

Getting Started using O2G REST API Services

This guide is intended for developers who want to start using Alcatel-Lucent Enterprise OmniPCX Open Gateway (O2G) REST API server. It helps understanding how to establish a connection, create an application session, and to start invoking API telephony endpoint services such as telephony monitoring.

ed 2402a

Legal notice:

www.al-enterprise.com The Alcatel-Lucent name and logo are trademarks of Nokia used under license by ALE. To view other trademarks used by affiliated companies of ALE Holding, visit: www.al-enterprise.com/en/legal/trademarks-copyright. All other trademarks are the property of their respective owners. The information presented is subject to change without notice. Neither ALE Holding nor any of its affiliates assumes any responsibility for inaccuracies contained herein. © Copyright 2024 ALE International, ALE USA Inc. All rights reserved in all countries.

Table of content

1	Introduction	3
1.1	Purpose of the document.....	3
1.2	Outlined O2G API flow diagrams.....	4
2	Getting O2G information	5
3	Authenticate with O2G	6
4	Establish a session with O2G	7
5	Session keepalive	8
6	Get OXE user information	9
6.1	Get OXE user login names	9
6.2	Get OXE user detailed information.....	10
7	Subscribing to services.....	11
8	Use case: Monitoring basic call telephony events	12
9	Use case: Monitoring unsupervised transfer call telephony events.....	21

1 Introduction

1.1 Purpose of the document

OmniPCX Open Gateway (refer to as O2G) provides an easy integration for business communications into your applications or processes with Alcatel-Lucent OmniPCX® Enterprise Communications Server (refer to as OXE).

Developers access OmniPCX Enterprise Communication Server capabilities, including call control, call center, management and analytics using open web services provided by OmniPCX Open Gateway (O2G) REST APIs.

This document helps understanding from an application point of view how to authenticate, create an application session, and start invoking O2G API endpoint services. It outlines the corresponding O2G API flow diagrams, content of API requests/responses and events issued by OmniPCX Enterprise through O2G (See *Figure 1: "Architecture overview"*)

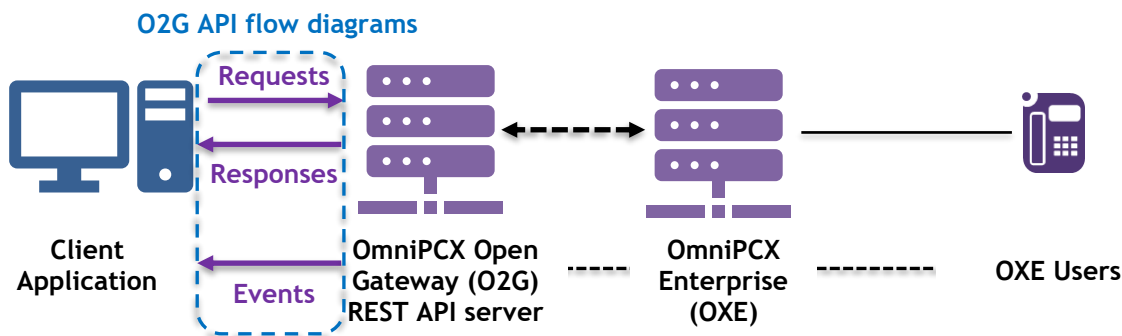


Figure 1: "Architecture overview"

Notes:

The purpose of this document is **not to provide a complete description** of the O2G REST API. Examples and flow diagrams described in the document do not expose the full range of parameters and options available to implement more complex use cases. While the fundamental principles remain valid, the content of the requests, responses, and events provided as examples to illustrate the usage of a REST API in the document may vary slightly from the most recent published version of the API.

If you wish to further explore your understanding and usage of the OmniPCX Open Gateway, Alcatel-Lucent Enterprise offer various options :

- **O2G REST API reference documentation:** a comprehensive and detailed description of O2G REST API.
<https://api.dspp.al-enterprise.com/omnipcx-open-gateway-02g/>
- **Alcatel-Lucent Enterprise Developer & Solution Partners Program (DSPP):** provides resources (sandbox) and support for the development and test of your application on top of O2G and OXE.
<https://api.dspp.al-enterprise.com/contact-us/>
- **Alcatel-Lucent Enterprise OmniPCX Open Gateway Mentoring Service Pack:** helps preparing your O2G deployments and software integration with the accompaniment of dedicated Alcatel-Lucent Enterprise experts.
Please, refer to your local Alcatel-Lucent Enterprise representative.

Examples and flow diagrams described in the document depend on setting up correctly O2G server infrastructure, acquiring necessary licenses, and establishing secure access through networking equipment. For this purpose, please refer to *O2G server installation manual* (available for Alcatel-Lucent Enterprise Business Partners).

1.2 Outlined O2G API flow diagrams

This document contains some essential O2G API flow diagrams depicting requests/responses and asynchronous events sent to/received from a client application to/from O2G server using O2G REST API. Basic examples illustrate the sequence of events received after subscribing to common services such as Telephony services and issued by some fundamental telephony interactions. This guide details the flow diagrams shown in *Figure 2: “O2G API flow diagrams outlined in this document”*

O2G API flow diagrams - Overview

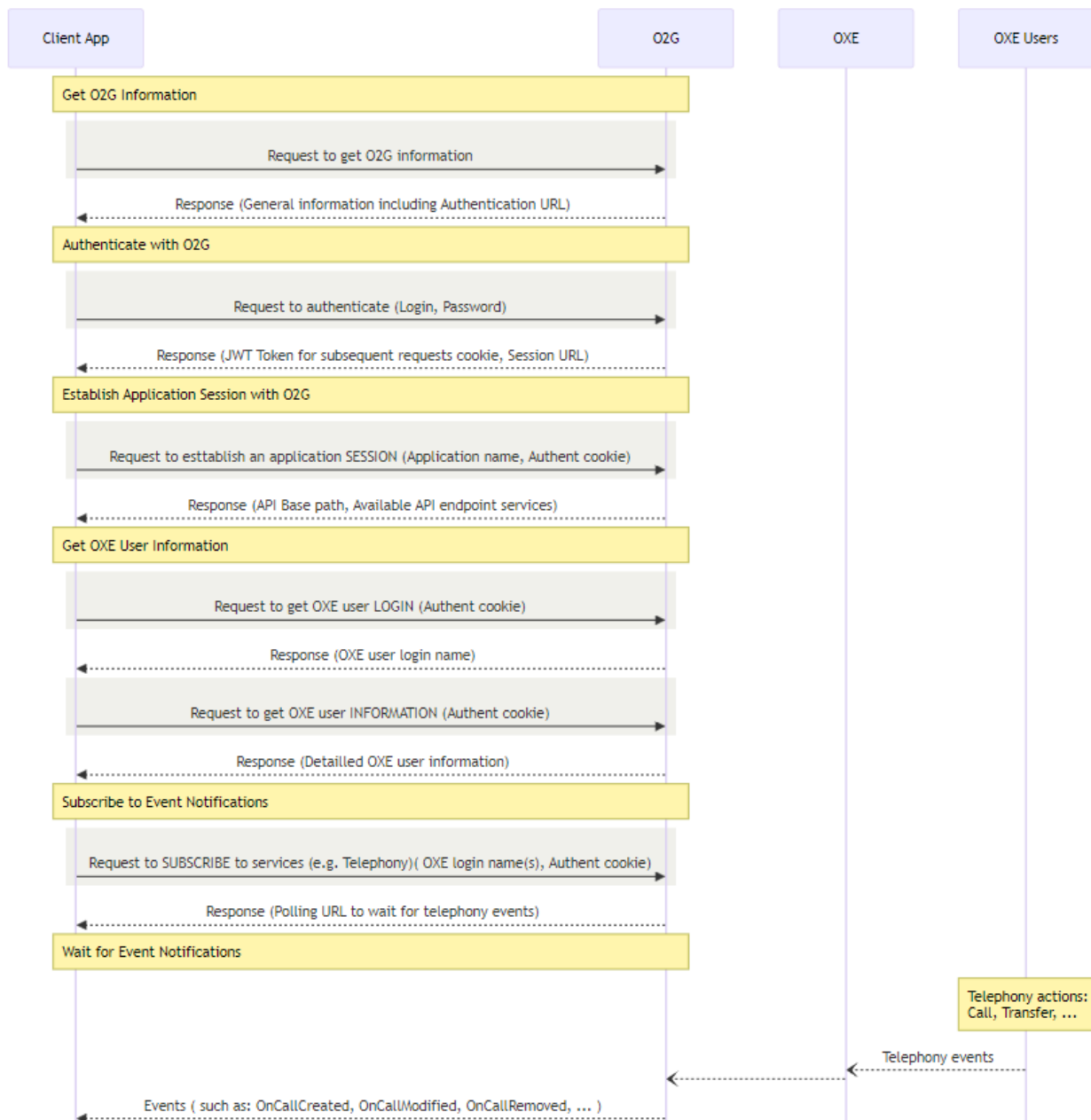


Figure 2: “O2G API flow diagrams outlined in this document”

2 Getting O2G information

An application can retrieve general information about O2G server REST API by implementing a GET http(s) request (❶) on the O2G hostname URL such as: ***http(s)://<host>:<port>/api/rest***

Upon successful completion of the request, the returned body (❷) includes, among other information, the URLs needed for authentication in the next step. Two different URLs may be returned based on the access point to the API: one for intranet access (*internalUrl*) and another for internet access (*publicUrl*).

