SEEBURGER

Communication Gateway: Messaging (Version 1.0) Documentation

Table of content

About

This document describes how to use the **Communication Gateway: Messaging** API.

You can use the **Communication Gateway: Messaging** API to exchange messages with your trading partners. The API enables you to send and receive messages without providing your own callback URL.

Instead of pushing messages to your system, the **Communication Gateway: Messaging** API requires you to poll messages from the service.

In addition to sending and retrieving messages, the API allows you to:

- Check the status of transmitted messages
- Acknowledge the download of messages

Prerequisites

Organizational requirements

To use the **Communication Gateway: Messaging** API, ensure that you meet the following organizational requirements:

- Create a user account in SEEBURGER Customer Portal: https://portal.seeburger.cloud
- Book a service that includes access to the **Communication Gateway: Messaging** API (for example, B2B/EDI Communication Services)
- Retrieve the API specification file from the API Catalog in the SEEBURGER Customer Portal. The API specification provides all relevant connection information of all endpoints for the **Communication Gateway: Messaging** API, including the basepath, URIs and other information.

Technical requirements

To connect to the **Communication Gateway: Messaging** API, ensure that your system meets the following technical requirements:

- Use Cloudlink Type: Messaging API
- Set up MBR Agreement(s) with your trading partner(s) (sender <-> receiver combinations)
- Use JSON format for both request and response messages
- Access the OpenAPI definition for the **Communication Gateway: Messaging** API in the API Catalog after you log in to the SEEBURGER Customer Portal
- Authenticate requests using an API key in the HTTP header:
    - **Header**: Authorization
    - **Value**: API-Key “YOUR_API_KEY”

API resource overview

You can use the **Communication Gateway: Messaging** API for two main purposes:

- Send messages to your communication partner and check their status
- Receive messages from your communication partner, download them, and acknowledge receipt

The **Communication Gateway: Messaging** API provides the following resources:

- Send messages and receive status information:
    - POST **/cloud/messaging/1.0/message**
      Use this endpoint to send messages to your communication partner.
    - GET **/cloud/messaging/1.0/status/{id}**
      Use this endpoint to retrieve the status of a dedicated message.
- Receive messages:
    - **GET /cloud/messaging/1.0/message-list**
      Use this endpoint to retrieve a list of messages that are available for download.
    - **GET /cloud/messaging/1.0/message/{id}**
      Use this endpoint to download a dedicated message.
      You can download a message multiple, even after you acknowledge it. As long as the system has not reorganized (deleted) the message, it remains available.
    - **PUT /cloud/messaging/1.0/acknowledgement/{id}**
      Use this endpoint to acknowledge that you have downloaded the message. After you send the acknowledgment, the system can delete the message from the cloud.

The Server URL is **https://api.seeburger.cloud:443**.

To retrieve the complete API specification file, follow the instructions from the chapter “Prerequisites – Organizational requirements”.

API flow overview

Send messages

To send messages using the **Communication Gateway: Messaging** API, perform a “POST” request.

Encode your message in Base64 and include it in a JSON envelope. After you send the request, the **Communication Gateway: Messaging** API returns a JSON response that contains a GMID (Global Message ID).

Use this GMID to retrieve the status of your message.

To check the status, send a GET request. The **Communication Gateway: Messaging** API returns the message status in JSON format.

Receive messages

To receive messages, interact with the **Communication Gateway: Messaging** API using multiple GET requests.

First, retrieve a list of available messages:

- The API returns the message list as a JSON file.
- Each message includes a unique GMID.

To download a message:

- Send a GET request and provide the GMID as a parameter.
- The API returns the message as a Base64-encoded payload within a JSON envelope.

The Cloud stores messages until you acknowledge them or until they expire:

- If you do not acknowledge a message, it expires automatically.
- When a message expires, the original sender receives a notification that the message expired and might not have been delivered.

You can download messages multiple times, even after you acknowledge them, as long as the system has not yet deleted them. Ensure that your system can handle duplicate messages.

To acknowledge a message:

- Send a PUT request to confirm that you have successfully downloaded the message.

  Alternatively, you can download a message with automatic acknowledgment, which removes the need for a separate acknowledgment request.

However, consider that if processing fails on your side and you need to restart, the message might no longer appear in the message list. The message list endpoint can optionally include already acknowledged messages, which helps you handle such scenarios.

