SERGEL CONNECT SMS REST API

MT and Delivery Reports

Version 2.8; Last updated September 23, 2026

Contents

Contents 2
Before you begin 4
Scope of this document 4
Capabilities of “Common” platform 4
Terms and glossary 5
Size limits 5
MT 5
MO 5
Charged, Premium 5
Bulk 5
Delivery Report 5
TON 5
KeyValue 5
Character Encoding 5
IP Addresses 5
Sending MT messages 6
Base URL:s 6
Authentication 6
HTTP Methods, statuses, and actions 7
Methods 7
DCS 9
TON 9
Priority 10
Error Result Codes 10
Example 11
Success Result 11
Batch sending MT messages 12
batchSendRequest 12
Batch sending example 18
Success Result 18

Quota 19
Quota Overview 19
Status 106 – Quota Exceeded 20
Sending flash sms 20
Scheduled delivery of MT messages 22
Example 22
Obfuscation 23
Obfuscation Example 23
Delivery Reports 24
Result Codes 26
Delivery Report Example 29
Appendix 1 30
Appendix 2 32
Silent Billing 32
Norway (Strex) only 32
Appendix 3 33
Supported TLS versions 33
Supported Ciphers 33
Changelog of this document 34

Before you begin

Please make sure that Sergel connect Support has provided you with the following information:

`Username, Password, platformId, platformPartnerId`

If you will be receiving Delivery Reports for your messages, please provide Sergel connect Support with an URL and they will also give you a **`gateId`** to use. For more information on Delivery Reports, see the “Delivery Reports” chapter.

To use Delivery Reports, make sure you have made an opening in any firewalls so that Common can connect to you to transfer Delivery Reports. The addresses to open for are listed below.

Scope of this document

This document will describe the Application Programming Interface (API) to send text messages through the Sergel connect “Common” platform. It will also describe the mechanism for delivering, to your platform, Delivery Reports for each message sent.

A separate document describes the API for *receiving* text messages.

Common is a REST API. This means it uses HTTP verbs to receive commands. A basic familiarity with REST APIs is assumed, as well as a familiarity with JSON.

Capabilities of “Common” platform

Common is a high-capacity, high-availability SMS Gateway designed to let you send and receive SMS Text messages, as well as receive a notification when the text message is received by the end-user.

A message can be free for the end-user to receive, or it can cost money. (Certain countries only). In certain markets, you can charge the end-users without actually sending them a message, so-called "Silent billing".

A message can be of any length up to the maximum defined by the GSM standard (254 segments)

It can contain any character in the UTF-8 2-byte character set. (Unicode 4-byte characters are not supported).

When sending free messages, the sender of the message can be set to any string of 2-11 characters, a-Z,0-9 (Must begin with a non-numeric character).

Common tracks the status of each message every step of the way until it is delivered to the end-user’s handset, and will provide you with this status through a Delivery Report. Delivery Reports can be sent in JSON, HTTP GET or POST formats.

Terms and glossary

Size limits

An SMS Text message can be a maximum of 140 bytes. With the most common character encoding, GSM-7, this translates to 160 characters. If your message is longer than 140 bytes, it must be split into multiple messages, and preceded by a header signifying that it is a multipart message. Common can handle this splitting, concatenation, and the overhead unless you want to do it yourself.

MT

Mobile Terminated. Refers to any SMS message which is sent to a mobile phone. (The message is terminated, or “ends”, at the phone.)

MO

Mobile Originated. Refers to any SMS message which is sent from a mobile phone. (The message’s origin, or beginning, is at the phone.)

Charged, Premium

An MT message can cost money for the end-user to receive. This is usually referred to as a charged message or premium message. If you are going to send charged messages to end-users, please review the rules and regulations for your country. Charged messages are only available in some countries and shortcodes. In certain markets, you can charge the end-users without actually sending them a message, so-called "Silent billing". Support will be happy to advise you if you are in doubt.

Bulk

A message which does not cost money for the end-user to receive. Bulk messages can set their Source (the “From”-field) of the message to any text, 2-11 characters a-Z. Using this feature to impersonate other parties will lead to a termination of your account.

Delivery Report

For each MT message we send, we can send you an acknowledgement when the message is confirmed received by the end-user’s handset. If the message fails for any reason, we will inform you about this. Delivery reports are mandatory for charged messages, optional for bulk.

TON

Type of Number. This identifies how systems shall interpret your Source (your “From”-field). It can be a Shortcode, an alphanumeric string, or a phone number (MSISDN). Same applies for the Destination, or recipient, of the message, though destination will almost always be an MSISDN.

KeyValue

Map with string key and string value where you may specify unique parameters.

Character Encoding

All communication to and from Common will be using UTF-8 encoding.

IP Addresses

When Common is delivering a Delivery Report to you, the requests can come from several different IP addresses.
Appendix 1 contains the hostnames and IP addresses that are currently active.
Please configure your firewalls so that these hosts/networks can connect to your systems to deliver messages.

Sending MT messages

Base URL:s

You will get one of these URL:s assigned to you when your account is created:

URL	IP
https://eu.linkmobility.io/sms	213.242.87.34, 62.67.62.97 Automatic failover between datacenters
https://n-eu.linkmobility.io/sms	213.242.87.34
https://c-eu.linkmobility.io/sms	62.67.62.97
https://s-eu.linkmobility.io/sms	217.163.95.194
~~**https://no.linkmobility.io/sms**~~	Decommissioned 2024-08-31
~~**https://deb.linkmobility.io/sms**~~	Decommissioned 2023-09-22
https://wsx.sp247.net/sms	194.71.165.27, 195.84.162.140 Legacy

Authentication

There are two ways of doing authentication: OAuth 2.0 or Basic Authentication.

OAuth 2.0

The preferred method of authentication is using OAuth 2.0. It requires the client to obtain a token to be used in other requests.

The only grant type currently supported is “client_credentials” where the “client_id” is the username and “client_secret” is the password provided by Support.