O2G API flow diagram - Getting O2G Information



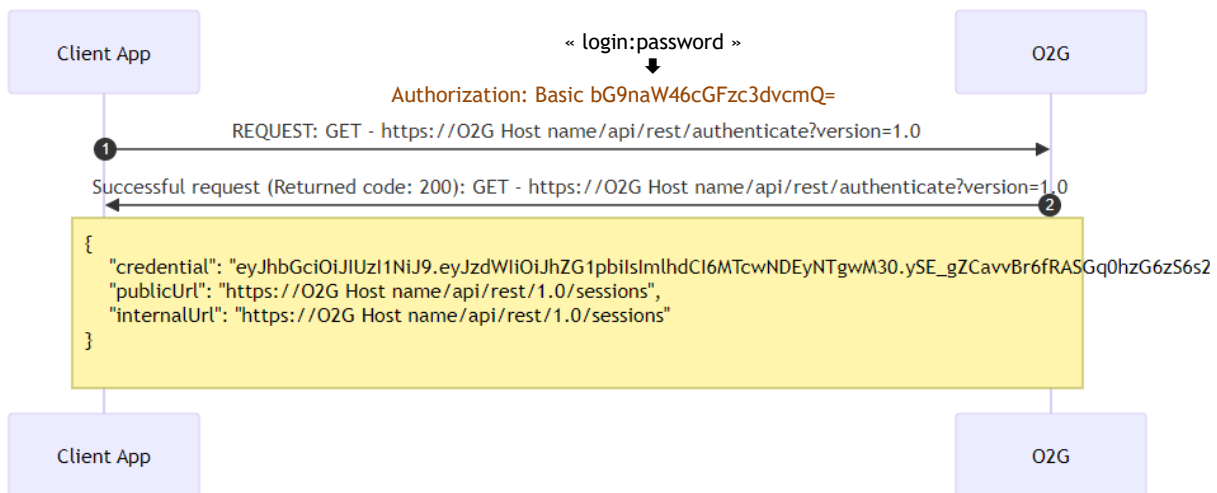
3 Authenticate with O2G

The first step for an application using O2G API is to authenticate with O2G using HTTP basic authentication sending a **login** and the associated **password** encoded in Base64 within the request header (❶) at one of the authentication URLs previously retrieved with O2G information. The login and password can be those of an OXE user or an O2G administrator account :

- For an OXE user:
 - o Login: is the extension number of the OXE subscriber prefixed by "oxe" (e.g. "oxe1001")
 - o Password: is the OXE user password associated to this extension number
- For an administrator
 - o Login: is an O2G account name managed within O2G server (e.g. "admin")
 - o Password: is the password associated to this O2G account

Upon successful completion of the request, the response (❷), includes a JWT token (*credential*) that has been calculated by O2G server based on login and password provided. This token must be passed in a cookie (AlcUserId) in all subsequent HTTP(S) requests, so that O2G can check their validity. URLs needed for session establishment in the next step are also provided in the response. Two different URLs may be returned based on the access point to the API: one for intranet access (*internalUrl*) and another for internet access (*publicUrl*).

O2G API flow diagram - Authenticate with O2G



4 Establish a session with O2G

After authentication, the application needs to establish a session with O2G by executing a request to the `'/sessions'` API service using one of the URLs retrieved in the Authentication step.

For that purpose, send an HTTP POST JSON request with an application name provided in the request body (❶). Ensure to include the 'AlcUserId' header cookie with the JWT token value received during the Authentication phase.

Upon successful completion of the request, the response (❷) includes the base URL paths to invoke O2G services along with the list of available services and their relative URL. As for previous request, two different base URLs may be returned based on the access point to the API: one for intranet access (`privateBaseUrl`) and another for internet access (`publicBaseUrl`).

O2G API flow diagram - Establish a session with O2G



.... List of available services

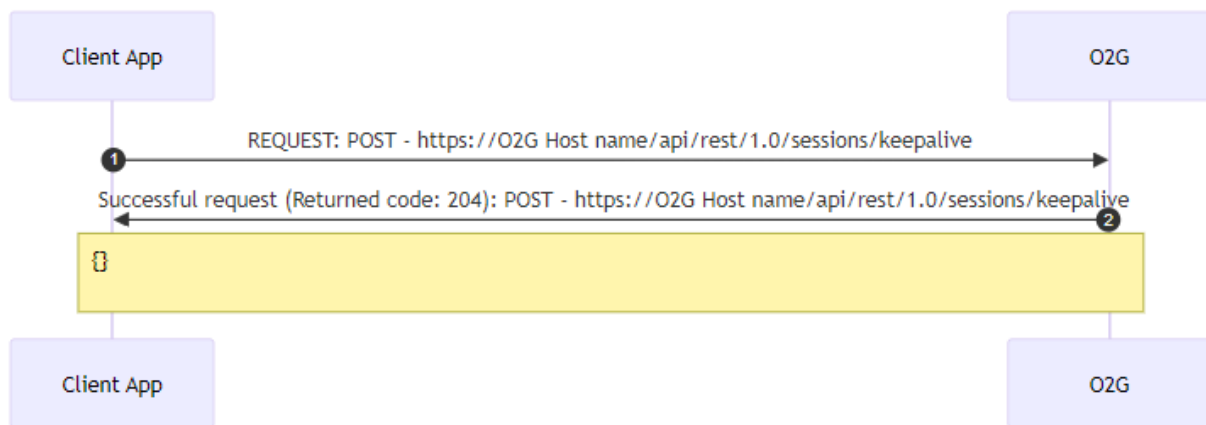
5 Session keepalive

For each session, O2G sets a 30-minute timer, and closes the session if no requests are made. O2G resets the timer to the 30-minute duration upon receiving any request or keepalive.

A keepalive can be sent using an HTTP POST request on “*/sessions/keepalive*” (❶). Ensure to include the 'AlcUserId' header cookie with the JWT token value received during the Authentication phase.

Upon successful completion of the request, a 204 HTTP response (❷) is returned indicating that the keepalive request has been considered and the timer has been reset.

O2G API flow diagram - Session keepalive



6 Get OXE user information

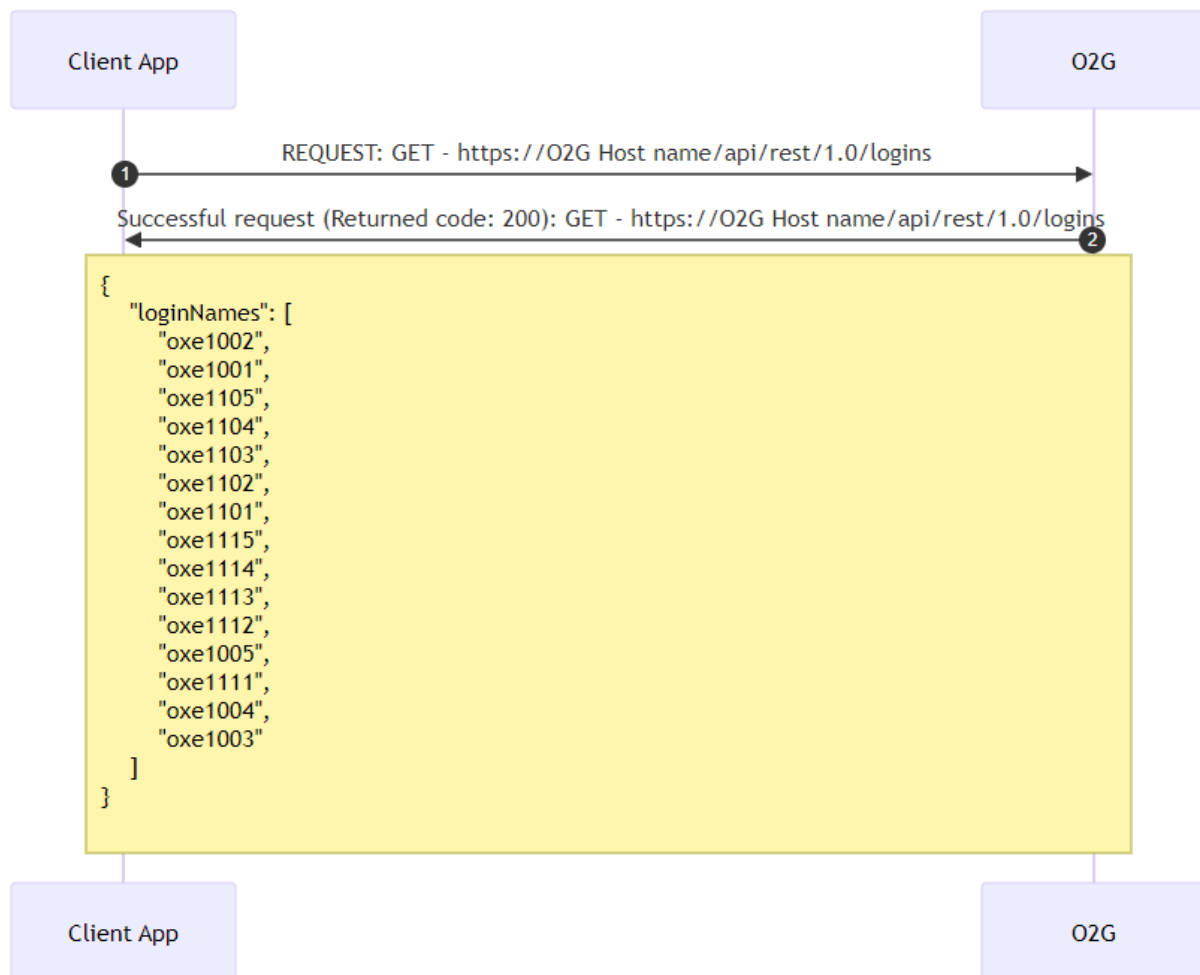
6.1 Get OXE user login names

A typical way of implementing an application using O2G REST API is to subscribe for a category of events issued by a subset of OXE users to be monitored. To do that, a first step consists in getting the login names of OXE users. To get OXE users login names send an HTTP GET request using the *logins* API endpoint service (❶). Ensure to include the 'AlcUserId' header cookie with the JWT token value received during the Authentication phase.

