DEC112 - Deaf Emergency Call 112
Border Gateway API

Version 0.5
Richard Prinz
Jänner 2019

PRELIMINARY VERSION – SUBJECT TO CHANGE
INTERNAL USE ONLY

This page was intentionally left blank.

Contents

1 DEC112 Border Gateway API description 5

1.1 Common objects 5

1.1.1 Timestamps 5

1.1.2 Request / Response 5

1.1.3 Call 6

1.1.4 Message 6

1.2 Errors 7

1.3 Events 7

1.3.1 New call event response (new_call) 8

1.3.2 New message during call (new_message) 8

1.3.3 Call changed its state (state_change) 9

1.4 Triggers 10

1.5 REST API 11

1.6 WebSocket API 12

1.7 API methods 12

1.7.1 get_active_calls 12

1.7.2 get_active_calls_count 13

1.7.3 get_call 13

1.7.4 get_call_alt 15

1.7.5 send 16

1.7.6 close_call 16

1.7.7 subscribe_new_calls 17

1.7.8 subscribe_call 17

Figures

Figure 1 SWAGGER UI Web interface....12

Abreviations

BGW	Border Gateway
DEC112	Deaf Emergency Call
JSON	Javascript Object Notation
PSAP	Public Safety Answering Point
REST	Representational State Transfer
WS	WebSocket

Terms

Call	This document uses the term "Call" for text based (text chat) communication between peers. There is no voice connection in the conventional sense.

1 DEC112 Border Gateway API description

The Border Gateway (BGW) API is accessible via two technologies: REST[1] and WebSocket[2]. Both technologies provide (almost) the same functionalities using the same method names. The WebSocket API provides an event mechanism which allows clients to be actively notified by BGW about new calls or messages.

Both technologies use JSON[3] for request and response.

1.1 Common objects

Some data formats and structures are common for multiple API methods and are described below.

1.1.1 Timestamps

All timestamps used in API messages are in the form `yyyy-mm-ddTHH:MM:SS.ssZ` where y=year, `m`=month, d=day, `M`=minutes, `S`=seconds, `s`=milliseconds. Time is always UTC and should be converted to local time by clients for display.

1.1.2 Location

All locations are in WGS84 degrees provided as follows:

```
{
  "lat": 48.227007156238,
  "lon": 16.3463576417416,
  "alt": 220,
  "rad": null,
  "method": "gps"
}
```

`alt` is the altitude in meters, `rad` a radius around a lat/lon point in meters (which then is actually a circle) and `method` the method how the position was determined (e.g. gps, manual, wifi etc.).

`lat` and `lon` are always provided with valid values (otherwise no location at all would be provided). All other attributes values are ‘null’ if not available.

1.1.3 Request / Response

Base structure of JSON message API call is as follows:

Request:

```
{
  "method": "api method to call",
  "xyz": "additional arguments"
}
```

[1] https://en.wikipedia.org/wiki/Representational_state_transfer
[2] https://tools.ietf.org/html/rfc6455
[3] https://tools.ietf.org/html/rfc7159.html

The server responds with the following base structure:

Response:

```
{
  "code": "result code",
  "runtime_ms": "runtime in ms"
}
```

`runtime_ms` is the runtime of the call to complete in milliseconds - only available if server runs in debug mode.

1.1.4 Call

A call is always provided in the following structure and is either identified by its `call_id` (generated by DEC112 APP) or its `call_id_alt` (provided via customer trigger or `null`).

```
{
  "method": "get_call",
  "call": {
    "created_ts": "2018-04-27T10:13:02.453Z",
    "call_id": "23",
    "call_id_alt": "wa23-996",
    "device_id": "5b80637fdf3bfe13de276791a5178028e8e7b3a0",
    "caller_id": 21,
    "caller":
    "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
    "length": 2,
    "chat": [
      {
        "created_ts": "2018-04-27T10:17:22.677Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      },
      {
        "created_ts": "2018-04-27T10:17:02.471Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      }
    ]
  },
}
```

A call is either active or closed. Active calls are ongoing calls and can be subscribed for events (see 1.7.1, 1.7.2, or 1.7.8). BGW records all messages received and sent during a call so closed calls can be retrieved later as described under 1.7.3 and 1.7.4.

1.1.5 Message

A message is part of a call either received from the caller or sent by PSAP and contains text, location and data portions via fields `texts`, `locations` and `data`. These fields are arrays and can contain 0 or more entries.