This would be posted to the access token URL as:

```
POST /auth/token
Content-Type: application/x-www-form-urlencoded

grant_type=client_credentials&client_id=user&client_secret=secret
```

Successful result with HTTP status 200:

```
{"access_token":"tokenxxx","token_type":"Bearer","expires_in":3599}
```

The access_token value should be used as a header in requests to be used for authorization:

```
Authorization: Bearer tokenxxx
```

The max age for a token is specified in the expires_in field in seconds.

Unsuccessful results will return a different HTTP status than 200 and the body will contain an error parameter in a JSON object with any of the following results: *invalid_request*, *unsupported_grant_type*, *invalid_client* or *internal_error*.

Basic Authentication

Authenticate in the HTTP request using Basic Authentication with the username and password provided by Support.

HTTP Methods, statuses, and actions

HTTP Method	Message sent	Message sent, no response	No access	Invalid request	Invalid login
POST	**200 OK** Returns SendResponse	**204 No Content**	**403 Forbidden** Returns ErrorResponse	**400 Bad Request** Returns ErrorResponse	**401 Unauthorized** Returns ErrorResponse

Methods

POST /sms/send

Submits a message object for delivery to a mobile phone. Set `Content-Type: application/json` in your request header and POST a JSON object with the following properties:

Parameter	Data type	Description
source	String	Required. This is the source number from where the message should be sent. The format is depending on the specified sourceTON.
sourceTON	TON	This is the source type of number. See allowed TON values below. Default ALPHANUMERIC.
destination	String	Required. This is the destination number. The format is depending on the specified destinationTON. Remember that MSISDNS include the country code and a leading plus sign. (+)
destinationTON	TON	This is the destination type of number. See allowed TON values below. Default MSISDN.
dcs	DCS	**Advanced.** This is the Data Coding Scheme that should be used when sending the SMS. See allowed DCS values in a separate table. Default TEXT.
userDataHeader	String	**Advanced.** This value may be specified when sending concatenated SMS, WAP-push, etc. The format is hex encoded 8-bit bytes. More information about valid UDH for long SMS may be given by Support upon request. Common will handle the splitting and

Parameter	Data type	Description
		concatenation of messages if you do not have a specific reason to do it yourself.
userData	String	This is the message content itself. The DCS specifies the format (encoding) on this value. Note that messages that messages of more than 140 bytes must be split into multiple messages. Common will do that automatically by default.
useDeliveryReport	Boolean	True indicates that a delivery report should be sent back when the message has come to a final state. (Delivered or failed) TRUE is mandatory for premium messages. Defaults to TRUE, and it is recommended to use delivery reports.
deliveryReportGates	List <String>	One or more gates that should be used for sending delivery reports. If you do not specify any Gates to deliver Delivery Reports to, make sure to set useDeliveryReport to FALSE. See the chapter on delivery reports for more information. Required for premium messages.
relativeValidityTime	Long	This specifies how long the message is supposed to live. If the message takes longer to deliver to the handset than the validityTime, the message will be discarded. The value is specified in milliseconds. Default is 48 hours (172800000).
absoluteValidityTime	Date	The absolute time when a message should expire. Minimum 15 minutes and maximum 48h in the future. Formatted according to RFC3339, e.g. 2010-03-30T12:59:40+02:00. Overrides relativeValidityTime if both are set.
tariff	Integer	Price, in local currency, in 1/100 of currency. For example, to send a message costing 5 NOK, this should be set to 500. If you are splitting a long message into multiple segments yourself, set price only on the first segment. Default 0.
currency	Currency	The currency should be set if the default country currency not to be used. Supported currencies are NOK, SEK, DKK, EUR, LTL. Ignored for non-premium messages.
vat	Integer	***Deprecated*** *- implementations should not use this field any more.* The VAT that used for the premium transaction, default differ per market. 2500 equals 25%. Absence or

Parameter	Data type	Description
		value = -1 means not set. Ignored for non-premium messages.
age	Integer	Allowed age for (adult) content. Optional. Not supported by all operators.
priority	Priority	See the **Priority** value table, Optional.
platformId	String	Your platformId. Provided to you by Support.
platformPartnerId	String	Your platformPartnerId. Provided to you by Support.
refId	String	Your own internal transaction ID. Not used for anything except as a reference. Optional.
productDescription	String	When sending premium messages, a description of the service. This will be printed on the end-user's phone bill. Ignored for non-premium messages.
productCategory	Integer	When sending premium messages, specify which category the service is. This lets the operator know which rates to apply to the message. Support or your sales contact will help you determine the correct productCategory to set. Ignored for non-premium messages.
moReferenceId	String	A reference to the ID of the MO message which triggered the MT message. Required for some operators.
customParameters	KeyValue	**Advanced.** Additional parameters may be specified if needed. Support will advise you if you need to use custom parameters.
ignoreResponse	Boolean	Indicates whether you want a response in the body when you submit the message. This is not a delivery report, only a confirmation of message submission. Default is false.

DCS

Data Coding Scheme sets the encoding used for the message. Default and recommended is TEXT.

DCS value	Description
GSM	GSM-7 default alphabet encoding.
BINARY	8-bit binary data.
UCS2	UCS-2 encoding
TEXT	Server-side handling of encoding and segmenting. Recommended.

TON

TON stands for Type of Number and designates how a number is to be interpreted.

TON value	Description
SHORTNUMBER	Shortnumber; 1-14 digits depending on country.

TON value	Description
ALPHANUMERIC	Up to 11 valid GSM characters. Some operators and some handsets don't accept all characters. Safe characters are a-z, A-Z, 0-9.
MSISDN	A mobile number, international format, starting with +.

Priority

It gives the sending a certain priority.

