobile number
IN	do	mandatory	^followupauthorize$	String “followupauthorize”
IN	callbackurl	mandatory	^http.{12,150}$	URL mbe4 uses to send callback to
IN	ordertype	mandatory	^web|wap$	Ordertype see list
IN	synchron	optional	^true$	This parameter enforches mbe4 to answer synchron. That causes long lasting requests NOT

				RECOMMENDED !
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list Responsecodes)
OUT	description	mandatory	^.{1,100}$	Responsecode Description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp

15 Subscription followupdirectcapture Request

Subscription followupdirectcapture

Creates a one phase follow up transaction within a existing subscription

Preconditions:

- Active subscription exists

Possible following steps:

- If result is “not final” wait for callback.

Subscription followupdirectcapture request(HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
subscriptionid ={subscriptionid}
&subscriptiondescription={subscriptiondescription}
&subscriptioninterval={subscriptioninterval}
&username={username}
&clientid={clientid}
&password={md5(password)}
&serviceid={serviceid}
&contentclass={contentclass}
&description={description}
&clienttransactionid={clienttransactionid}
&amount={amount}
&subscriberid={subscriberid}
&do=followupdirectcapture
&callbackurl={callbackurl}
&ordertype={ordertype}
&timestamp={timestamp}
```

Subscription followupdirectcapture response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	subscriptionid	mandatory	^[a-zA-Z0-9]{1,32}$	subscriptionid of subscription
IN	subscriptiondescription	mandatory	^[a-zA-Z0-9 \.,!?\-]{1,20}$	Subscriptions description
IN	subscriptioninterval	mandatory	^[0-9]{1,3}$	Interval between 2 followups

IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	Username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4
IN	password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted Password
IN	serviceid	mandatory	^[0-9]{5}$	serviceid provided by mbe4
IN	contentclass	mandatory	^[0-9]{1,2}$	Contentclass see list
IN	description	mandatory	^.{1,100}$	Content description
IN	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
IN	amount	mandatory	^[1-9]{1}[0-9]{0,4}$	Amount in cent (must be not higher than the amount of the subscription setup transaction)
IN	subscriberid	mandatory	^491[5-7]{1}[0-9]{8,9}$	Subscribers mobile number
IN	do	mandatory	^ followupdirectcapture$	String “followupdirectcapture”
IN	callbackurl	mandatory	^http.{12,150}$	URL mbe4 uses to send callback to
IN	ordertype	mandatory	^web|wap$	ordertype
IN	synchron	optional	^true$	This parameter enforces mbe4 to answer synchron. That causes long lasting requests NOT RECOMMENDED!
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-	Timestamp

			]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list)
OUT	description	mandatory	^.{1,100}$	Responsecode Description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	or	optional	^.{0,20}[,]{1}.{0,20}[,]{1}.{0,20}[,]{1}.{0,20}$	OperatorResponse - can contain more detailed reasons of transaction rejections

16 Subscription terminate Request

Subscription terminate

Subscription terminate terminates an active subscription

Preconditions:

- Active subscription

Possible following steps:

Subscription terminate request (HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
subscriptionid ={subscriptionid}
&username={username}
&clientid={clientid}
&password={md5(password)}
&serviceid={serviceid}
&clienttransactionid={clienttransactionid}
&reason={reason}
&subscriberid={subscriberid}
&do=subscriptionterminate
&callbackurl={callbackurl}
&ordertype=web
&timestamp={timestamp}
```

Subscription terminate response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}

responsecode={responsecode}
&description={description}
&transactionid={ transactionid }
&timestamp={timestamp}
```

Parameter

IN/OUT	Name	Presence	Format	Description
IN	subscriptionid	mandatory	^[a-zA-Z0-9]{1,32}$	subscriptionid of subscription
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4

IN	password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted password
IN	serviceid	mandatory	^[0-9]{5}$	serviceid provided by mbe4
IN	clienttransactionid	mandatory	^[-a-zA-Z0-9_]{1,95}$	Unique transaction id created by the client
IN	reason	mandatory	^[-_*+()&!?:.;,&öäüÄÖÜß\sa-zA-Z0-9]{1,100}$	Reason for this termination
IN	subscriberid	mandatory	^491[5-7]{1}[0-9]{8,9}$	subscribers mobile number
IN	do	mandatory	^subscriptionterminate$	String “subscriptionterminate”
IN	callbackurl	mandatory	^http.{12,150}$	URL mbe4 uses to send callback to
IN	ordertype	mandatory	^web|wap$	Ordertype der Subscription.
IN	synchron	optional	^true$	This parameter enforches mbe4 to answer synchron. That causes long lasting requests NOT RECOMMENDED!
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (see list)

OUT	description	mandatory	^.{1,100}$	Responsecode Description
OUT	transactionid	mandatory	^[0-9]{1,10}$	Mbe transactionid
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp

17 Subscription Status

Subscription Status

Subscription Status can be used to retrieve the status of an existing subscription

Preconditions:

- Subscription exists

Transaction Status request (HTTP GET/POST):