NOTE: *Clients must ensure to handle zero or more than one entry in these fields.*

`texts` contains strings, `location` location objects each with `lat` and `lon` properties (in WGS84 degrees) and `data` is an array of arbitrary JSON objects not further specified which holds additional data a user enters in the DEC112 App like medications etc. in key/value form.

NOTE: *A client must handle also possible nested key/key/key/value structures in data.*

The field `received_ts` contains the timestamp when this message was received by the BGW.

NOTE*: At the moment the message object below differs from the message object used in a call.chat structure (see 1.1.4) slightly. This will change in the future so that both message objects are the same.*

```
{
  "received_ts": "2018-04-27T09:50:54.282Z",
  "texts": [
      "test message 1",
      "another message"
  ],
  "locations": [
    {
      "lat": 48.227121736854315,
      "lon": 16.346366358920932,
      "alt": 220,
      "rad": 3500,
      "method": "gps"
    }
  ],
  "data": [
      {
          "name": "Jown Dow",
          "age": 99
      }
  ]
}
```

1.2 Errors

In case of an error the response is:

```
{
  "message": "error message",
  "code": "result code"
}
```

`code` is either 4xx or 5xx and `message` a string containing the error message. If server runs in debug mode `message` contains the full runtime error message (including possible stack traces if available). Otherwise only basic information’s are provided in the `message`.

1.3 Events

The BGW can notify clients about new calls or messages and call state changes. To be notified a client must first subscribe an event by one of the following methods. This only works when using the WebSocket API!

- `subscribe_new_calls`
- `subscribe_new_message`
- `state_change`

The client then receives one of this event notification messages back:

1.3.1 New call event response (new_call)

This message is sent to subscribers of this event when the border gateway receives a new call.

Event:

```
{
  "event": "new_call",
  "created_ts": "2018-04-27T10:13:02.453Z",
  "call_id": 23,
  "call_id_alt": null,
  "caller_uri": "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
  "code": 200
}
```

1.3.2 New message during call (new_message)

This message is sent to subscribers of this event when a new message for an active call is received. This event is sent also when messages are sent from the border gateway using the `send` method (in which case the `origin` attribute has the value `local`).

Event:

```
{
  "event": "new_message",
  "created_ts": "2018-04-27T09:47:56.772Z",
  "call_id": "22",
  "call_id_alt": null,
  "caller_uri": "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
  "message": {
    "received_ts": "2018-04-27T09:50:54.282Z",
    "origin": "remote",
    "texts": ["This is a test message\r\n"],
    "locations": [
      {
        "lat": 48.227121736854315,
        "lon": 16.346366358920932,
        "alt": 220,
        "rad": 3500,
        "method": "gps"
      }
    ],
    "data": []
  },
  "code": 200
}
```

1.3.3 Call changed its state (state_change)

A `state_change` event is sent whenever the state of a call changes. The following numerical value states are defined:

1. UNDEFINED
2. NEW_CALL
3. IN_CALL
4. STALE
5. CLOSED_BY_CALLER
6. CLOSED_BY_CENTER
7. CLOSED_BY_SYSTEM
8. ERROR

UNDEFINED (0)

Should (and is) not be used.

NEW_CALL (1)

Sent after receiving initial message of a new call. This event is sent AFTER the `new_call` and `new_message` events are sent for this call.

IN_CALL (2)

Sent when the second message of a call is received and after the `new_message` event was sent.

STALE (3)

A call enters this state after not receiving any messages for more than `sip.call_stale_timeout_ms` (configuration option).

CLOSED_BY_CALLER (4)

Sent when caller ends call.

CLOSED_BY_CENTER (5)

Sent when PSAP ends call.

CLOSED_BY_SYSTEM (6)

Sent when call is in STALE state and for not receiving any messages for more than `sip.call_close_timeout_ms` (configuration option)

ERROR (8)

Sent when call enters error state.