Upon successful completion of the request, the response (❷) includes the list of the login names.

Note: A session established using an administrator account can get the list of all the users (login names). With a session established for a user, only information of this user (login name) is returned. The flow diagram shown below illustrates a request/response exchange when using a session established using an admin account whereby a list of login names is returned.

O2G API flow diagram - Get OXE user logins



6.2 Get OXE user detailed information

It might be of interest for an application to get details about OXE users. To get detailed information about an OXE user send an HTTP GET request using the `users/{login name}` API endpoint service (❶). Ensure to include the 'AlcUserId' header cookie with the JWT token value received during the Authentication phase.

Upon successful completion of the request, the response (❷) includes detailed information about the user.

Note: A session established using an administrator account can get detailed information about any user. For an applicative session established for a user, only details for that user can be accessible.

The flow diagram shown below illustrates a request/response exchange to retrieve detailed information about an OXE user.

O2G API flow diagram - Get OXE user information



7 Subscribing to services

Once a session with the O2G server is successfully established, a typical client application implementation involves subscribing to a category of services (e.g. Telephony) and to wait for events generated by the activity of the subscribed service category.

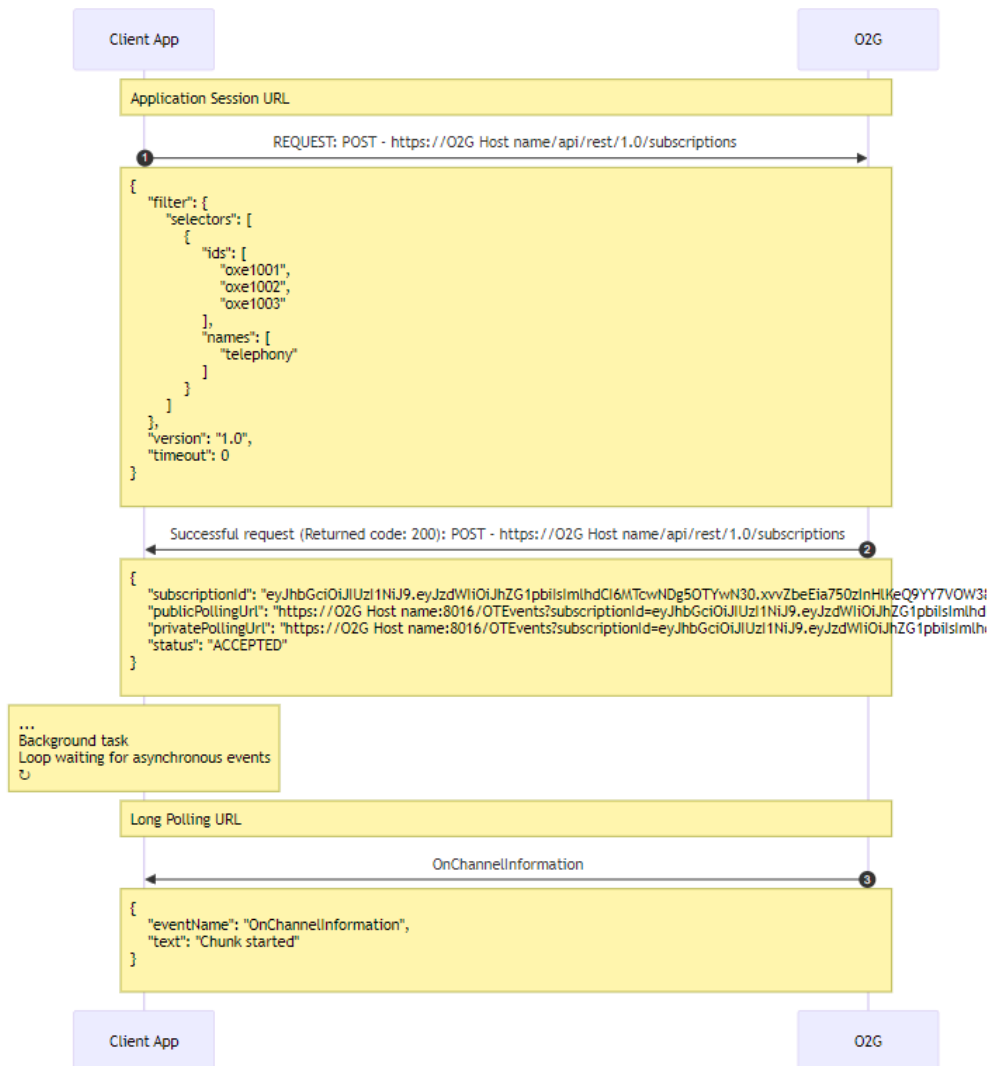
As an example, within a session initiated with an admin account, it is possible to subscribe to Telephony events awaited from a subset of OXE users by sending an HTTP POST JSON request (❶) providing the list of OXE users and the “telephony” category of events in “filter” and “selectors” of request body. Ensure to include the ‘AlcUserId’ header cookie with the JWT token value received during the Authentication phase.

Upon successful completion of the request:

- A response (❷) is returned including long polling URL paths that a background task of the Client App can use to wait on events generated by telephony activity and delivered by the server asynchronously. Two different URLs may be returned based on the access point to the API: one for intranet access (privatePollingUrl) and another for internet access (publicPollingUrl).
- A first event “OnChannelInformation” is provided on the long polling URL (❸) indicating that the event generation process is ready and started: “Chunk started”.

Note: Providing the parameter “timeout” set to 0 value in the request (❶) indicates O2G server to maintain the eventing channel on the long polling URL without timeout.

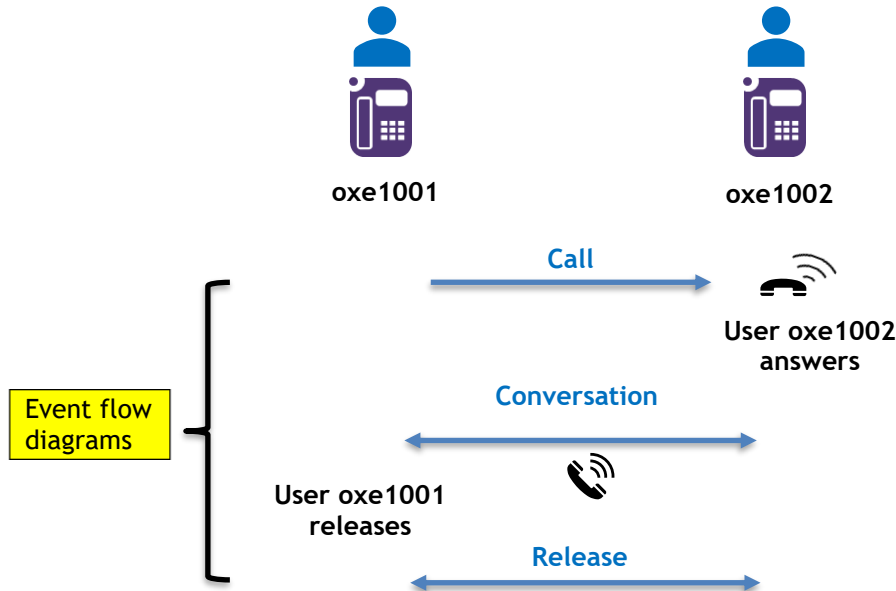
O2G API flow diagram - Subscribe to Telephony Events



8 Use case: Monitoring basic call telephony events

The flow diagram illustrates the sequence of Telephony events that an application receives when monitoring OXE SIP Extensions oxe1001 and oxe1002 for the following basic call use case:

- User oxe1001 calls User oxe1002
- User oxe1002 answers the call from User oxe1001
- User oxe1001 releases the call