Priority value	Description
HIGH	For high priority of sending
NORMAL	For normal priority of sending
LOW	For low priority of sending

Error Result Codes

Result Code	Description
101100	Invalid authentication. Please check your username and password.
101101	Access denied. Please check your username and password.
106000	Unknown Error. Please contact Support and include your whole transaction.
106001	Parse Error. The object to send in the request is badly formatted.
106102	Unable to access SMSC credentials
106200	Invalid or missing platformId. Please check your platformId.
106201	Invalid or missing platformPartnerId. Please check your platformPartnerId.
106202	Invalid or missing currency for premium message. Please check your price and currency.
106300	No gates available. Please contact Support and include your whole transaction.
106301	Specified gate unavailable. Please check your gateId.
106	quota exceeded in process message, messageParts: {}

Example

In this example, the platformId and platformPartnerId and deliveryReportGates are set to invalid values. The values that are correct for you will be provided by Support.

A minimal example, including only required fields. This would send the message “Hello world” to the Norwegian phone number +4799999999, and not use a delivery report. The sender is set to “SERGEL”.
This JSON would be POSTed to **https://[your assigned URL]/sms/send**

```
{
  "source": "SERGEL",
  "destination": "+4799999999",
  "userData": "Hello world",
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": false
 }
```

Success Result

On a successful request, Common will reply with HTTP 200 OK, or HTTP 204 No Content if “ignoreResponse” is set to TRUE.
In the body you will find the messageId of the message:

```
{
      "messageId":"Dcshuhod0PMAAAFQ+/PbnR3x",
      "resultCode":1005,
      "description":"Queued"
}
```

If the customParameter “replySmsCount” with the case insensitive String value “true” is found in the sending request, then the reply will have an extra parameter called “smsCount” that has an integer value, it shows the amount of message parts or SMS sent per SendRequestMessage.

```
{
      "messageId":"Dcshuhod0PMAAAFQ+/PbnR3x",
      "resultCode":1005,
      "description":"Queued",
      "smsCount":1
}
```

If there’s an invalid value or the case insensitive String value “false”, then the “smsCount” parameter wouldn’t be shown. The returned smsCount is a preliminary value, an estimation, of message parts and the final smsCount will be found in the delivery report.
Please note that this is not a delivery report. Save the messageId; when the delivery report arrives, it will include the same messageId.

Batch sending MT messages

If you want to send many messages at one time, you can use the Batch Sender to send multiple messages at once, reducing connection overhead. You will receive an array of responses when you submit, with the **messageId** and **refId** of each message posted.
Sending a batch MT message is similar to sending a single MT message, except that certain parameters are moved into a **sendRequestMessages** parameter, which you then post an array of.
The names and types and functions of all parameters except sendRequestMessages are the same as if you were sending a single MT message. Delivery reports are handled normally.
The URL for submitting batch messages is
https://[your assigned URL]/sms/sendbatch

batchSendRequest

Parameter	Data type	Description
useDeliveryReport	Boolean	True indicates that a delivery report should be sent back when the message has come to a final state. (Delivered or failed) TRUE is mandatory for premium messages. Defaults to TRUE, and it is recommended to use delivery reports.
deliveryReportGates	List <String>	One or more gates that should be used for sending delivery reports. If you do not specify any Gates to deliver Delivery Reports to, make sure to set useDeliveryReport to FALSE. See the chapter on delivery reports for more information. Required for premium messages.
relativeValidityTime	Long	This specifies how long the message is supposed to live. If the message takes longer to deliver to the handset than the validityTime, the message will be discarded. The value is specified in milliseconds. Default is 48 hours (172800000).
absoluteValidityTime	Date	The absolute time when a message should expire. Minimum 15 minutes and maximum 48h in the future. Formatted according to RFC3339, e.g. 2010-03-30T12:59:40+02:00. Overrides relativeValidityTime if both are set.
priority	Priority	See the **Priority** value table, optional.

Parameter	Data type	Description
platformId	String	Your platformId. Provided to you by Support.
platformPartnerId	String	Your platformPartnerId. Provided to you by Support.
customParameters	KeyValue	**Advanced.** Additional parameters may be specified if needed. Support will advise you if you need to use custom parameters.
ignoreResponse	Boolean	Indicates whether you want a response in the body when you submit the message. This is not a delivery report, only a confirmation of message submission. Default is true.
sendRequestMessages	List <sendRequestMessage>	An array of messages. The maximum number of messages allowed within the array is 1000. See the following table for its content.

sendRequestMessage

Parameter	Data type	Description
source	String	Required. This is the source number from where the message should be sent. The format is depending on the specified sourceTON.
sourceTON	TON	This is the source type of number. See allowed TON values below. Default ALPHANUMERIC.
destination	String	Required. This is the destination number. The format is depending on the specified destinationTON. Remember that MSISDNS include the country code and a leading plus sign. (+)
destinationTON	TON	This is the destination type of number. See allowed TON values below. Default MSISDN.
dcs	DCS	**Advanced.** This is the Data Coding Scheme that should be used when sending the SMS. See allowed DCS values in a separate table. Default TEXT.
userDataHeader	String	**Advanced.** This value may be specified when sending concatenated SMS, WAP-push, etc. The format is hex encoded 8-bit bytes. More information about valid UDH for long SMS may be given by Support upon request. Common will

Parameter	Data type	Description
		handle the splitting and concatenation of messages if you do not have a specific reason to do it yourself.
userData	String	This is the message content itself. The DCS specifies the format (encoding) on this value. Note that messages that messages of more than 140 bytes must be split into multiple messages. Common will do that automatically by default.
tariff	Integer	Price, in local currency, in 1/100 of currency. For example, to send a message costing 5 NOK, this should be set to 500. If you are splitting a long message into multiple segments yourself, set price only on the first segment. Default 0.
currency	Currency	The currency should be set if the default country currency not to be used. Supported currencies are NOK, SEK, DKK, EUR. Ignored for non-premium messages.
vat	Integer	***Deprecated*** *- implementations should not use this field anymore.* The VAT that used for the premium transaction, default differ per market. 2500 equals 25%. Absence or value = -1 means not set. Ignored for non-premium messages.
age	Integer	Allowed age for (adult) content. Optional. Not supported by all operators.
refId	String	Your own internal transaction ID. Not used for anything except as a reference. Optional.
productDescription	String	When sending premium messages, a description of the service. This will be printed on the end-user's phone bill. Ignored for non-premium messages.
productCategory	Integer	When sending premium messages, specify which category the service is. This lets the operator know which rates to apply to the message. Support or your sales contact will help you determine the correct productCategory to set. Ignored for non-premium messages. The acceptable productCategory values are specified under **productCategory values.**
moReferenceId	String	A reference to the ID of the MO message which triggered the MT message. Required for some operators.
customParameters	KeyValue	**Advanced.** Additional parameters may be specified if needed.