Event:

```
{
  "event": "state_change",
  "created_ts": "2018-04-27T10:13:02.453Z",
  "call_id": 23,
  "call_id_alt": null,
  "caller_uri": "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
  "state": 2,
  "code": 200
}
```

1.4 Triggers

Depending on configuration of the BGW the PSAP can be notified about new or closed DEC112 calls by means of triggers. Triggers are provided as plugins to the BGW and can be anything: from JSON or XML posted via http to a PSAP backend system over a file written to a file share up to a simple I/O pin set to high or low. Currently a standard DEC112 trigger is defined which posts the following JSON structures via http POST to a receiving PSAP webserver. BGW can send triggers when calls are opened and when calls are closed (this can be identified by inspecting ‘trigger_type’ property).

Open trigger example:

```
{
    "trigger_type": "OPEN",
    "call_id": "59d72799-97cb-4da0-8b90-6d8118deb8a3",
    "call_id_alt": null,
    "device_id": "5b80637fdf3bfe13de276791a5178028e8e7b3a0",
    "caller_uri": "sip:c09f1aabc26bccca48756@service.dec112.at",
    "caller_name": "Max Mustermann <sip:c09f1aabc26bccca48756@service.dec112.at>",
    "caller_id": "21",
    "called_uri": "sip:133@service.dec112.at",
    "lang": "en",
    "is_test": false,
    "service": "default",
    "sip_history": [],
    "client_version": "1.3.3",
    "received_ts": "2018-09-28T13:27:56.047Z",
    "origin": "remote",
    "web_view_url": "http://service.dec112.at:8080/viewer/?call_id=59d72799-97cb-
4da0-8b90-6d8118deb8a3&api_key=secret_key",
    "api_url": "http://service.dec112.at:8080/api/v1/call/c59d72799-97cb-4da0-
8b90-6d8118deb8a3?api_key=secret_key",
    "texts": [
        "This is a deaf emergency call from Max Mustermann  (
         Phone: 00436769513801) sent at 09/28/2018, 3:27:56 PM.
         Current position Lat: N 48.19293239; Lon: E 16.33494889.\r\n"
    ],
    "locations": [
        {
            "lat": 48.19293239,
            "lon": 16.33494889,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
        }
    ],
    "data": [
        {
            "name": "Max Mustermann",
            "tel": "00436769513801",
            "email": "info@min.at",
            "adr": {
                "street": "Strasse 42",
                "locality": "Dort",
                "code": "1234",
                "country": "Irgendwo"
            },
            "notes": {
                "Geschlecht": "M",
                "Größe": "175 cm",
                "Blutgruppe": "B"
            }
        }
    ]
}
```

Close trigger example:

```
{
    "created_ts": "2018-09-28T13:27:56.047Z"
    "caller_uri": "sip:c09f1aabc26bccca48756@service.dec112.at",
    "caller_name": "Max Mustermann <sip:c09f1aabc26bccca48756@service.dec112.at>",
    "call_id": "59d72799-97cb-4da0-8b90-6d8118deb8a3",
    "call_id_alt": null,
    "lang": "en",
    "is_test": false,
    "trigger_type": "CLOSE",
    "web_view_url": "http://service.dec112.at:8080/viewer/?call_id=59d72799-97cb-
4da0-8b90-6d8118deb8a3&api_key=secret_key",
    "api_url": "http://service.dec112.at:8080/api/v1/call/c59d72799-97cb-4da0-
8b90-6d8118deb8a3?api_key=secret_key"
}
```

The open trigger contains all SIP caller information as well as location and base data. The first message sent by the DEC112 App is sent automatically without user intervention and contains a default text, the current location and all additional data the user has entered in the App like blood group, medications etc.

There is no need for the PSAP system to answer this triggers. The BGW can be configured to send such triggers as “fire and forget”.

DEC112 generates its own call IDs and PSAPs do normally the same. To identify a DEC112 call not only by its DEC112 call ID the PSAP system can answer the open trigger providing an alternate (PSAP generated) call ID. The predefined DEC112 trigger extracts the alternate call ID from the following response:

```
{
    "call_id_alt": "xyz0815"
}
```

The BGW stores this alternated ID so that calls can be accessed with the `get_call_alt` (see 1.7.4) method. With this method the DEC112 call ID can be determined for further API calls from the alternate PSAP call ID.

1.5 REST API

This version of the REST API is reachable on the BGW via http://host:port/api/v1.

For testing, development, and automatic client generation, the API provides SWAGGER[4] metadata (including the interactive SWAGGER UI) accessible via http://host:port/api-v1/api-docs.(SWAGGER metadata) and http://host:port/api/ (SWAGGER UI).