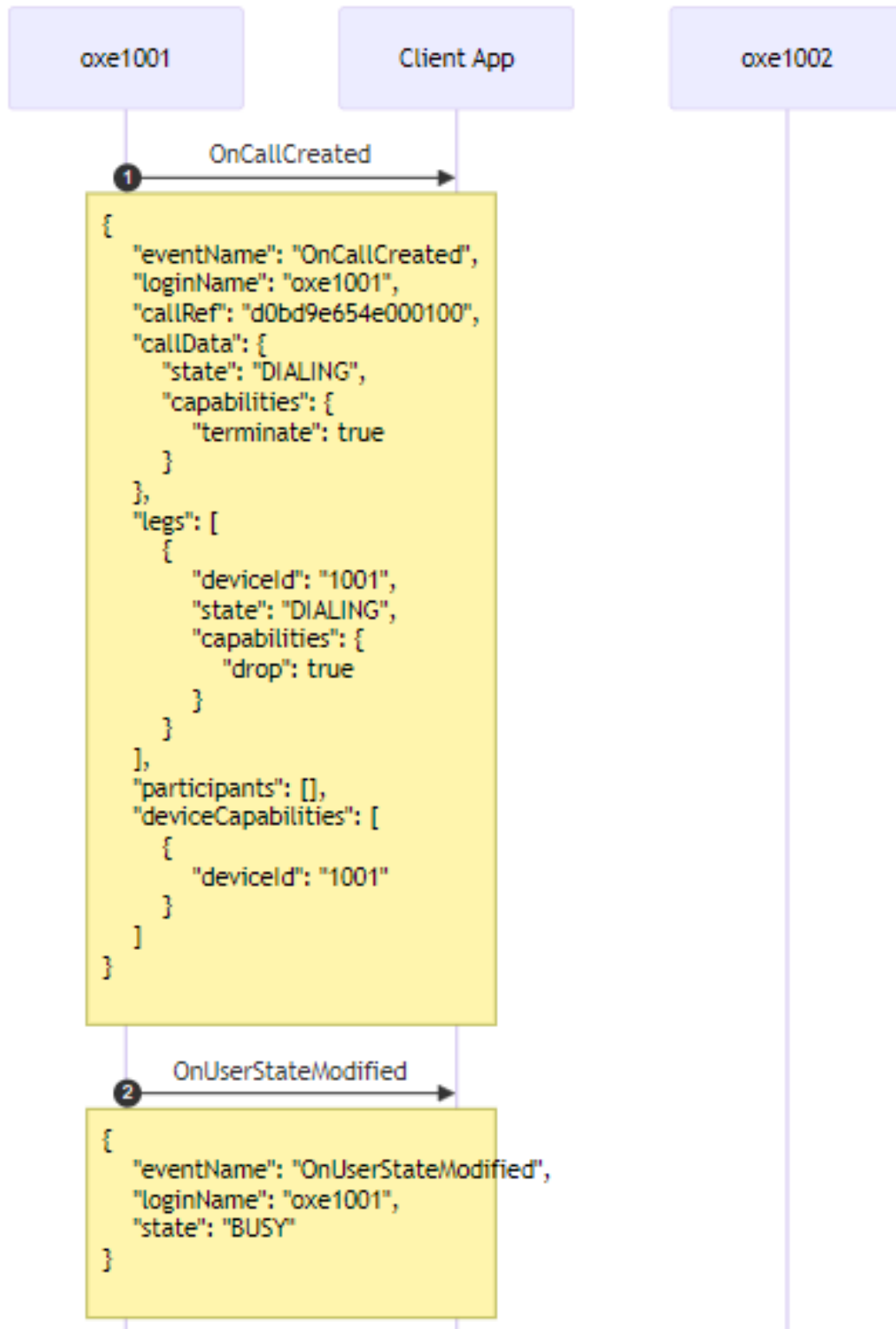


The Client Application must have subscribed to “telephony” category of events with an admin session(*) for the users involved in the use case (refer to “subscribing to services” section).

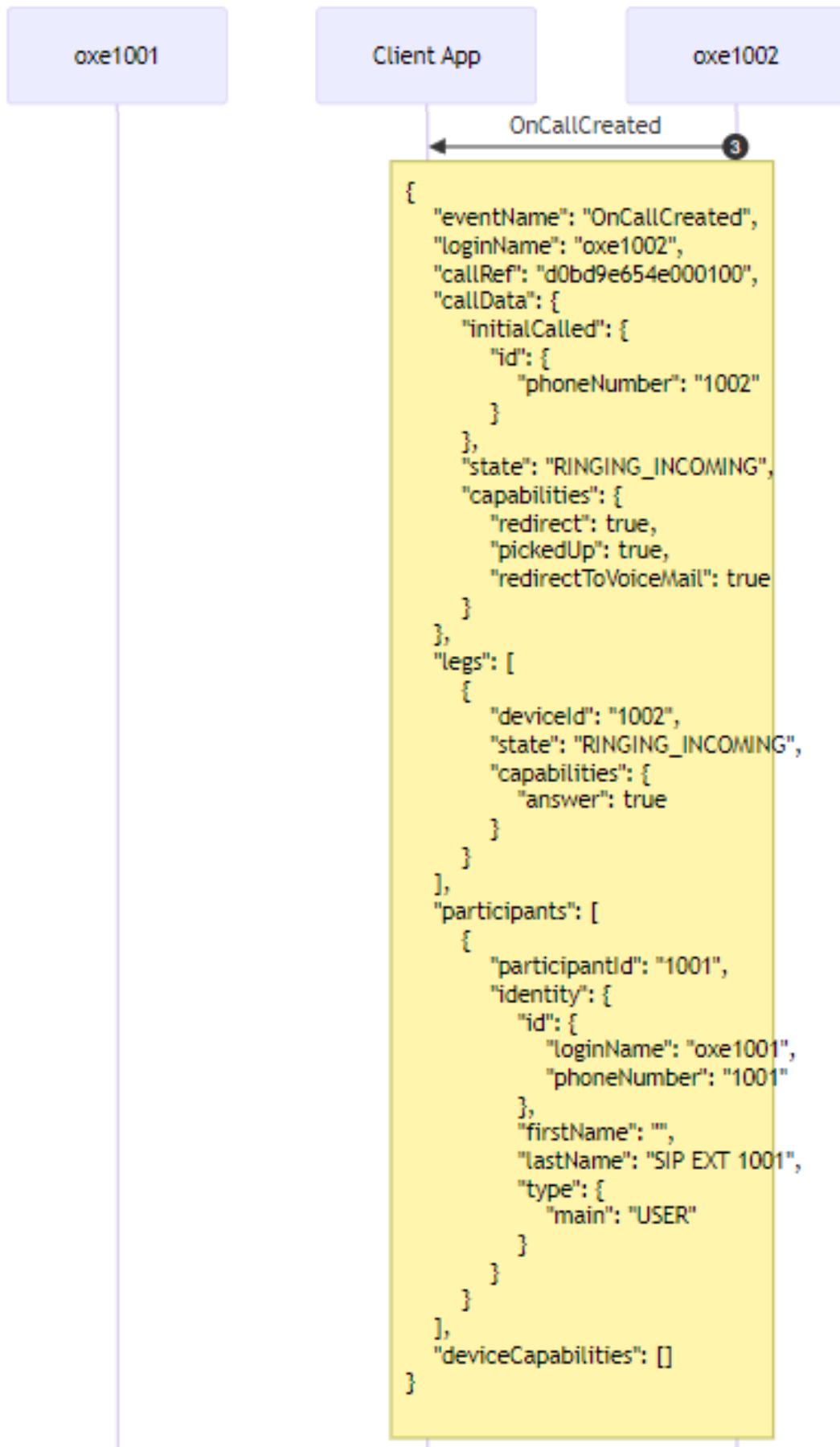
(*) Using an admin session allows the application to monitor several OXE users.

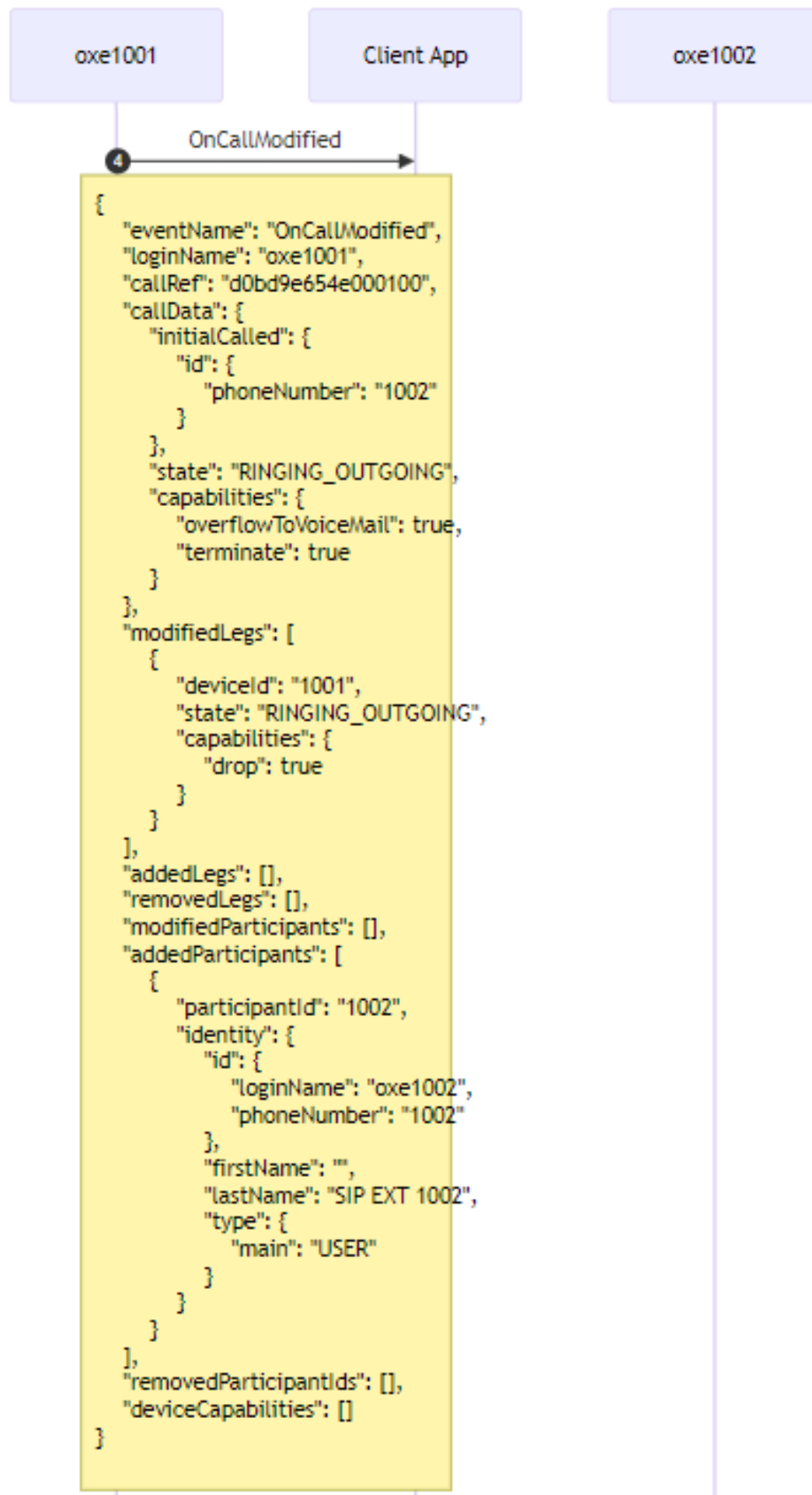
Please note that to simplify the representation of the event flow diagrams in the following pages, these diagrams only focus on the events received by the Client application using the O2G REST API to monitor user telephony activity of OXE. They do not depict OXE and O2G servers that generates and transmits these events.

