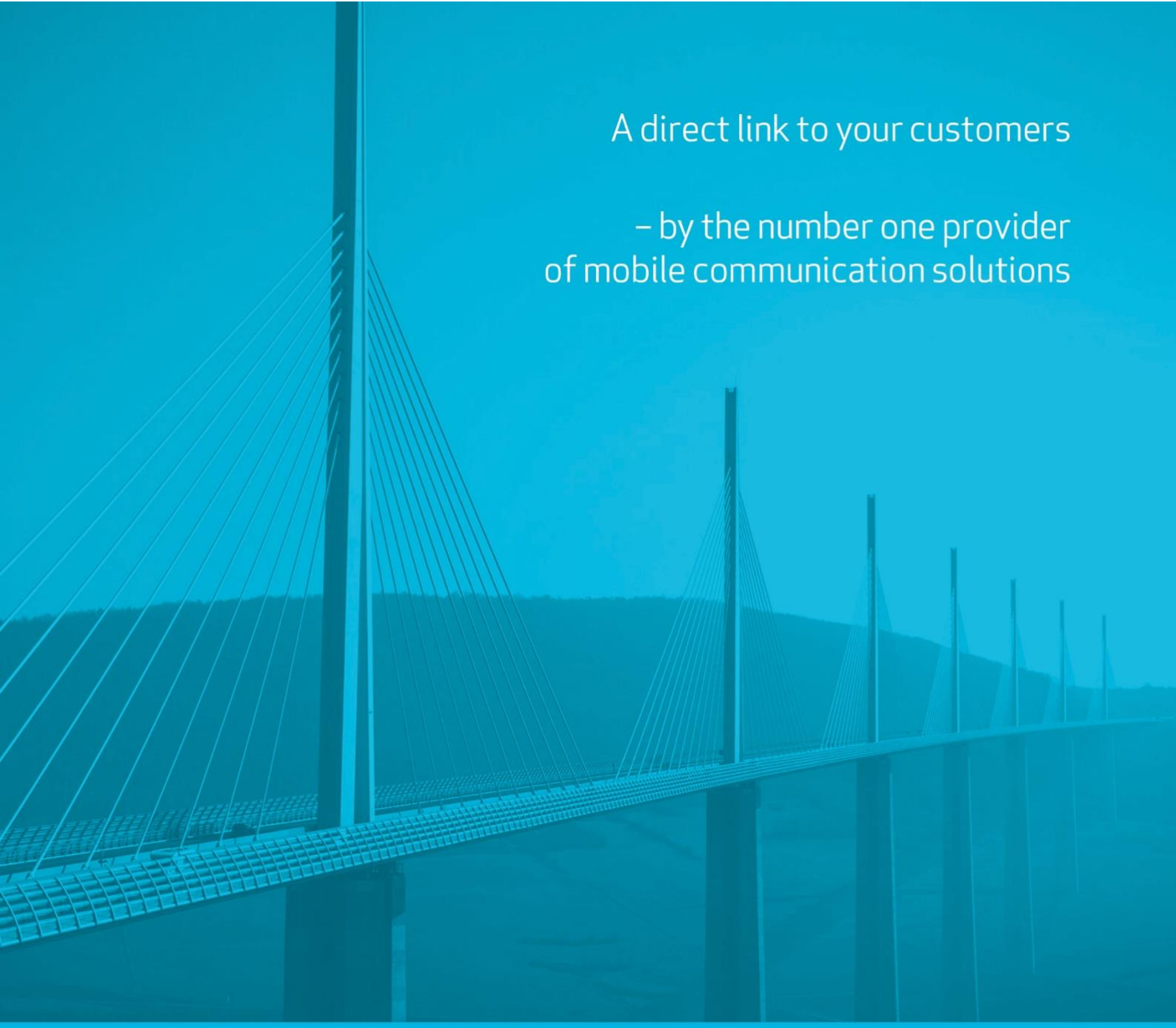

LINK Mobility
Common Platform API Description

Gate API

Version 2.1; Last updated March 14, 2024

For help, see the following link https://linkmobility.com/support/
The most up-to-date version of this document is available at
https://www.linkmobility.com/developer/

link mobility

Contents

Before you begin

To use this API, you will need a **username**, **password**, **platformId** and **platformPartnerId**. These will be provided to you by support. If necessary, make an opening in your firewall so that common can connect to your system. The list of addresses for common will be find in the appendix.

Scope of this document

This document will describe in detail the methods within the API for creating, retrieving, updating and deleting gates towards the Common API.