```
https://billing.mbe4.de/http/transaction?
username={username}
&clientid={clientid}
&password={md5(password)}
&serviced={serviceid}
&do=subscriptionstatus
&subscriptionid={subscriptionid}
&timestamp={timestamp}
```

Transaction status response (HTTP GET/POST):

```
HTTP/1.1 200 OK
Content-Length: {content length}
```

responsecode={responsecode}
&subscriptionstatus ={subscriptionstatus}
&description={description}
&subscriberid={ subscriberid }
&operatorid={ operatorid}
&timestamp={timestamp}				

Parameter				
IN/OUT	Name	Presence	Format	Description
IN	username	mandatory	^[-a-zA-Z0-9_]{10,30}$	username provided by mbe4
IN	clientid	mandatory	^[0-9]{5}$	clientid provided by mbe4
IN	Password md5 encrypted	mandatory	^[a-zA-Z0-9]{32}$	md5 encrypted Password
IN	serviceid	mandatory	^[0-9]{5}$	serviceid provided by mbe4
IN	do	mandatory	^subscriptionstatus$	String “subscriptionstatus”
IN	subscriptionid	mandatory	^[a-zA-Z0-9]{1,32}$	Subscriptionid of Subscription
IN	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp
OUT	responsecode	mandatory	^[0-9]{1,6}$	0=OK … (siehe Liste Responsecodes)

OUT	subscriptionstatus	mandatory	^[0-9]{1,4}$	Status of Subscription
OUT	description	mandatory	^.{1,100}$	Status Description
OUT	subscriberid	mandatory	^491[5-7]{1}[0-9]{8,9}$	subscribers mobile number
OUT	operatorid	mandatory	^[-0-9a-zA-Z]{1-20}$	Operator see List
OUT	timestamp	mandatory	^[2]{1}[0]{1}[0-2]{1}[0-9]{1}[-]{1}(([0]{1}[0-9]{1})|([1]{1}[0-2]{1}))[-]{1}[0-3]{1}[0-9]{1}[T]{1}[0-2]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[:]{1}[0-5]{1}[0-9]{1}[.]{1}[0-9]{3}[Z]{1}$ Format: YYYY-MM-DDTHH:MM:SS.mmmZ Beispiel: 2009-01-01T10:00:00.000Z	Timestamp

18 List Responsecodes

Statuscode	Description
0	OK
1	NOT FINAL – request was processed successfully but the answer is not final. (e.g. a TAN was sent to the subscriber and tan received must be called) This status is not used anymore
2	authorization failed
3	capture failed
4	terminate failed
5	refund failed

6	prepair failed
7	transaction failed
8	subscription terminate failed
10	subscriber already has an active subscription
11	fraud prevention (mbe4 did not allow to execute this transaction)
12	Captcha wrong
101	invalid parameter
109	transaction in wrong status
110	wrong PIN
111	too many PIN attempts - transaction closed
112	subscriber aborted transaction
113	no route to operator or subscriberid invalid
121	subscriberid unascertainable
126	sending TAN SMS failed
150	subscriptionid unknown
151	subscriptionid not unique
152	subscription terminated
200	internal server error
201	system currently unavailable
1001	transaction failed after TAN transmission

19 List Contentclasses

ContentClass	Description
1	News/Info
2	Chat/Flirt

3	Game
4	Ringtone
5	Image/Logo
6	Videoclip
7	Music File
8	Localization
9	Voting