Parameter	Data type	Description
		Support will advise you if you need to use custom parameters. These additional parameters are overridden by those that are in the batchSendRequest.

productCategory values

Product categories defined by Sergel connect :

Value	Description
1	CPA_REGULAR_RINGTONES
2	CPA_REGULAR_RINGBACK_TONES
3	CPA_REGULAR_MUSIC_FULL_TRACK
4	CPA_REGULAR_WALLP_ANIM
5	CPA_REGULAR_VIDEOS
6	CPA_REGULAR_NEWS
7	Not in use
8	CPA_REGULAR_LOTTERY
9	Not in use
10	CPA_REGULAR_VOTING
11	CPA_REGULAR_MOBILE_MARKETING
12	Not in use
13	Not in use
14	CPA_REGULAR_COMMUNITY
15	CPA_REGULAR_INFO_SERVICE
16	CPA_REGULAR_MIXED_CONTENT
17	GAS_CHARITY
18	GAS_CONCERT_TICKETS
19	GAS_MEMBERSHIP_FEE
20	GAS_PHYSICAL_GOODS_NON_FOOD
21	MEDIA_CPA_INTERNET_TV
22	MEDIA_CPA_INTERNET_FILM
23	MEDIA_CPA_E_BOOK
24	MEDIA_CPA_E_MAGAZINE
25	MEDIA_CPA_E_NEWSPAPER
26	Not in use
27	CPA_REGULAR_OTHER
28	GAS_CINEMA_TICKETS
29	GAS_BOOK
30	GAS_MUSIC_CD
31	GAS_MAGAZINE
32	GAS_ACCESS_FEE_SPORTS
33	GAS_ZERORATED_SMS

34	GAS_CLASSIFIED_ADS
35	GAS_CLASSIFIED_ADS_ABOVE_PRICE_200
36	Note: --- NOT TO BE USED ----
37	MANUAL_SERVICES
38	PHYSICAL_GOODS_FOOD_DRINK
39	TRANSPORT_BUS
40	TRANSPORT_TRAIN
41	HEALTH_SERVICES
42	PARKING
43	AUTOMATIC NUMBER DIRECTORY SERVICES
44	ALERT SERVICES
45	INFORMATION SERVICES
46	MEDIA
47	MEMBERSHIP_COMMERCIAL
48	FISHING_LICENSE
49	GIFT_CERTIFICATE
50	MANUAL_SERVICE_REGISTERED
51	CINEMA_TICKET
52	ROADTAX
53	NON_HUMANITARIAN_DONATIONS

Strex productCategory values

The list of valid Strex business models: *Strex Payment, Transport, Donation, Media, Information Services, Automatic Number Directory Service, Mobile Content Services, Mobile Content Services 18+, Alert Services, Voting*.

Note: SERGEL CommonLayer product categories do **not use** these Strex business models: **Media, Information Services, Automatic Number Directory Service, Alert Services**.

Value	**Description**	**Strex service code**	**Strex business model**
1	CPA_REGULAR_RINGTONES	03001	MOBILECONTENTSERVICES
2	CPA_REGULAR_RINGBACK_TONES	03002	MOBILECONTENTSERVICES
3	CPA_REGULAR_MUSIC_FULL_TRACK	03002	MOBILECONTENTSERVICES
4	CPA_REGULAR_WALLP_ANIM	03004	MOBILECONTENTSERVICES
5	CPA_REGULAR_VIDEOS	03007	MOBILECONTENTSERVICES
6	CPA_REGULAR_NEWS	02001	MOBILECONTENTSERVICES
7	Not in use	N/A	N/A
8	CPA_REGULAR_LOTTERY	03012	MOBILECONTENTSERVICES
9	Not in use	N/A	N/A
10	CPA_REGULAR_VOTING	04002	VOTING

11	CPA_REGULAR_MOBILE_MARKETING	02001	MOBILECONTENTSERVICES
12	Not in use	N/A	N/A
13	Not in use	N/A	N/A
14	CPA_REGULAR_COMMUNITY	03011	MOBILECONTENTSERVICES
15	CPA_REGULAR_INFO_SERVICE	04001	MOBILECONTENTSERVICES
16	CPA_REGULAR_MIXED_CONTENT	03011	MOBILECONTENTSERVICES
17	GAS_CHARITY	14002	DONATION
18	GAS_CONCERT_TICKETS	05010	STREX-PAYMENT
19	GAS_MEMBERSHIP_FEE	14001	STREX-PAYMENT
20	GAS_PHYSICAL_GOODS_NON_FOOD	15001	STREX-PAYMENT
21	MEDIA_CPA_INTERNET_TV	16001	STREX-PAYMENT
22	MEDIA_CPA_INTERNET_FILM	16002	STREX-PAYMENT
23	MEDIA_CPA_E_BOOK	16003	STREX-PAYMENT
24	MEDIA_CPA_E_MAGAZINE	16004	STREX-PAYMENT
25	MEDIA_CPA_E_NEWSPAPER	16005	STREX-PAYMENT
26	Not in use	N/A	N/A
27	CPA_REGULAR_OTHER	02001	INFORMATIONSERVICES
28	GAS_CINEMA_TICKETS	05010	STREX-PAYMENT
29	GAS_BOOK	15001	STREX-PAYMENT
30	GAS_MUSIC_CD	15001	STREX-PAYMENT
31	GAS_MAGAZINE	15001	STREX-PAYMENT
32	GAS_ACCESS_FEE_SPORTS	13003	STREX-PAYMENT
33	GAS_ZERORATED_SMS	89001	STREX-PAYMENT
34	GAS_CLASSIFIED_ADS	08001	STREX-PAYMENT
35	GAS_CLASSIFIED_ADS_ABOVE_PRICE_200	08001	STREX-PAYMENT
36	Note: --- NOT TO BE USED ----	NA	STREX-PAYMENT
37	MANUAL_SERVICES	02001	STREX-PAYMENT
38	PHYSICAL_GOODS_FOOD_DRINK	10001	STREX-PAYMENT
39	TRANSPORT_BUS	06002	TRANSPORT
40	TRANSPORT_TRAIN	06001	TRANSPORT
41	HEALTH_SERVICES	07001	STREX-PAYMENT
42	PARKING	06005	TRANSPORT
43	AUTOMATIC NUMBER DIRECTORY SERVICES	16008	AUTOMATICNUMBERDIRECTORYSERVICE
44	ALERT SERVICES	02001	ALERTSERVICES
45	INFORMATION SERVICES	02001	INFORMATIONSERVICES