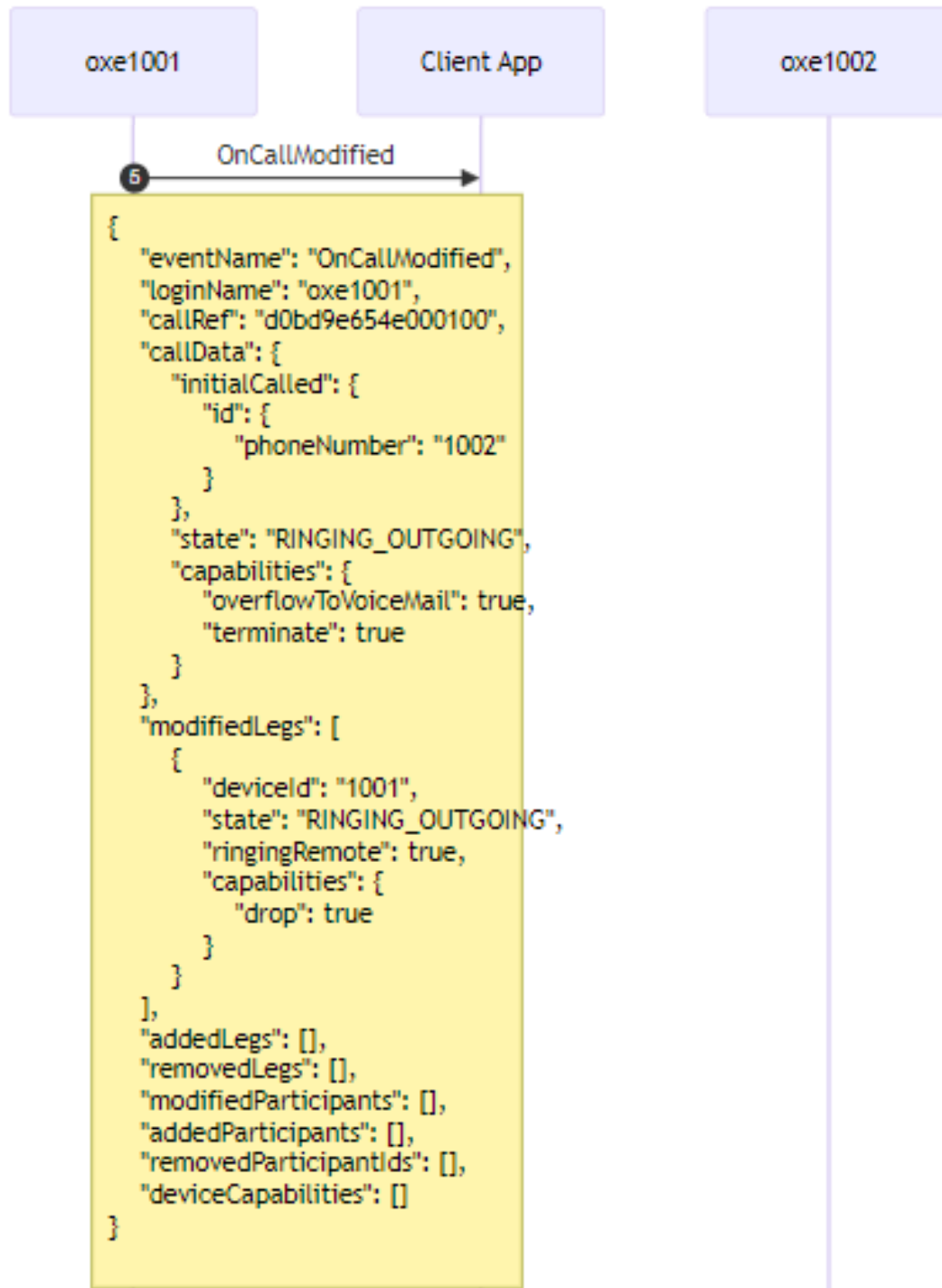
- oxe1001 calls oxe1002, the resulting call is identified by “callRef” (call Reference)