10	Competition
11	Portal access
12	Software
13	Document
14	Ticket
15	Horoskop
16	Leisure
17	Outdoor
18	Finances
19	Shopping
20	E-Mail
21	Charity
22	Erotic

20 List Operatorids

ID	Operator
262-01	T-Mobile Germany
262-02	Vodafone D2 Germany

262-07	O2 Germany
262-MODE	Mobilcom Debitel Germany
PP	PayPal
HEI_CC	CreditCard
HEI_ELV	DirectDebit
MPASS	mpass
CAB	Click and Buy
GIRO	Giropay

21 List TAN SMS Texts

OperatorID	Text	Absender
262-01	*[Client]*: Zum Bezahlen von *[x.x]* Euro geben Sie bitte folgenden Bezahl-Code ein: *PIN* (Vorgang *transaction-#*)	+491234 (BezahlCode)
262-02	Vodafone: Zum Bezahlen von *xx.xx* Euro bei *Vendor-Name* geben Sie bitte folgenden Bezahl-Code beim Händler ein: *PIN* (Vorgang *transaction-#*)	**6729** (mpay)
262-03	Anbei erhalten Sie den Bezahlcode zur Bestellung eines kostpflichtigen Dienst in Höhe von *xx,xx* EURO: *PIN*	1232111
262-07	Zum Bezahlen von xx,xx Euro für ihren Service *Description* bei *Vendor-Name* geben Sie bitte Bezahlcode *PIN* ein. Mit der Eingabe lösen Sie einen Zahlungsvorgang aus.	66245
262-MODE	configurable	configurable

22 List Transactionsstatus

Status	Description
-1	UNKNOWN
0	CREATED
1	TAN-SMS SENT

2	WAITING FOR CALLBACK (TAN)
3	TAN RECEIVED
4	AUTHORIZATION CREATED
5	WAITING FOR CALLBACK (AUTHORIZATION)
6	AUTHORIZED
7	CAPTURE CREATED
8	WAITING FOR CALLBACK (CAPTURE)
9	CAPTURED
10	TERMINATE CREATED
11	WAITING FOR CALLBACK (TERMINATE)
12	TERMINATED
13	REFUND CREATED
14	WAITING FOR CALLBACK (REFUND)
15	REFUNDED
16	DIRECTCAPTURE CREATED
17	WAITING FOR CALLBACK (DIRECTCAPTURE)
19	DIRECTCAPTURE_ONETIMEPIN CREATED
20	DIRECTCAPTURE_ONETIMEPIN TAN REQUESTED
21	DIRECTCAPTURE_ONETIMEPIN TAN SENT
22	DIRECTCAPTURE_ONETIMEPIN TAN RECEIVED
23	WAITING FOR CALLBACK (DIRECTCAPTURE_ONETIMEPIN)
24	AUTHORIZE_ONETIMEPIN CREATED
25	AUTHORIZE_ONETIMEPIN TAN REQUESTED
26	AUTHORIZE_ONETIMEPIN TAN SENT
27	AUTHORIZE_ONETIMEPIN TAN RECEIVED
28	WAITING FOR CALLBACK (AUTHORIZE_ONETIMEPIN)
29	SUBSCRIPTION_SETUP_ONETIMEPIIN CREATED
30	SUBSCRIPTION_SETUP_ONETIMEPIIN TAN REQUESTED
31	SUBSCRIPTION_SETUP_ONETIMEPIIN TAN SENT
32	SUBSCRIPTION_SETUP_ONETIMEPIIN TAN RECEIVED
33	WAITING FOR CALLBACK (SUBSCRIPTION_SETUP_ONETIMEPIIN)
34	SUBSCRIPTION_SETUP CREATED
35	WAITING FOR CALLBACK (SUBSCRIPTION_SETUP)
36	SUBSCRIPTION_TERMINATE CREATED
37	WAITING FOR CALLBACK (SUBSCRIPTION_ TERMINATE)
38	SUBSCRIPTION TERMINATED

201	CREATED WEBFLOW
202	INITIALIZED WEBFLOW
203	AUTHORIZED WEBFLOW
204	CAPTURED WEBFLOW
205	REFUNDED WEBFLOW
206	CHARGEBACK
299	TERMINATED WEBFLOW

23 List Subscriptionstatus

Status	Name	Description
0	CREATED	Subscription is created but not active
1	ACTIVATED	Subscription is active. Follow-ups are allowed
2	TERMINATED	Subscription is terminated
3	LOW_MONEY	Subscription can be ACTIVATED with an successful followup