46	MEDIA	16005	MEDIA
47	MEMBERSHIP_COMMERCIAL	14004	STREX-PAYMENT
48	FISHING_LICENSE	13001	STREX-PAYMENT
49	GIFT_CERTIFICATE	09001	STREX-PAYMENT
50	MANUAL_SERVICE_REGISTERD	13006	STREX-PAYMENT
52	ROADTAX	06007	TRANSPORT

Batch sending example

The following JSON would send a message to two recipients at the same time.

```
{
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": true,
  "deliveryReportGates": [
    "BVldZyQt"
  ],
  "sendRequestMessages": [
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination": "+4746910822",
      "userData": "Hello world, first message",
      "refId": "wir7kkw"
    },
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination": "+4741560067",
      "userData": "Hello world, second message",
      "refId": "qts883r"
    }
  ]
}
```

Success Result

On a successful request, Common will reply with HTTP 200 OK, or HTTP 204 No Content if “ignoreResponse” is set to TRUE. In the body you will find an array of Json objects, every object is the result of every message sent, and the messageId of every message too:

```
[
      {
            "messageId": "QC5BGwiuYk0AAAFiQ08nTFOS",
            "refId": “myRefId”,
            "resultCode": 1005,
            "message": "Queued"
      },
      {
```

```
            "messageId": "QC5BHHuqylsAAAFiQ08nX2ph",
            "refId": “myRefId”,
            "resultCode": 1005,
            "message": "Queued"
      }
]
```

If the customParameter “replySmsCount” with the case insensitive String value “true” is found in the sending request, then the reply will have an extra parameter called “smsCount” that has an integer value, it shows the amount of message parts or SMS sent per SendRequestMessage in all the messages sent.

```
[
      {
            "messageId": "QC5BGwiuYk0AAAFiQ08nTFOS",
            "refId": “myRefId”,
            "resultCode": 1005,
            "message": "Queued",
            "smsCount": 1
      },
      {
            "messageId": "QC5BHHuqylsAAAFiQ08nX2ph",
            "refId": “myRefId”,
            "resultCode": 1005,
            "message": "Queued",
            "smsCount": 1
      }
]
```

If there’s an invalid value or the case insensitive String value “false”, then the “smsCount” parameter wouldn’t be shown.
Please note that this is not a delivery report. Save the messageId; when the delivery report arrives, it will include the same messageId.

Quota

Quota Overview

A quota defines the maximum number of SMS messages that can be sent within a specified time interval (such as per day, week, month, or indefinitely). Each quota is uniquely identified by a quotaId (UUID) and is reset according to the customer’s time zone. Quotas can be assigned at the country, region, or default level through a QuotaProfile. Quota can also be dynamically assigned using QuotaMapping. This maps a parentQuotaId (UUID) and a unique quotaKey (e.g., sender or user) to a specific quotaId.

A quota is set in accordance with your local support, your assigned account manager or by default if nothing is specified.

Status 106 – Quota Exceeded

An SMS message may be blocked with status code 106 ("quota exceeded") when:

- The message exceeds the defined limit for its corresponding quotaId within the current interval.
- The destination country or region has no quota assigned (i.e., is explicitly blocked with a null quota mapping in the profile).
- There is no matching quota and no default quota is defined, resulting in rejection.

In these cases, the system prevents further message processing to enforce customer or destination-based limits and avoid misuse.

Sending flash sms

This is possible by just adding the customParameter “flash.sms” with the case insensitive String values “true” or “false” within the request object. The default value for this customParameter is “false”.

Example within the object to **POST** in /sms/send/

```
{
  "source": "SERGEL",
  "destination": "+4799999999",
  "userData": "Hello world",
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": false,
  "customParameters":{
          "flash.sms":"true"
     }
 }
```

In the case of batch sendings, this value can be added within the batchSendRequest or within the SendRequestMessage. But if this customParameter is added within the batchSendRequest, then it will override its value for all the messages within this single batchSendRequest.

Example 1

Here, all the messages will be sent as flash sms, even if the flash.sms customParameter is found with the value “false”within a sendRequestMessage:

```
{
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": true,
  "deliveryReportGates": [
    "BVldZyQt"
  ],
  "customParameters":{
          "flash.sms":"true"
  },
```

```
  "sendRequestMessages": [
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination":  "+4746910822",
      "userData": "Hello world, first message",
      "refId": "wir7kkw",
     "customParameters":{
          "flash.sms":"false"
      }
    },
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination": "+4741560067",
      "userData": "Hello world, second message",
      "refId": "qts883r"
    }
  ]
}
```

Example 2

Here, the first message will be sent as a flash SMS, meanwhile the second one and the third one will be sent as normal SMS. This will work if the customParameter “flash.sms” is absent in the batchSendRequest.

```
{
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": true,
  "deliveryReportGates": [
    "BVldZyQt"
  ],
  "sendRequestMessages": [
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination":  "+4746910822",
      "userData": "Hello world, first message",
      "refId": "wir7kkw",
     "customParameters":{
          "flash.sms":"true"
      }
    },
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination": "+4741560067",
```

```
      "userData": "Hello world, second message",
      "refId": "qts883r",
     "customParameters":{
          "flash.sms":"false"
      }
    },
    {
      "source": "2333",
      "sourceTON": "SHORTNUMBER",
      "destination": "+4741560096",
      "userData": "Hello world, third message",
      "refId": "qts847r"
    }
  ]
}
```

Scheduled delivery of MT messages

Messages may be scheduled for a later delivery but at most 3 months in the future.