Implementation

Resource POST /cloud/messaging/1.0/message

Use the **/cloud/messaging/1.0/message** endpoint of the **Communication Gateway: Messaging** API to send messages to your communication partner.

Provide the request in JSON format. Encode the message payload in Base64 and include it in the “Attachments” section.

You can provide the following fields either in the HTTP header or in the JSON message body.

Field name	Description
receiverid	Mandatory (**): Specifies the SEEID of your communication partner on the SEEBURGER MBR Gateway.
txid	Mandatory (*): Specifies the TransactionID used to uniquely identify a message. The system uses this for duplicate detection.
subject	Optional (*): Specifies a subject that the system forwards to your communication partner, if the receiving protocol supports it.
filename	Optional (*): Specifies the filename that the system forwards to your communication partner, if the receiving protocol supports it.
ttl	Optional (*): Specifies the time to live (TTL) of the message in minutes for automatic expiration.
priority	Optional (*): Currently not used. Supported values are: • 0 / EXTRA • 1 / HIGH • 2 / MEDIUM / STANDARD • 3 / LOW
routingtype	Optional (*): Specifies the routing type. Supported values are: • cbr • ids

	• ip • mbr • peppol • peppol-test • ssp • van If you do not provide a value, the API uses “mbr” as the default. **Note**: the routing type “echo” is not available in this field. To test the connection, you must configure this in the B2B directory.
vanalias	Mandatory (**): This “van” alias identifies the recipient subsidiary. This field is required when you use **routingtype**:**van** but is not used for any other routing types.

Legend (*):

- Mandatory** (conditional):
  - receiverid
    - Required for routing type mbr.
    - For cbr, the system reads the receiver from the message (for example, EDIFACT or ANSI X12)
    - For peppol / peppol-test, the system reads the receiver from the SBDH
    - For ip, ids, and ssp, the system uses predefined internal receivers
    - For van, the system determines the receiver based on vanalias
  - vanalias
    - Required when routingtype = van
    - Identifies the recipient subsidiary
- Mandatory*:
  This field is required. You can include it either in the HTTP header or in the JSON body. If you provide it in both places, the **Communication Gateway: Messaging API** uses the value from the JSON body. Invalid values are ignored or replaced with default values.

- Optional:
  You can provide this field in the HTTP header or JSON body. If you provide it in both places, the API uses the value from the JSON body. Invalid values are ignored or replaced with default values.

Attachments

The envelope may contain a maximum of one attachment. The following fields are available:

Field name	Description
name	Optional: Currently not used.
data	Mandatory: Contains the message payload encoded in **Base64**.

Response

The **Communication Gateway: Messaging** API returns a JSON response with the following envelope fields:

Field name	Description
status	HTTP status code: • 200 OK • 400 Bad Request • 401 Unauthorized • 403 Forbidden
statustext	Possible return values: • COMPONENT_ERROR • COMPONENT_ERROR_CONTENTEXTRACTOR • CONFIGURATION_ERROR • DATA_ERROR • DELETED • ERROR, EXPIRED • MDN_ERROR

	• MESSAGE_QUEUED • PENDING • RECALLED • READY_FOR_DOWNLOAD • SUCCESS • SUCCESS_DOWNLOADED • TERMINATED • SUCCESS_POLLQUEUE • ECHO-ERROR • BLOCKED
gmid	Global Message ID generated by the SEEBURGER Cloud. Use this ID to query the message status.

1.1.1. Example Requests

```
{
"message": {
   "envelope": {
      "subject": "mySubject",
      "filename": "filename.txt",
      "receiverid":"SEEGWE312345678",
      "txid": "1234567890",
      "routingtype": "mbr"
   },
   "attachments": [
      {
         "name": "Payload",
```

"data":
"VGhpcyBpcyBhIHNhbXBsZSBJTlZPSUNFIHdoaWNoIHdpbGwgYmUgQmFzZSA2NCBFbmNvZGVk"
}
]
}
}

1.1.2. Example response

{
"envelope": {
"status": 200,
"statustext": "SUCCESS",
"gmid": "33c91b10-07ec-11f1-b1c3-216c0a0e4811"
}
}

Resource GET /cloud/messaging/1.0/status/{id}

Use the GET **/cloud/messaging/1.0/status/{id}** endpoint of the **Communication Gateway: Messaging** API to retrieve the status of a dedicated message.

To request the status of a message, send a GET request and provide the GlobalMessageID (GMID) as the **{id}** path parameter.