To use the REST API a client needs an API key. This key can be provided either as http

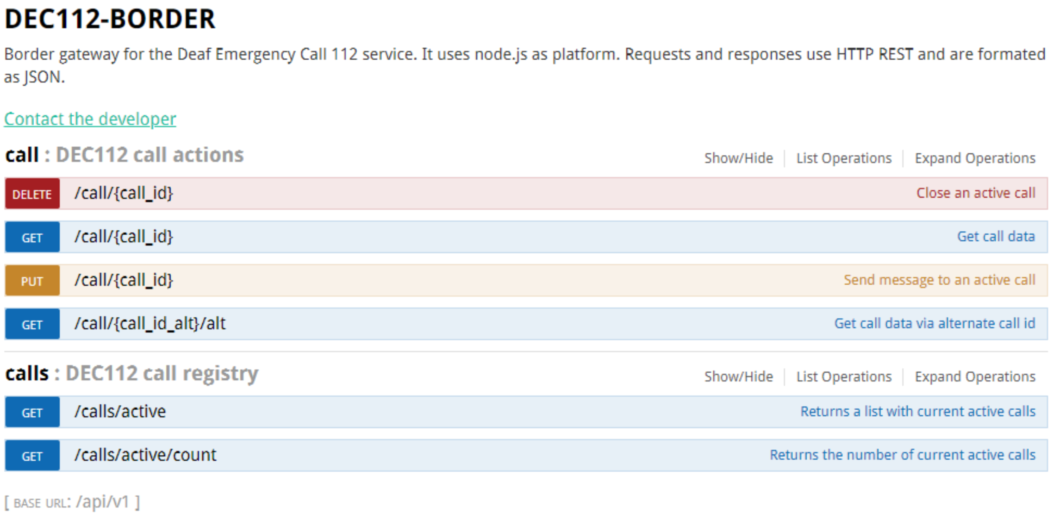
header 'api_key' or as URL parameter 'api_key='.

http header:

```
api_key: my_secret_api_key
```

URL parameter:

http://host:port/api/v1/calls/active?api_key=my_secret_api_key



DEC112-BORDER

Border gateway for the Deaf Emergency Call 112 service. It uses node.js as platform. Requests and responses use HTTP REST and are formated as JSON.

Contact the developer

call : DEC112 call actions

DELETE	/call/{call_id}	Close an active call
GET	/call/{call_id}	Get call data
PUT	/call/{call_id}	Send message to an active call
GET	/call/{call_id_alt}/alt	Get call data via alternate call id

calls : DEC112 call registry

GET	/calls/active	Returns a list with current active calls
GET	/calls/active/count	Returns the number of current active calls

[BASE URL: /api/v1]

Figure 1 SWAGGER UI Web interface

1.6 WebSocket API

The WebSocket API is reachable on the same base URL as the REST API with a `ws` scheme e.g ws://host:port/api/v1. It defines the `dec112` WebSocket protocol which must be provided when opening the socket.

The protocol uses JSON messages as described above and in the following sections.

To use the REST API a client needs an API key. This key can be provided either as http header 'api_key' or as URL parameter 'api_key='.

http header:

[4] https://swagger.io/

```
api_key: 48a4f4deda61fcdca604dc4db993c2e0
```

URL parameter:

ws://host:port/api/v1?api_key=48a4f4deda61fcdca604dc4db993c2e0

1.7 API methods

The action to perform is specified with the `method` property as described below. Optional the `tag` property can be specified with any arbitrary value to match request and response by the client.

1.7.1 get_active_calls

Returns the current active calls.

REST Request

http method: GET

http://host:port/api/v1/calls/active?api_key=my_secret_api_key

WS Request:

```
{
  "method": "get_active_calls"
}
```

Response:

```
{
  "method": "get_active_calls",
  "calls": [
    {
      "created_ts": "2018-04-27T10:13:02.453Z",
      "caller_uri": "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
      "call_id": "23",
      "call_id_alt": null
    }
  ],
  "code": 200,
  "runtime_ms": "1.06438"
}
```

1.7.2 get_active_calls_count

Returns the number of current active calls.

REST Request

http method: GET

http://host:port/api/v1/calls/active/count?api_key=my_secret_api_key

WS Request:

```
{
  "method": "get_active_calls_count"
}
```

Response example:

```
{
  "method": "get_active_calls_count",
  "count": 1,
  "code":200,
  "runtime_ms": "0.38333"
}
```

1.7.3 get_call

Returns the call identified by `call id (e.g. 23)`.