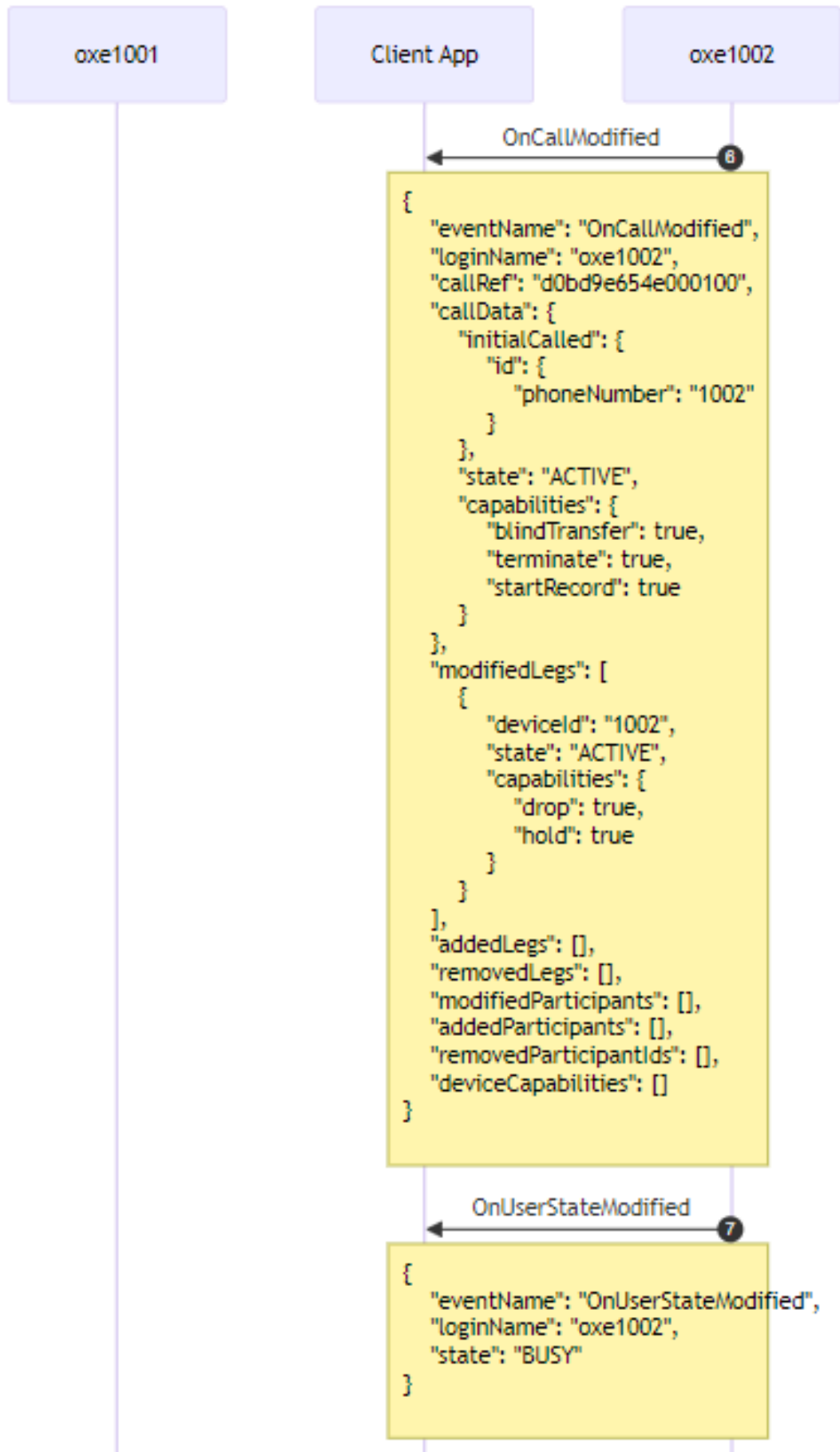
- Ringing phases

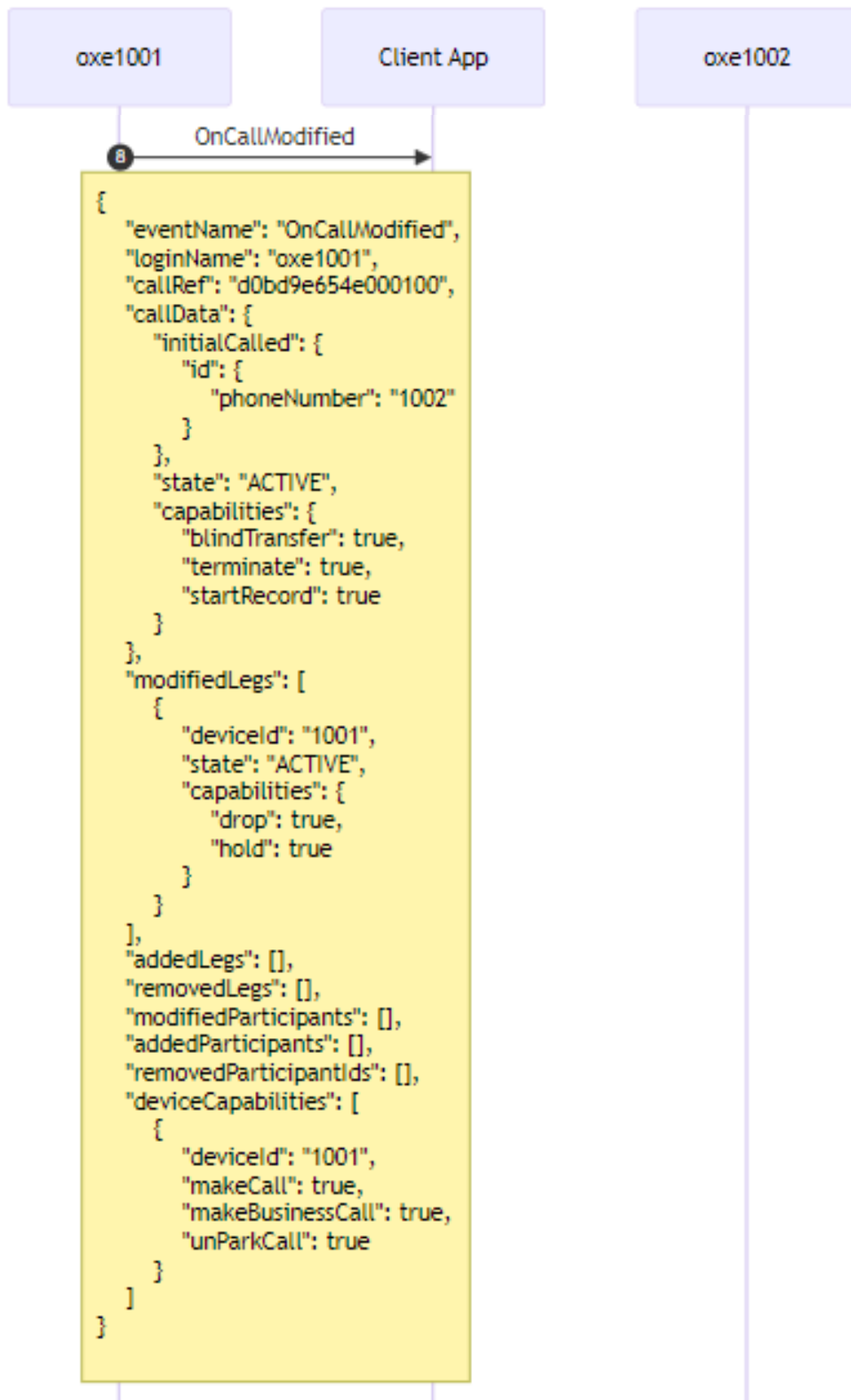




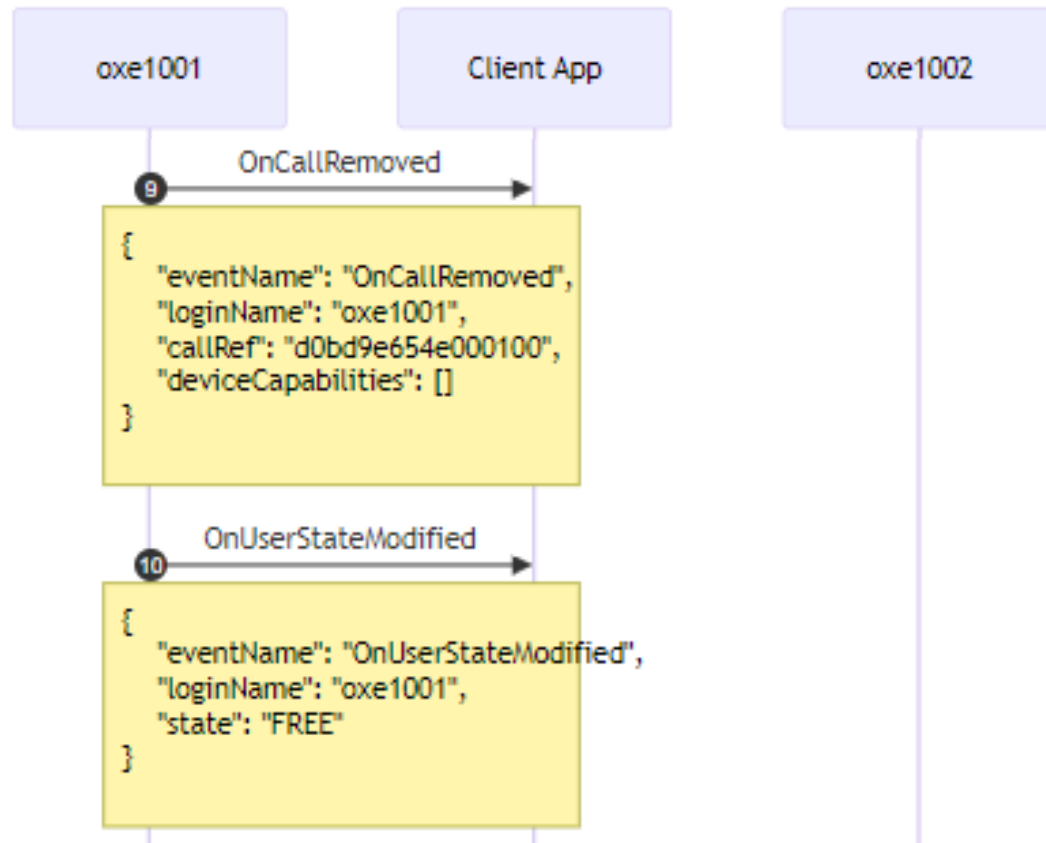


- oxe1002 answers to the call => call is established between 1001 and 1002

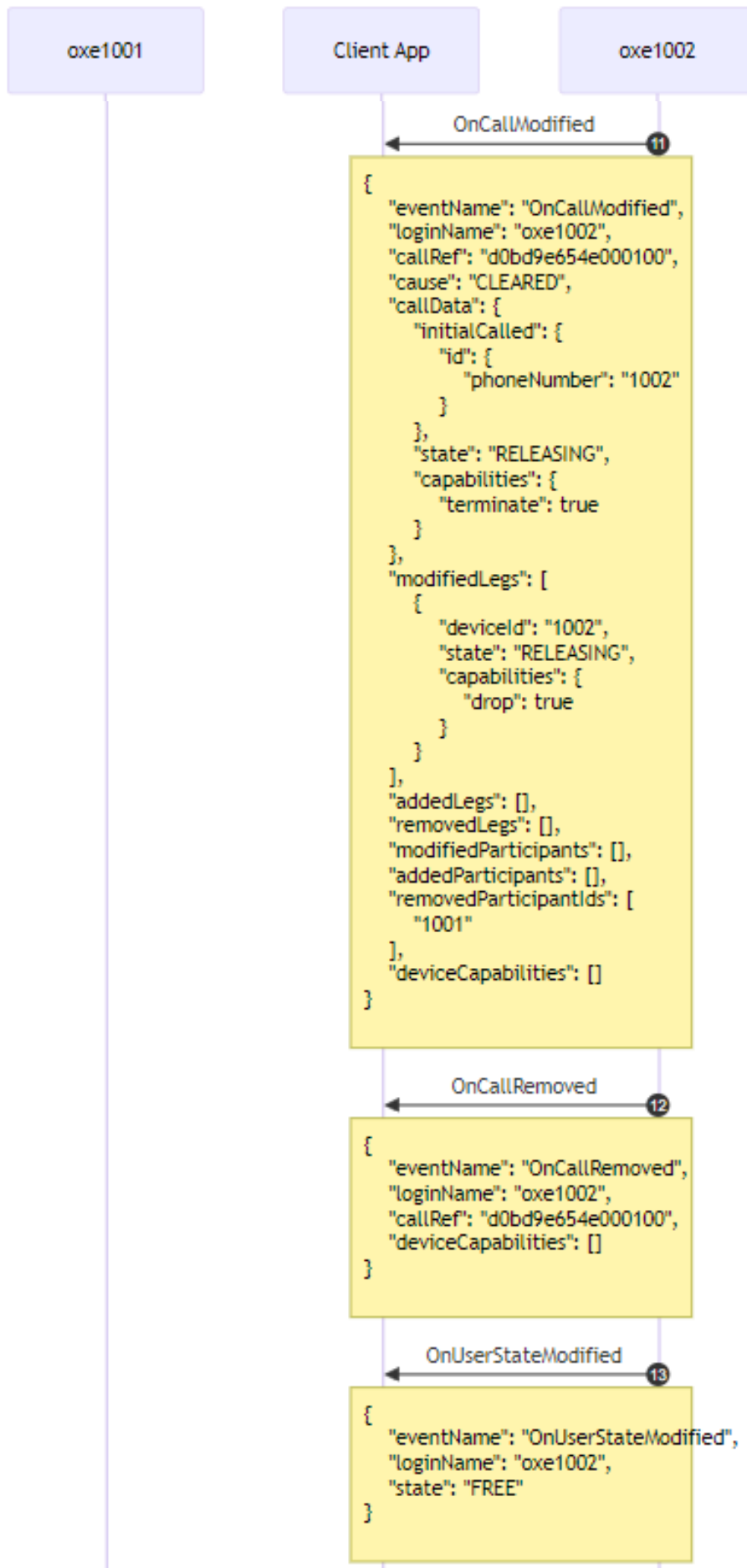




- oxe1001 releases the call and gets free.