The **Communication Gateway: Messaging** API returns a response in JSON format with the following envelope fields:

Field name	Description
status	HTTP Status with status code: • 200 OK • 400 Bad Request • 401 Unauthorized • 403 Forbidden
statustext	Processing status: • COMPONENT_ERROR • COMPONENT_ERROR_CONTENTEXTRACTOR • CONFIGURATION_ERROR • DATA_ERRO • DELETED • ERROR • EXPIRED • MDN_ERROR • MESSAGE_QUEUED • PENDING • RECALLED • READY_FOR_DOWNLOAD • SUCCESS

	• SUCCESS_DOWNLOADED • TERMINATED • SUCCESS_POLLQUEUE • ECHO-ERROR • BLOCKED • No data found
gmid	Global Message ID generated by the SEEBURGER Cloud

1.1.3. Example Requests

GET /cloud/messaging/1.0/status/33c91b10-07ec-11f1-b1c3-216c0a0e4811

1.1.4. Example response


```
{
	"envelope": {
		"status": 200,
		"statustext": "SUCCESS",
		"gmid": "33c91b10-07ec-11f1-b1c3-216c0a0e4811"
	}
}
```

Resource GET /cloud/messaging/1.0/message-list

Use the GET **/cloud/messaging/1.0/message-list** endpoint of the **Communication Gateway: Messaging** API to retrieve a list of messages that are available for download.

To request the list of available messages, perform a GET request to resource **/cloud/messaging/1.0/message-list**.

You can optionally filter the results using the following query parameters:

Parameter name	Description
includeacknowledged	Specifies whether to include already acknowledged messages. Possible values: • true • false Default value: false
routingtype	Filters messages by routing type. Supported values are: • cbr • ids • ip • mbr • peppol • peppol-test • ssp • van • echo
maxentries	Limits the number of returned entries. Provide a positive integer that does not exceed the default limit. If you provide an invalid value, the API uses the default.
sendername	Filters messages by a specific sender name.

Response

The **Communication Gateway: Messaging** API returns a response in JSON format with the following envelope fields:

Field name	Description
status	Response status (OK)
success	Indicates successful processing (OK)
timestamp	Date and time when the response message was created. Format YYYY-MM-DDThh:mm:zzz **Example**: 2026-02-12T11:25:28.205Z[UTC]
data.itemsstate	
totalentries	Total number of messages in the result
data.messages	Array of message envelopes (0..n)

Each message envelope contains:

Field name	Description
gmid	Global Message ID generated by the SEEBURGER Cloud
sladeliverytime	Delivery timestamp (Unix time)
expirydate	Expiration timestamp (Unix time)
subject	Subject provided by your communication partner (if available)
filename	Filename provided by your communication partner (if available)
sendername	Name of your communication partner
senderid	The individual SEEID of your communication partner on the SEEBURGER MBR Gateway

routingtype	The routing type used for the message. Possible values are: • cbr • ids • ip • mbr • peppol • peppol-test • ssp • van • echo If not provided the API uses “mbr” as the default.

1.1.5. Example Requests

GET /cloud/messaging/1.0/message-list

1.1.6. Example response


```
{
  "status": "OK",
  "success": "OK",
  "timestamp": "2026-02-12T11:25:28.205Z[UTC]",
  "data":   {
    "itemsstate": {"totalentries": "5"},
    "messages":      [
      {"envelope":        {
        "gmid": "6d9bc5e0-0805-11f1-b1c3-216c0a0e4811",
        "sladeliverytime": "1770895491028",
        "expirydate": "1771241084539",
        "subject": "subject1",
```

```
        "filename": "filename1.txt",
        "sendername": " SEEGWE312345678",
        "senderid": " SEEGWE312345678",
       "routingtype": "mbr"
      }},
      {"envelope":        {
        "gmid": "811790e0-0805-11f1-b1c3-216c0a0e4811",
        "sladeliverytime": "1770895519088",
        "expirydate": "1771241117094",
        "subject": "subject2",
        "filename": "filename2.txt",
        "sendername": " SEEGWE312345678",
        "senderid": " SEEGWE312345678",
        "routingtype": "mbr"
      }}
    ]
  }
}
```

Resource GET /cloud/messaging/1.0/message/{id}

Use the **/cloud/messaging/1.0/message/{id}** endpoint of the **Communication Gateway: Messaging** API to download a dedicated message.