Add the custom parameter “scheduledTime” with the value as the date that the message should be sent. The date should be formatted according to RFC3339.

Example

```
{
  "source": "SERGEL",
  "destination": "+4799999999",
  "userData": "Hello world",
  "platformId": "0",
  "platformPartnerId": "0",
  "useDeliveryReport": false,
  "customParameters": {
    "scheduledTime":"2017-06-07T15:30:00Z"
  }
}
```

Obfuscation

To use obfuscation, you need to add one of these customParameters on the request object, depending on the need:

Parameter	Value
`obfuscate.userData`	true, this indicates that the userData should be obfuscated and the content itself will be replaced with the text OBFUSCATED.
`obfuscate`	true, this indicates that the userData should be obfuscated and the content itself will be replaced with the text OBFUSCATED. This will also replace the destination with the countryCode and the first three digits (e.g. +4512345678 will be replaced to +45123)

Please be aware that when you use the obfuscate function, you are not able to search in message logs for destination number or message content.

Obfuscation Example

Example for “obfuscate.userData”

```
"customParameters" : {
    "obfuscate.userData" : "true"
  }
```

Example for “obfuscate”

```
"customParameters" : {
    "obfuscate" : "true"
  }
```

Delivery Reports

When an MT message is delivered to a handset, or fails for any reason, you will receive a callback with a delivery report. This is required for charged messages, optional (but recommended) for free messages. If the message is multi-part message, then it will get a delivery report for each part. It can be sent in JSON, XML, or HTTP GET/POST key/value pairs. If you want to change your format or your URL, please contact Support.

Common requires that your receiver responds with a HTTP status of 200 OK to acknowledge receipt of the delivery report. For added reliance, Common can also require that your receiver responds with a certain string in the body as well; this is optional. If you want this, please contact Support and they will enable it on your Gate.

Delivery reports will be POSTed to your service from the following IPs, please make sure there is an opening in your firewall for the hosts listed in the Appendix 1 in this document.

Delivery reports contain the following fields:

Field	Data type	Description
refId	String	If you used a refId when submitting the message, this will be mirrored here. If not, this will be null.
id	String	This is Common's internal message ID for this message. It mirrors the ID which was given to you when submitting the message. If the message is a multipart message, the id will have the following pattern *{id}${n}** where *n* is the ordinal number that identifies the part. ***Example:** *abc123$0* is the first part of the message with id *abc123*
operator	String	The telecom operator the message was sent to (The end-users's operator)
sentTimestamp	DateTime	The timestamp when Common sent the message to the telecom operator. UTC time formatted according to RFC3339.
timestamp	DateTime	The timestamp from the telecom operator for this status message. UTC time formatted according to RFC3339.

resultCode	Integer	The status of the message. For what the different codes mean, see Status codes table below.
operatorResultCode	String	The unmapped status of the message from the operator. Each telecom operator has different statuses and this is only provided for debugging or reference, resultCode is the real status.
segments	Integer	The number of segments (of 140 bytes) the message was split into for delivery.
gateCustomParameters	<List>KeyValue	If there are any custom parameters set on your gate, they will be provided here. Usually blank.
customParameters	<List>KeyValue	If there are any extra fields in the delivery report Common receives from the operator, they will be listed here. Note: The parameters source and destination (as defined in the request) added by default to the customParameters.

Result Codes

The most common result code is 1001 Delivered. This code indicates a successful delivery (and payment, if charged) of the message. Most statuses are final, indicating that the message either has been successfully delivered, or failed in a non-recoverable way.

resultCode	Description	Transaction State
0	Unknown error	FINAL: NOT DELIVERED, NOT BILLED*
1	Temporary routing error	FINAL: NOT DELIVERED, NOT BILLED*
2	Permanent routing error	FINAL: NOT DELIVERED, NOT BILLED*
3	Maximum throttling exceeded	FINAL: NOT DELIVERED, NOT BILLED*
4	Timeout	FINAL: UNKNOWN DELIVERY, UNKNOWN BILLING*
5	Operator unknown error	FINAL: UNKNOWN DELIVERY, UNKNOWN BILLING*
6	Operator error	FINAL: NOT DELIVERED, NOT BILLED*
104	Configuration error	FINAL: NOT DELIVERED, NOT BILLED*
105	Internal error (internal Sergel connect error)	FINAL: NOT DELIVERED, NOT BILLED*
106	Quota Exceeded	FINAL: NOT DELIVERED, NOT BILLED*
1000	Sent (to operator)	**TEMP**: NOT DELIVERED, NOT BILLED*
1001	Billed and delivered	FINAL: DELIVERED, BILLED* (if applicable)
1002	Expired	FINAL: NOT DELIVERED, NOT BILLED*
1003	Deleted	FINAL: NOT DELIVERED, NOT BILLED*
1004	Mobile full	FINAL: NOT DELIVERED, NOT BILLED*
1005	Queued	**TEMP**: NOT DELIVERED, NOT BILLED*
1006	Not delivered	FINAL: NOT DELIVERED, NOT BILLED*
1007	Delivered, Billed delayed	**TEMP**: DELIVERED, NOT BILLED*
1008	Billed OK (charged OK before sending message)	**TEMP**: NOT DELIVERED, BILLED*