REST Request

http method: GET

http://host:port/api/v1/call/23?api_key=my_secret_api_key

WS Request:

```
{
  "method": "get_call",
  "call_id": "23"
}
```

Response example:

```
{
  "method": "get_call",
  "call": {
    "created_ts": "2018-04-27T10:13:02.453Z",
    "call_id": "23",
    "call_id_alt": "wa23-996",
    "device_id": "5b80637fdf3bfe13de276791a5178028e8e7b3a0",
    "caller_id": 21,
    "caller":
    "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
    "length": 14,
    "chat": [
      {
        "created_ts": "2018-04-27T10:17:22.677Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      },
      {
        "created_ts": "2018-04-27T10:17:02.471Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      }
    ]
  },
  "code": 200,
  "runtime_ms": "60.70412"
}
```

1.7.4 get_call_alt

Returns the call identified by alternate call ID in `call_id_alt` (e.g. wa23-996)

REST Request

http method: GET

http://host:port/api/v1/call/wa23-996/alt?api_key=my_secret_api_key

WS Request:

```
{
  "method": "get_call_alt",
  "call_id_alt": "wa23-996"
}
```

Response example:

```
{
  "method": "get_call_alt",
  "call": {
    "created_ts": "2018-04-27T10:13:02.453Z",
    "call_id": "23",
    "call_id_alt": "wa23-996",
    "device_id": "5b80637fdf3bfe13de276791a5178028e8e7b3a0",
    "caller_id": 21,
    "caller": "sip:a36e0d23777c09f1aabc26bccca48756@service.dec112.at",
    "length": 14,
    "chat": [
      {
        "created_ts": "2018-04-27T10:17:22.677Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      },
      {
        "created_ts": "2018-04-27T10:17:02.471Z",
        "origin": "remote",
        "texts": [],
        "locations": [
          {
            "lon": 16.3463576417416,
            "lat": 48.227007156238,
            "alt": 220,
            "rad": 3500,
            "method": "gps"
          }
        ],
        "data": []
      }
    ]
  },
  "code": 200,
  "runtime_ms": "60.70412"
}
```

1.7.5 send

Send a message to active call identified by `call_id` (e.g. 23)

Note: *the response is sent back to the client the moment the underlying SIP message is either successfully sent or raised an error. This can, depending on the SIP stack state, take up to a few seconds. The response result code represents the underlying SIP response code.*

REST Request

http method: PUT

http://host:port/api/v1/call/23?api_key=my_secret_api_key

```
{
  "message": "Hello World"
}
```

WS Request:

```
{
  "method": "send",
  "call_id": "23",
  "message": "Hello World"
}
```

Response example:

```
{
  "method": "send",
  "code":200,
  "runtime_ms": "0.391"
}
```

1.7.6 close_call

Actively closes a call identified by `call_id`. An optional `message` could be provided which will be sent to the caller. If no `message` is provided a default, localized message configured on the BGW will be sent. To prevent a message being sent, set the `message` attribute to the string `//SILENT`.

REST Request

http method: DELETE

http://host:port/api/v1/call/23?api_key=my_secret_api_key

```
{
  "message": "Hello World"
}
```

WS Request:

```
{
  "method": "close_call",
  "call_id": "23",
  "message": "Hello World"
}
```

Response example:

```
{
  "method": "close_call",
  "code":200,
  "runtime_ms": "0.391"
}
```

1.7.7 subscribe_new_calls

Subscribe the `new_call` event. After this call the `new_call` event is sent when the border gateway receives a new call.

REST Request:

Not available.

WS Request:

```
{
  "method": "subscribe_new_calls"
}
```

Response example:

```
{
  "method": "subscribe_new_calls",
  "code":200,
  "runtime_ms": "0.49305"
}
```

1.7.8 subscribe_call

Subscribe to be notified for a specific call identified by call_id to be notified when new messages are received for this call. A `new_message`event will be sent in this case.

REST Request:

Not available.

WS Request:

```
{
  "method": "subscribe_call",
  "call_id": "42"
}
```

Response example:

```
{
  "method": "subscribe_call",
  "code":200,
  "runtime_ms": "0.42305"
}
```

--- End of document ---