To request a message, you need the GMID (Global Message ID). You can retrieve the GMID by first calling the GET **/cloud/messaging/1.0/message-list endpoint**.

Then send a GET request to resource **/cloud/messaging/1.0/message/{id}** where **{id}** is the GMID of the message you want to download.

You can download a message multiple times, even after you acknowledge it, as long as the system has not yet deleted it. Ensure that your system can handle duplicate messages.

In addition to the API key, you can provide the following HTTP query parameter:

Field name	Description
autoacknowledge	Optional. Specifies whether the **Communication Gateway: Messaging** API should automatically mark the message as downloaded. Possible values are: • true • false If the parameter is missing or set to “false”, you must perform a dedicated acknowledge REST request. **Note**: If you use autoacknowledge=true, the system marks the message as successfully received as soon as you download it. If processing later fails on your side, the message may no longer appear as unacknowledged. For this reason, acknowledge messages only after your processing has completed successfully.

Response

The **Communication Gateway: Messaging** API returns a response in JSON format with the following envelope fields:

Field name	Description

message.envelope	
subject	Subject of the message, if the receiving protocol supports a subject
filename	Filename of the message, if the receiving protocol supports a filename
senderid	The individual SEEID of your communication partner on the SEEBURGER MBR Gateway
receiverid	Your individual SEEID on the SEEBURGER MBR Gateway
routingtype	The routing type used for the message. Possible values are: • cbr • ids • ip • mbr • peppol • peppol-test • ssp • van • echo
message.attachments	
name	Filename of the attachment, if the receiving protocol supports a filename
data	This is the message that was sent to you encoded in Base64.

1.1.7. Example Requests

GET /cloud/messaging/1.0/status/33c91b10-07ec-11f1-b1c3-216c0a0e4811

1.1.8. Example response

```
{"message": {
  "envelope":   {
    "subject": "subject",
    "filename": "filename.txt",
    "senderid": " SEEGWE312345678",
    "receiverid": " SEEGWE312345678",
    "routingtype": "mbr"
  },
  "attachments": [   {
    "name": "filename.txt",
    "data":
"VGhpcyBpcyBhIHNhbXBsZSBJTlZPSUNFIHdoaWNoIHdpbGwgYmUgQmFzZSA2NCBFbmNvZGVk"
  }]
}}
```

Resource PUT /cloud/messaging/1.0/acknowledgement/{id}

Use the PUT **/cloud/messaging/1.0/acknowledgement/{id}** endpoint of the **Communication Gateway: Messaging** API to acknowledge that you have successfully downloaded a dedicated message.

To acknowledge a message:

1. Retrieve a list of all available messages using GET **/cloud/messaging/1.0/message-list**.
2. Identify the message by its GMID (Global Message ID).
3. Download the message using GET **/cloud/messaging/1.0/message/{id}**.
4. Send a PUT request to **/cloud/messaging/1.0/acknowledgement/{id}** where **{id}** is the GMID.

After you acknowledge a message, the system can delete it from the cloud.

Response

The **Communication Gateway: Messaging** API returns only HTTP status code:

- 202 Acknowledged – The system successfully acknowledged the message
- 404 Message not found – The system could not find the message