1009	Billed OK and NOT Delivered	FINAL: NOT DELIVERED, BILLED*
1010	Expired, generated by SERGEL	FINAL: UNKOWN DELIVERY, UNKNOWN BILLING*
1011	Billed OK and sent (to operator)	**TEMP**: NOT DELIVERED, BILLED*
1012	Delayed (temporary billing error, system will try to resend)	**TEMP**: NOT DELIVERED, NOT BILLED* (resending)
1013	Message sent to operator, Bill delayed	**TEMP**: NOT DELIVERED, NOT BILLED*
1014	SenderId blocked	FINAL: NOT DELIVERED, NOT BILLED*
2000	Invalid source, the specified source number or Alpha is invalid	FINAL: NOT DELIVERED, NOT BILLED*
2001	Source shortnumber not supported, the source TON may not be set to shortnumber	FINAL: NOT DELIVERED, NOT BILLED*
2002	Source alpha not supported, the source TON may not be set to alpha	FINAL: NOT DELIVERED, NOT BILLED*
2003	Source MSISDN not supported, the source TON may not be set to MSISD	FINAL: NOT DELIVERED, NOT BILLED*
2100	Destination shortnumber not supported, the destination TON may not be set to shortnumber	FINAL: NOT DELIVERED, NOT BILLED*
2101	Destination alpha not supported, the destination TON may not be set to alpha	FINAL: NOT DELIVERED, NOT BILLED*
2102	Destination MSIDN not supported, the destination TON may not be set to MSISDN	FINAL: NOT DELIVERED, NOT BILLED*
2103	Operation blocked, requested operation is not supported for the specified destination	NOT DELIVERED, NOT BILLED*
2104	Unknown subscriber	FINAL: NOT DELIVERED, NOT BILLED*
2105	Destination blocked (subscriber permanently barred)	FINAL: NOT DELIVERED, NOT BILLED*
2106	Number error	FINAL: NOT DELIVERED, NOT BILLED*
2107	Destination temporarily blocked (subscriber temporarily barred)	FINAL: NOT DELIVERED, NOT BILLED*

2108	Invalid destination	FINAL: NOT DELIVERED, NOT BILLED*
2200	Charging error	FINAL: NOT DELIVERED, NOT BILLED*
2201	Subscriber has low balance	FINAL: NOT DELIVERED, NOT BILLED*
2202	Subscriber barred for overcharged (premium) messages	FINAL: NOT DELIVERED, NOT BILLED*
2203	Subscriber too young (for this particular content)	FINAL: NOT DELIVERED, NOT BILLED*
2204	Prepaid subscriber not allowed	FINAL: NOT DELIVERED, NOT BILLED*
2205	Service rejected by subscriber	FINAL: NOT DELIVERED, NOT BILLED*
2206	Subscriber not registered in payment system	FINAL: NOT DELIVERED, NOT BILLED*
2207	Subscriber has reached max balance	FINAL: NOT DELIVERED, NOT BILLED*
3000	GSM encoding is not supported	FINAL: NOT DELIVERED, NOT BILLED*
3001	UCS2 encoding is not supported	FINAL: NOT DELIVERED, NOT BILLED*
3002	Binary encoding is not supported	FINAL: NOT DELIVERED, NOT BILLED*
4000	Delivery report is not supported	FINAL: NOT DELIVERED, NOT BILLED*
4001	Invalid message content	FINAL: NOT DELIVERED, NOT BILLED*
4002	Invalid tariff	FINAL: NOT DELIVERED, NOT BILLED*
4003	Invalid user data	FINAL: NOT DELIVERED, NOT BILLED*
4004	Invalid user data header	FINAL: NOT DELIVERED, NOT BILLED*
4005	Invalid data coding	FINAL: NOT DELIVERED, NOT BILLED*
4006	Invalid VAT	FINAL: NOT DELIVERED, NOT BILLED*
4007	Unsupported content for destination	FINAL: NOT DELIVERED, NOT BILLED*

*__BILLED__, __NOT BILLED__ & __UNKNOWN BILLING__ Transaction state indicates the state when sending premium SMS.

Delivery Report Example

The following example is an example of a successfully delivered message. refId and id have been set to invalid values in this example.


```
{
  "refId": "0",
  "id": "0",
  "operator": "no.telenor",
  "sentTimestamp": "2015-11-19T09:37:35Z",
  "timestamp": "2015-11-19T09:37:00Z",
  "resultCode": 1001,
  "operatorResultCode": "2",
  "segments": 1,
  "gateCustomParameters": {},
  "customParameters": {
    "received": "2015-11-19 10:37:36",
    "source": "2333",
    "destination": ""+4746910822"
  }
}
```


The following example is an example of a message which was attempted sent to a phone number which does not exist. refId and id have again been set to invalid values in this example.


```
{
  "refId": "0",
  "id": "0",
  "operator": null,
  "sentTimestamp": "2015-11-19T10:17:37Z",
  "timestamp": "2015-11-19T10:17:37Z",
  "resultCode": 2106,
  "operatorResultCode": null,
  "segments": 1,
  "gateCustomParameters": {},
  "customParameters": {
    "received": "2015-11-19 11:17:37",
    "source": "2333",
    "destination": ""+4746910823"
  }
}
```

Sergel
we care

Appendix 1

The following hosts are currently used for outgoing messaging.