- oxe1002 acknowledges call release and gets free.



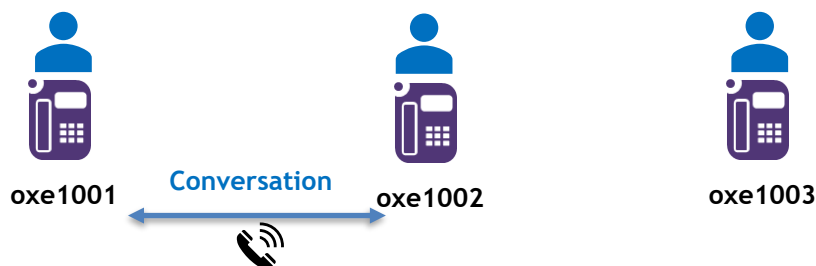
9 Use case: Monitoring unsupervised transfer call telephony events

The flow diagram illustrates a typical sequence of Telephony events generated by an unsupervised transfer call scenario involving three SIP OXE Extensions (oxe1001, oxe1002 and oxe1003).

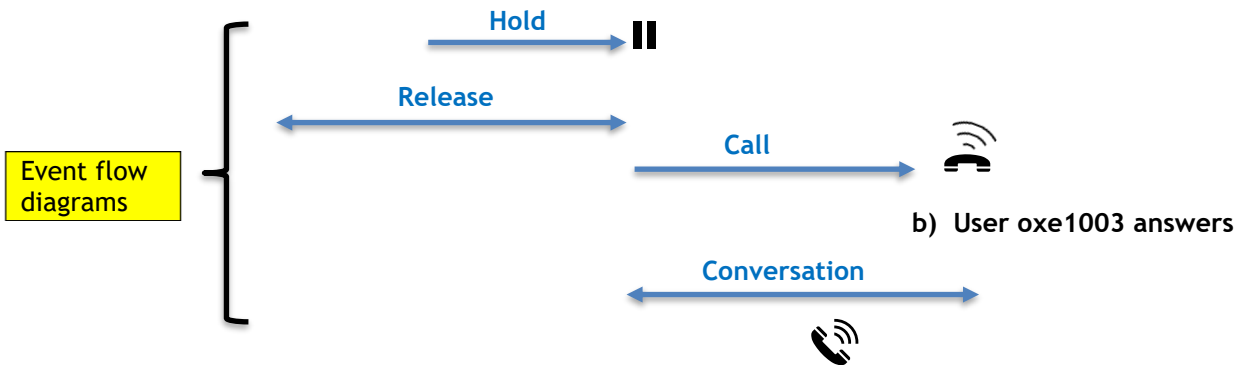
The two users oxe1001 and oxe1002 are already engaged in a conversation identified by CallRef (Call Reference)

The flow diagram describes the sequence of events issued when following actions are executed by users:

- a) User oxe1001 issues an unsupervised transfer to oxe1003
 - ➔ oxe1002 is put on hold
 - ➔ oxe1002 rings oxe1003
 - ➔ oxe1001 call is released
- b) User oxe1003 answers the call
 - ➔ oxe1002 and oxe1003 are in conversation



User oxe1001 executes an unsupervised transfer to oxe1003

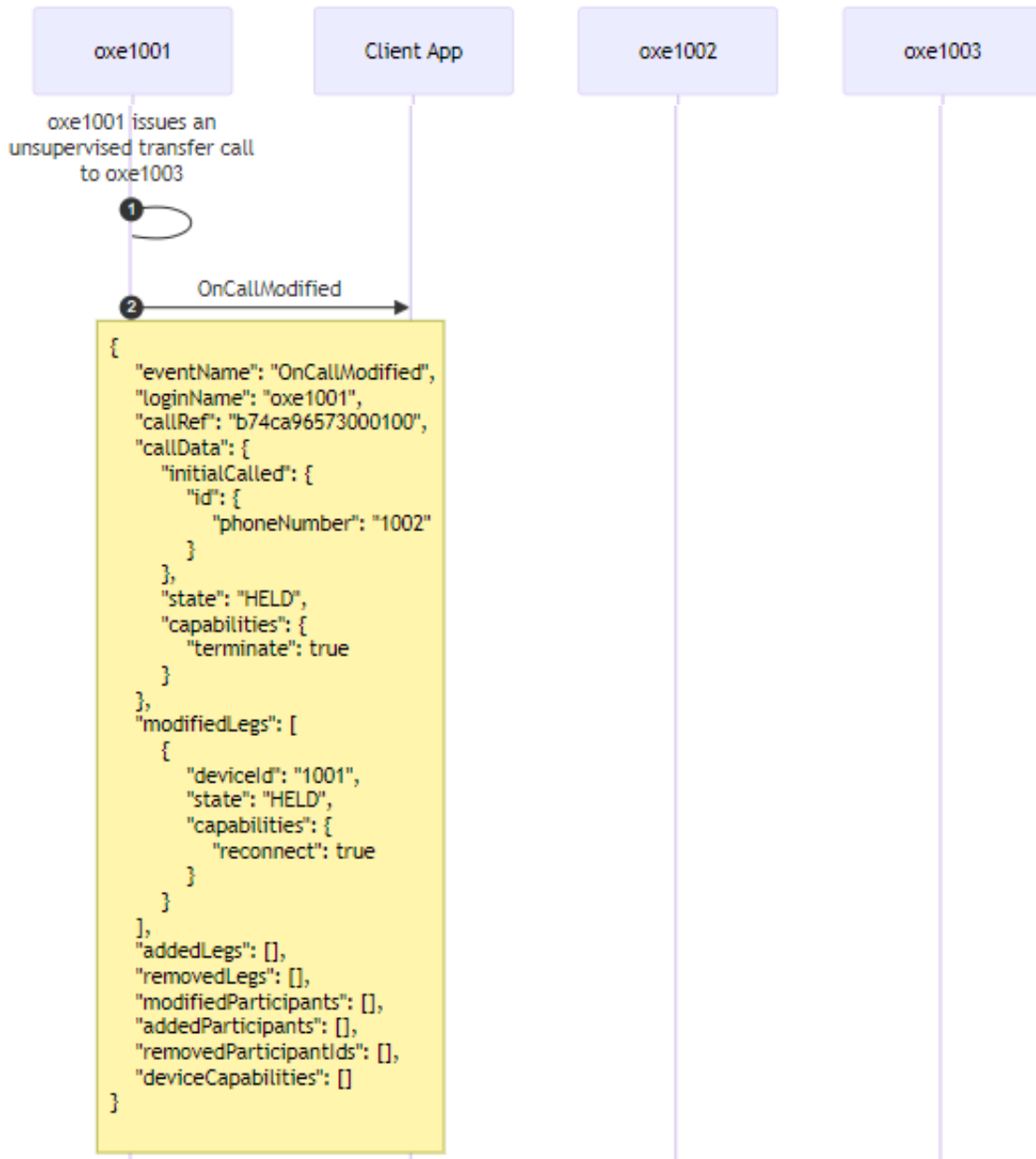


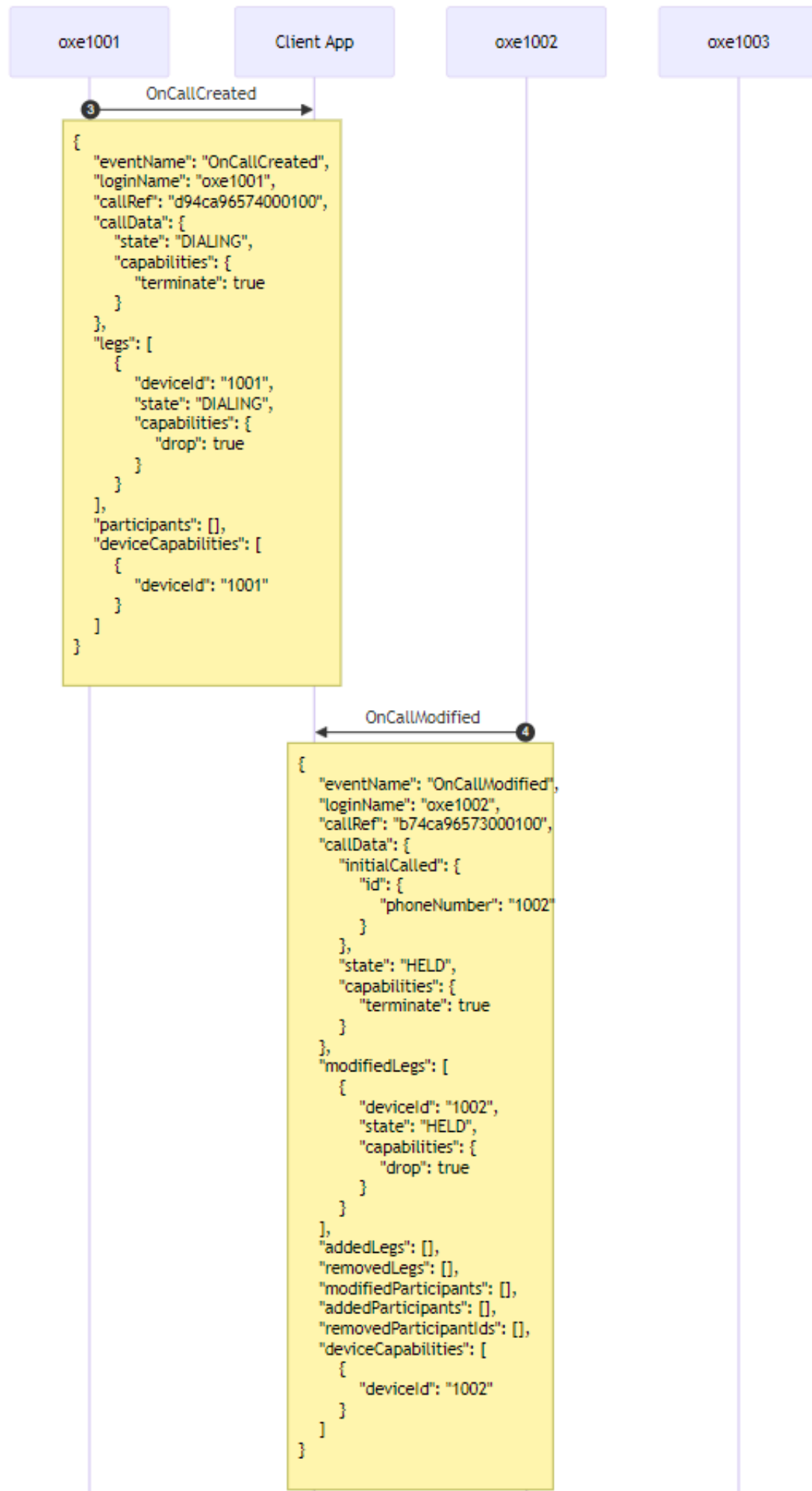
The Client Application must have subscribed to “telephony” category of events with an admin session(*) for the users involved in the use case (refer to “subscribing to services” section).