1.1.9. Example Requests

PUT /cloud/messaging/1.0/acknowledgement/33c91b10-07ec-11f1-b1c3-216c0a0e4811

1.1.10. Example Response

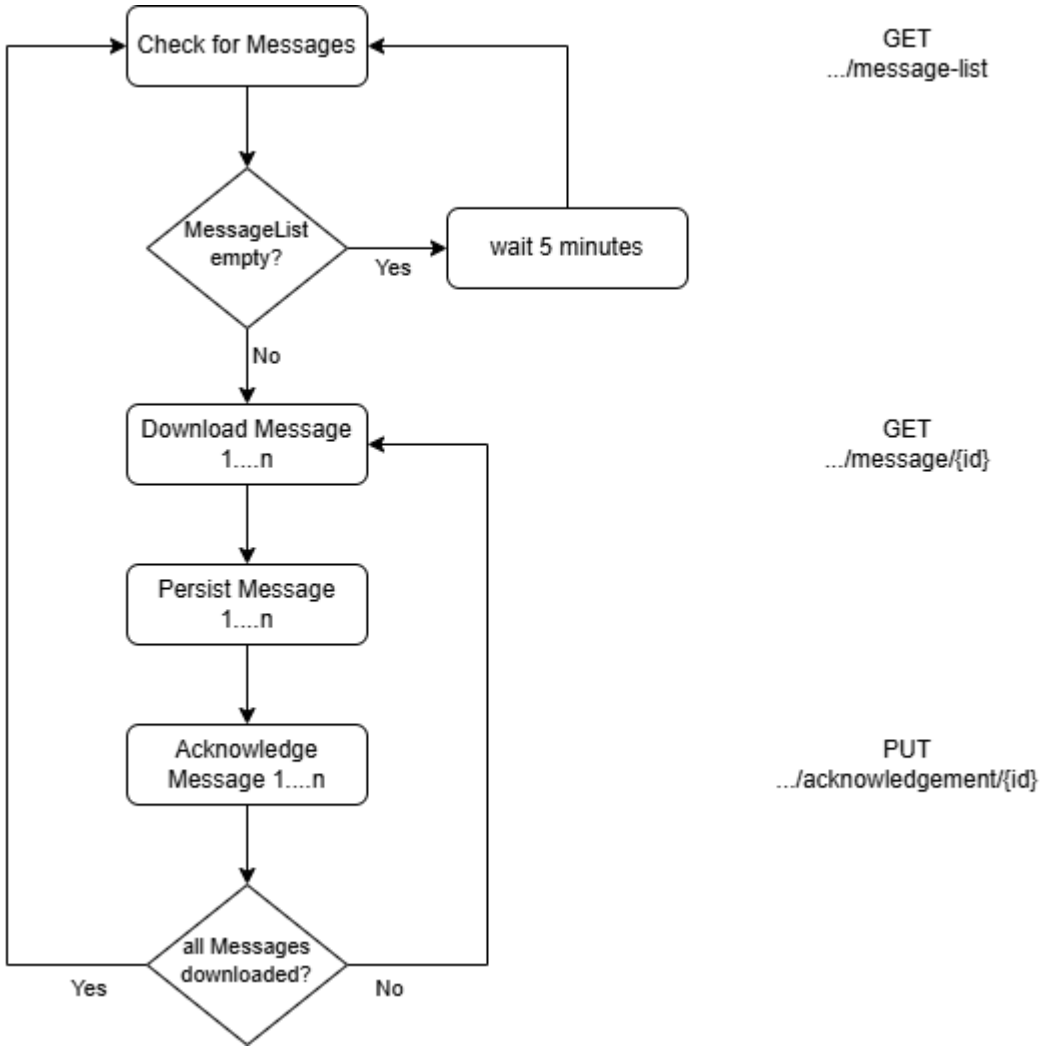
- http 202 Acknowledged
- http 404 Message not found

Suggestions for client implementation

To receive and download messages from the **Communication Gateway: Messaging API** in a robust way, follow this recommended approach:

5. Poll the message-list endpoint every 5 minutes for new messages.
   Use a message limit between 1 and 100.
6. If messages are available, process them one by one:
   6.1. Download the first message by its GMID.
   6.2. Acknowledge the message only after you confirm that your system has downloaded and stored it successfully.
   6.3. Continue with the next message until you process all messages.
7. After completing the prior steps successfully, call the message-list endpoint again immediately and repeat the download and acknowledgment process for any newly available messages.
8. Repeat steps 1 through 3 until the message-list endpoint no longer returns any messages. Then return to the regular polling interval of 5 minutes.

The process would look like this:



Important: Do not run multiple message-list polling loops in parallel. Otherwise, your system may download the same unacknowledged message more than once.

Messages remain visible in the message-list response until you acknowledge them. This behavior can create race conditions if multiple polling loops run at the same time.

You can still download and acknowledge messages in parallel threads, provided that the next message-list request is only executed after all active download threads have finished.

Best practice pseudo-code:

```
DO LOOP
    GET Message List (100)                  -- 100 messages / GET is just an example
    IF <Message List> IS EMPTY
        WAIT 5 min                                              -- Use reasonable polling interval
    ELSE
        PARALLEL (3)                        -- 3 Threads is just an example
            GET <Message-ID> FROM <Message List>   -- Get and reserve new message to fetch
     IF <SUCCESS> THEN
            GET <Message> BY <Message-ID>               -- Fetch message from API
            STORE <Message>                 -- Persist locally
            PUT <Message-ID>                -- Acknowledge
     END IF
        EXIT PARALLEL ONCE ALL THREADS RETURNED
    END IF
UNTIL (<Downloading Stopped>)
```