Hostname(s)	IP address(es)
sp1.n-eu.linkmobility.io	193.142.108.11
sp2.n-eu.linkmobility.io	193.142.108.12
sp3.n-eu.linkmobility.io	193.142.108.13
sp4.n-eu.linkmobility.io	193.142.108.14
sp5.n-eu.linkmobility.io	193.142.108.15
sp6.n-eu.linkmobility.io	193.142.108.16
sp1.c-eu.linkmobility.io	37.202.24.11
sp2.c-eu.linkmobility.io	37.202.24.12
sp3.c-eu.linkmobility.io	37.202.24.13
sp4.c-eu.linkmobility.io	37.202.24.14
sp5.c-eu.linkmobility.io	37.202.24.15
sp6.c-eu.linkmobility.io	37.202.24.16
sp1.s-eu.linkmobility.io	185.73.14.11
sp2.s-eu.linkmobility.io	185.73.14.12
sp3.s-eu.linkmobility.io	185.73.14.13
sp4.s-eu.linkmobility.io	185.73.14.14
sp5.s-eu.linkmobility.io	185.73.14.15
sp6.s-eu.linkmobility.io	185.73.14.16
socks1.sp247.net	195.84.162.34
socks2.sp247.net	194.71.165.71
socks3.sp247.net	195.84.162.16
socks4.sp247.net	194.71.165.98
socks5.sp247.net	195.84.162.3
socks6.sp247.net	194.71.165.122
s1.n-eu.linkmobility.io	213.242.87.36
s2.n-eu.linkmobility.io	213.242.87.37
s3.n-eu.linkmobility.io	213.242.87.38
s4.n-eu.linkmobility.io	213.242.87.39
s5.n-eu.linkmobility.io	213.242.87.40
s6.n-eu.linkmobility.io	213.242.87.41
s1.c-eu.linkmobility.io	62.67.62.101
s2.c-eu.linkmobility.io	62.67.62.102
s3.c-eu.linkmobility.io	62.67.62.103
s4.c-eu.linkmobility.io	62.67.62.104
s5.c-eu.linkmobility.io	62.67.62.105
s6.c-eu.linkmobility.io	62.67.62.106
s1.s-eu.linkmobility.io	217.163.95.196
s2.s-eu.linkmobility.io	217.163.95.197
s3.s-eu.linkmobility.io	217.163.95.198
s4.s-eu.linkmobility.io	217.163.95.199
s5.s-eu.linkmobility.io	217.163.95.200

s6.s-eu.linkmobility.io	217.163.95.201
~~**s1.no.linkmobility.io**~~	213.242.87.68 (decommissioned 2024-08-31)
~~**s2.no.linkmobility.io**~~	213.242.87.69 (decommissioned 2024-08-31)
~~**s3.no.linkmobility.io**~~	213.242.87.70 (decommissioned 2024-08-31)
~~**s4.no.linkmobility.io**~~	213.242.87.71 (decommissioned 2024-08-31)
~~**s5.no.linkmobility.io**~~	213.242.87.72 (decommissioned 2024-08-31)
~~**s6.no.linkmobility.io**~~	213.242.87.73 (decommissioned 2024-08-31)
~~**s1.deb.linkmobility.io**~~	62.67.62.68 (decommissioned 2023-08-31)
~~**s2.deb.linkmobility.io**~~	62.67.62.69 (decommissioned 2023-08-31)
~~**s3.deb.linkmobility.io**~~	62.67.62.70 (decommissioned 2023-08-31)
~~**s4.deb.linkmobility.io**~~	62.67.62.71 (decommissioned 2023-08-31)
~~**s5.deb.linkmobility.io**~~	62.67.62.72 (decommissioned 2023-08-31)
~~**s6.deb.linkmobility.io**~~	62.67.62.73 (decommissioned 2023-08-31)

Appendix 2

Silent Billing

To perform a Silent Billing (billing the end-user without them receiving a text message on their phone) set the customParameter "chargeOnly" to "true" (the string "true", not the Boolean true). Silent billing is only available in certain markets and is bound by additional agreements and restrictions. Your sales associate or Support will advise you if you are in doubt.

Examples

The following example shows how to send a premium (charged) message. The following message would cost 1 NOK for the end-user to receive. It is sent from Norwegian shortcode 2333 to Norwegian phonenumber 41560067 (country code +47). The delivery report is delivered to a predetermined gateId. (Delivery reports are required for charged messages. Only TON “SHORTNUMBER” is accepted for charged messages.)

```
{
  "source": "2333",
  "sourceTON": "SHORTNUMBER",
  "destination": "+4741560067",
  "userData": "This message costs 1 NOK to receive.",
  "tariff": 100,
  "currency": "NOK",
  "platformId": "0",
  "platformPartnerId": "0",
  "refId": "9ui5kKL",
  "productDescription": "Informational message from 2333",
  "productCategory": 15,
  "useDeliveryReport": true,
  "deliveryReportGates": [
    "0"
  ]
}
```

Norway (Strex) only

You must also set the customParameter "authorize" to "true", and "strex.username" to your company's Strex MerchantID. On the FIRST time you bill an end-user silently, you must also set the customParameter "strex.securityLevel" to 2. On the second and subsequent requests, this parameter should not be present.
If you want to send a MT message without charging, then you should have the following values: *tariff = NOK0* and *chargeOnly = true*.

Appendix 3

Supported TLS versions

To ensure the highest level of security and performance, TLS 1.3 is strongly recommended for all connections to the API. TLS 1.3 offers several advantages over previous versions, including:

- Improved Performance: Faster handshake process, reducing connection latency.
- Stronger Security: Removal of obsolete and vulnerable cryptographic algorithms (e.g., SHA-1, RC4, and static RSA).
- Forward Secrecy: Enhanced protection of session keys, preventing decryption even if the server's private key is compromised.
- Simplified Protocol: Reduced complexity leads to fewer implementation errors and better maintainability.

Although TLS 1.2 is still supported for backward compatibility, it is considered legacy. Clients and servers should be updated to use TLS 1.3 wherever possible.

HTTP is deprecated and LINK **strongly recommend** using HTTPS if HTTP is being used today.

Supported Ciphers

TLS	**Ciphers**
1.3	TLS_AES_128_GCM_SHA256 (0x1301) TLS_AES_256_GCM_SHA384 (0x1302) TLS_CHACHA20_POLY1305_SHA256 (0x1303)
1.2	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f) TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030) TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8) Support for the following ciphers below is removed 2025-10-15: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x9e) TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x9f) TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028) TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027) TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013) TLS_RSA_WITH_AES_256_CBC_SHA256 (0x3d) TLS_RSA_WITH_AES_128_CBC_SHA256 (0x3c)