Terms and glossary

Gate

A Gate, or gateway, is an endpoint used to deliver MO, MT DLR, Address lookup and URL shortener callbacks, as well as the asynchronous callbacks from other services, that are described in their respective documents.

ErrorResponse
A json object that contains an error code, a description and its translation.

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 message, optional for bulk.

MoMessage
Or 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).

Base URL:s

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

https://eu.linkmobility.io/gate (automatic failover)
https://n-eu.linkmobility.io/gate
https://c-eu.linkmobility.io/gate
https://s-eu.linkmobility.io/gate
~~**https://no.linkmobility.io/gate**~~ **(will be decommissioned after 2024-08-31)**
~~**https://deb.linkmobility.io/gate**~~ **(decommissioned 2023-08-31)**
https://wsx.sp247.net/gate (legacy)

Authentication

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

OAuth 2.0

The preferred way 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":"xxxxxxx","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 xxxxxx
```

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 a 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	No resource	Resource exist	No access to resource	Invalid Request
GET	404 (Not found) **Return:** ErrorResponse	200 (OK) **Return:** Correct Resource	403 (Forbidden) **Return:** ErrorResponse	400 (Bad Request) **Return:** ErrorResponse
POST	201 (Created) **Return http header:** *location: {resource location}* **Return:** empty body	409 (Conflict) **Return:** ErrorResponse	403 (Forbidden) **Return:** ErrorResponse	400 (Bad Request) **Return:** ErrorResponse
PUT	405 (Method not Allowed) **Return:** ErrorResponse	204 (OK – No content) **Return:** ErrorResponse	403 (Forbidden) **Return:** ErrorResponse	400 (Bad Request) **Return:** ErrorResponse
DELETE	405 (Method not Allowed) **Return:** ErrorResponse	204 (OK – No content) **Return:** ErrorResponse	403 (Forbidden) **Return:** ErrorResponse	400 (Bad Request) **Return:** ErrorResponse

Methods

Add a PartnerGate

Adds a platform gate to the storage by posting a platformGate object.

HTTP Method	POST	
URI	`/partnergate`	
Consumable Resource	A partnerGate object without the Id.	
Response http headers	`Location`	Contains the URI to use to retrieve the recently added gate.

Get a PartnerGate using Id

Retrieves a PartnerGate by its ID.

HTTP Method	GET	
URI	/partnergate/platform/*{platformId}*/partner/*{partnerId}*/id/*{gateId}*	
Path Parameters	*platformId*	The Id of the platform
	partnerId	The Id of the platformPartner
	gateId	The Id of the PlatformGate

Get a PartnerGate using refId

Retrieves a platform gate by its refId. It can be used only in a platform level.

HTTP Method	GET	
URI	/partnergate/platform/*{platformId}*/partner/{partnerId}/refid/*{refId}*	
Path Parameters	*platformId*	The Id of the platform
	partnerId	The Id of the platformPartner
	refId	The refId of the platformGate

Update a PartnerGate

Updates a PartnerGate in the storage by putting a PlatformGate object.

HTTP Method	PUT	
URI	/partnergate/platform/*{platformId}*/partner/{partnerId}/id/*{gateId}*	
Path Parameters	*platformId*	The Id of the platform
	partnerId	The Id of the platformPartner
	gateId	The Id of the PlatformGate
Consumable Resource	A PartnerGate including the id in the object.	

List PartnerGates

Retrieves a list of all the PartnerGates under a single platformPartner.

HTTP Method	GET

URI	/partnergate/platform/*{platformId}*/partner/{partnerId}	
Path Parameters	*platformId*	The Id of the platform
	partnerId	The Id of the platformPartner

Remove a PartnerGate

Deletes the PartnerGate from the storage.

HTTP Method	DELETE	
URI	/partnergate/platform/*{platformId}*/partner/{partnerId}/id/*{gateId}*	
Path Parameters	*platformId*	The Id of the platform
	partnerId	The Id of the platformPartner
	gateId	The Id of the PlatformGate

Data types

PartnerGate

This object describes a gate that has a platform partner level.

Parameter	Data type	Description
id	String	**[AUTOGENERATED]** The unique id for the gate. This is generated when the gate is created.
refId	String	A reference Id that can be given by the user.
ttl	Long	This is the time to live for the requests done towards this certain gate in milliseconds. The default value is 48 hours in milliseconds.
acknowledge	Boolean	False if any 2xx HTTP status code will be treated as a successful request. True otherwise if some specific object in the body is expected with the HTTP 2xx response. If this is true and no body is found within the request, then the request will be done again until the response returns HTTP 2xx and an object within the response.
throttle	Integer	**[OPTIONAL]** The maximum requests that can be done per second.

destinations	List <Destination>	One or more endpoints that will be triggered when the gate is on use. *(See Destination)*
type	String	The value should be **PARTNER_GATE**.
gateType	String	Only a couple of Strings are allowed as values here: **ALL:** This determines if the object* will be sent through all the destinations given within this single gate. **ONE_ORDERED:** This determines if the object* will be sent to one destination, chosen sequentially. **ONE_RANDOM:** this determines if the object* will be sent to a random destination picked up from the list of destinations. *The object can be a certain DLR or MoMessage or other object that can be forwarded through the gates
platformId	String	The platform Id provided by support
platformPartnerId	String	The platform partner id provided by support
customParameters	KeyValue	**[ADVANCED]** Additional parameters may be specified if needed. Support will advise you if you need to use custom parameters.

Example:

```
{
    "type": "PARTNER_GATE",
    "id": "abc123E",
    "refId": "TEST",
    "platformId": "MY_PLATFORM",
    "platformPartnerId": "MY_PARTNER",
    "destinations": [
        {
            "url": "https://endpoint.example.com/receive",
            "contentType": "application/json",
            "username": "",
            "password": "",
            "customParameters": {}
        }
    ],
    "gateType": "ONE_ORDERED",
    "ttl": 172800000,
    "acknowledge": false,
    "throttle": null,
    "customParameters": {}
}
```

Destination

The destination describes specifically how and where to send the object. This object contains the following parameters:

Parameter	Data type	Description
url	URL	The actual URL to do a request to. The following protocols are supported within the URL: 1. HTTP 2. HTTPS The objects are sent through these protocols using the HTTP method POST as default. To change this, add the properly custom Parameter as described within the *customParameter* description. The following URL can be given for dropping the incoming object: *do://drop*
contentType	ContentType	A single MIME type that determines the format of the object to be sent. The following MIME types are supported: 1. application/json 2. application/xml 3. application/x-www-form-urlencoded
username	String	**[OPTIONAL]** Used only if the destination requires basic authentication. Only basic authentication is supported.
password	String	**[OPTIONAL]** Used only if the destination requires basic authentication. Only basic authentication is supported.
customParameters	KeyValue	**[ADVANCED]** Additional parameters may be specified if needed. Some helpful custom parameters within a single destination are the following: 1. **method**: This is added to change the default HTTP method. Only the following methods are allowed as values: **GET** and **PUT**. 2. **template**: see *Destination templates.* Support will advise you if you need to use custom parameters.

Destination templates

The templates are used to extract specific values from an incoming object and send them as URL parameters. This template will setup query parameters into the URL with the injected values from the incoming objects.

Incoming objects

There are two incoming objects that has the following extractable values:

DeliveryReport

This object comes in when the gate is supposed to handle delivery reports.

Parameter	Data type	Description
id	String	The messageId of the message
operator	String	The operator that handled the message
sentTimestamp	Date	The time that the message was sent
timestamp	Date	The time that the delivery report was handled
resultCode	String	The final result code of the message
operatorResultCode	String	The final result code from the operator
destination	Number (MSISDN)	The destination number in msisdn format
source	Number (Shortnumber)	The source in alphanumeric format. It can be as msisdn, shortnumber or alphanumeric.

SmsMessage

This object comes in when the gate is connected to a Keyword and it is supposed to be delivered to a certain point if the message goes to the route declared.

Parameter	Data type	Description
messageId	String	The messageId of the message.
destination	Number (MSISDN)	The destination number in msisdn format.
source	Number (Shortnumber)	The source in alphanumeric format. It can be as msisdn, shortnumber or alphanumeric.
userData	String	The content of the message itself.

Basic Setup

The template format should have the following pattern as value within the custom parameter with the key *template*:

```
key1=${methodName1}&key2={methodName2} or key1=methodName1&key2=methodName2
```

where *methodName1* and *methodName2* corresponds to methods like *getMethodName1()* and *getMethodName2()* in the incoming object. The following example extracts the *messageId* and the *resultCode* from a *deliveryReport* object:

Example:

Destination for Delivery reports handling:

```
{
	"url": "https://endpoint.example.com/receive",
	"contentType": "application/json",
	"username": "",
	"password": "",
	"customParameters": {
		"template": "id=${id}&resultCode=${resultCode}"
	}
}
```

Result URL:

```
https://endpoint.example.com/receive?id=MyIde123456&resultCode=1001
```

Template with dates

It is possible to manipulate the format of a date to send it through the URL as query parameter. To achieve that, the format of the value can be given by inserting the **date** key right after the parameter's name separated with the character '|' with the following pattern:

```
"template": "time=${parameter|date|yyyy-MM-dd}"
```

The legal input is a Long/long containing the number of milliseconds counting from January 1st of 1970 or a String on a format *yyyyMMddhhMMss.*

Example:

Destination for Delivery reports handling:

```
{
	"url": "https://endpoint.example.com/receive",
	"contentType": "application/json",
	"username": "",
	"password": "",
	"customParameters": {
		"template": "time=${timestamp|date|yyyy-MM-dd}"
```

```
        }
}
```

Result URL:

```
https://endpoint.example.com/receive?time=2019-12-05
```

Template with numbers

The format of the number can be manipulated as well. To achieve this, you can add the special formats with the same pattern used for dates. For formatting the numbers, the correct pattern should look like this:

```
"template": "time=${parameter|number|keyword_for_formatting}"
```

The following keywords for formatting this data type should be used:

***nocountry*:** Used for shortnumber. This means that the country within the shortcode will be skipped. So instead of SE-1234, the result will be only 1234.

Example:

Destination for Delivery reports handling:

```
{
        "url": "https://endpoint.example.com/receive",
        "contentType": "application/json",
        "username": "",
        "password": "",
        "customParameters": {
                "template": "source=${source|number|nocountry}"
        }
}
```

Result URL:

```
https://endpoint.example.com/receive?source=1234
```

***noplus*:** Used for msisdn. This means that the plus sign will be skipped within the number itself, so +46123456789 will become 46123456789.

Example:

Destination for Delivery reports handling:

```
{
        "url": "https://endpoint.example.com/receive",
        "contentType": "application/json",
        "username": "",
        "password": "",
        "customParameters": {
                "template": "destination=${destination|number|noplus}"
        }
}
```

Result URL:

```
https://endpoint.example.com/receive?destination=46123456789
```

Error Codes

Error Code	Message
103000	UNKNOWN_ERROR
103050	STORAGE_ERROR
103100	INVALID_AUTHENTICATION
103101	ACCESS_DENIED
103201	INVALID_PLATFORM_ID
103202	INVALID_PARTNER_ID
103211	INVALID_REF_ID
103212	DUPLICATE_REF_ID
103290	INVALID_FORWARDING_OBJECT
103300	MISSING_PARAMETERS
103301	INVALID_ID
103302	INVALID_GATE
103303	NO_GATES_FOUND
103304	GATE_NOT_FOUND
103400	ILLEGAL_RETURN_TYPE

Appendix 1

The following hosts are currently used for outgoing messaging.

Hostname(s)	IP address(es)
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 (will be decommissioned after 2024-08-31)
~~**s2.no.linkmobility.io**~~	213.242.87.69 (will be decommissioned after 2024-08-31)
~~**s3.no.linkmobility.io**~~	213.242.87.70 (will be decommissioned after 2024-08-31)
~~**s4.no.linkmobility.io**~~	213.242.87.71 (will be decommissioned after 2024-08-31)
~~**s5.no.linkmobility.io**~~	213.242.87.72 (will be decommissioned after 2024-08-31)
~~**s6.no.linkmobility.io**~~	213.242.87.73 (will be decommissioned after 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

Supported TLS versions

From 2020-11-15 will TLS 1.2 or higher be required for all HTTPS connections.

Support for TLS 1.0 and 1.1 will be discontinued. Versions 1.0 and 1.1 of TLS are older protocols that have been deprecated and are considered as security risks in the Internet community.

LINK strongly recommend to use HTTPS if HTTP is being used today. HTTP is deprecated as of 2020-09-01 by LINK, and will be removed in the future. Date for HTTP removal is not yet decided.

Changelog of this document

Date	Version	Author	Changes
2016-03-30	1.0	BMS	Initial version
2019-12-27	1.1	EP	More description about the API in a platform partner level
20202-08-28	1.2	JS	Added Appendix 2 about TLS
2021-02-19	1.3	TL	The URL for Developers, From HTTP to HTTPS
2021-03-12	1.4	EP	Correction on the

			implementation of the “update PartnerGate” method.
2021-04-29	1.5	TL	Changed to new homepage URL
2023-06-23	1.6	FS	Updated some IPs and URLs as legacy
2023-10-12	2.0	KCN	Added OAuth 2.0 authentication. Added eu.linkmobility.io endpoint.
2024-03-14	2.1	FS	Updated some IPs and URLs as legacy