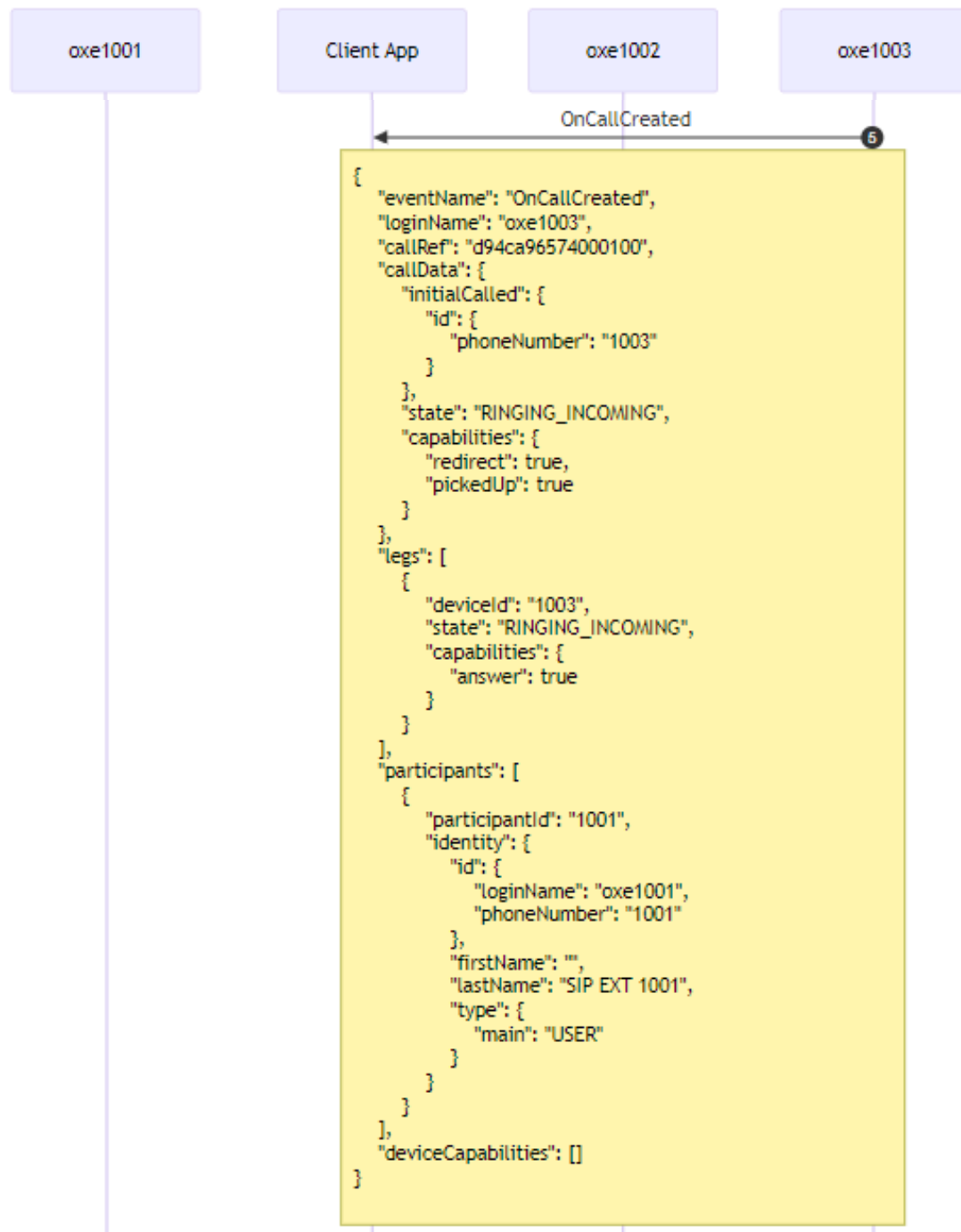
(*) Using an admin session allows the application to monitor several OXE users.

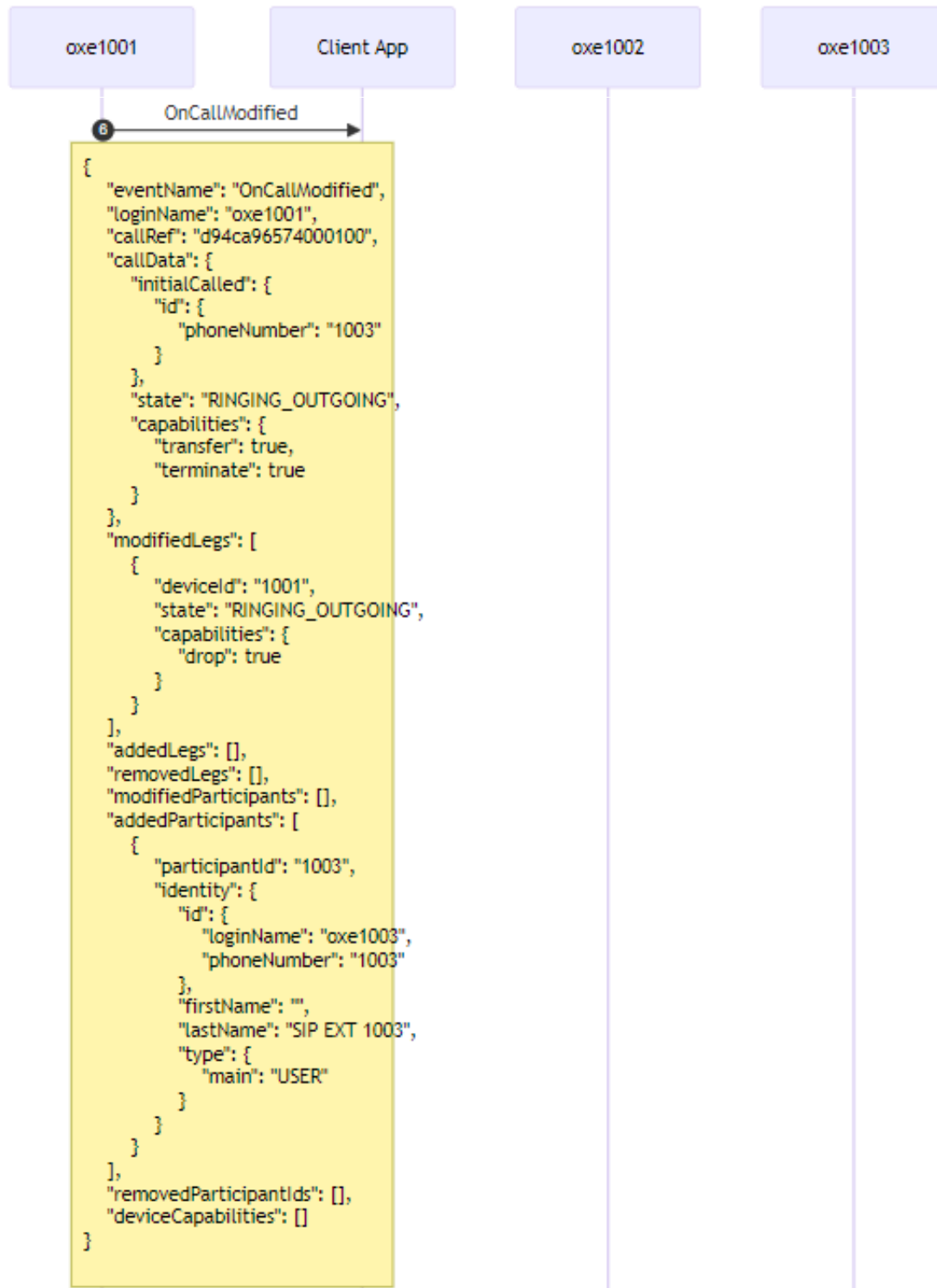
Please note that to simplify the representation of the event flow diagrams in the following pages, these diagrams only focus on the events received by the Client application using the O2G REST API to monitor user telephony activity of OXE. They do not depict OXE and O2G servers that generates and transmits these events.

oxe1001 and oxe1002 are in conversation
 Current active call oxe1001-oxe1002 is "b74ca96573000100"



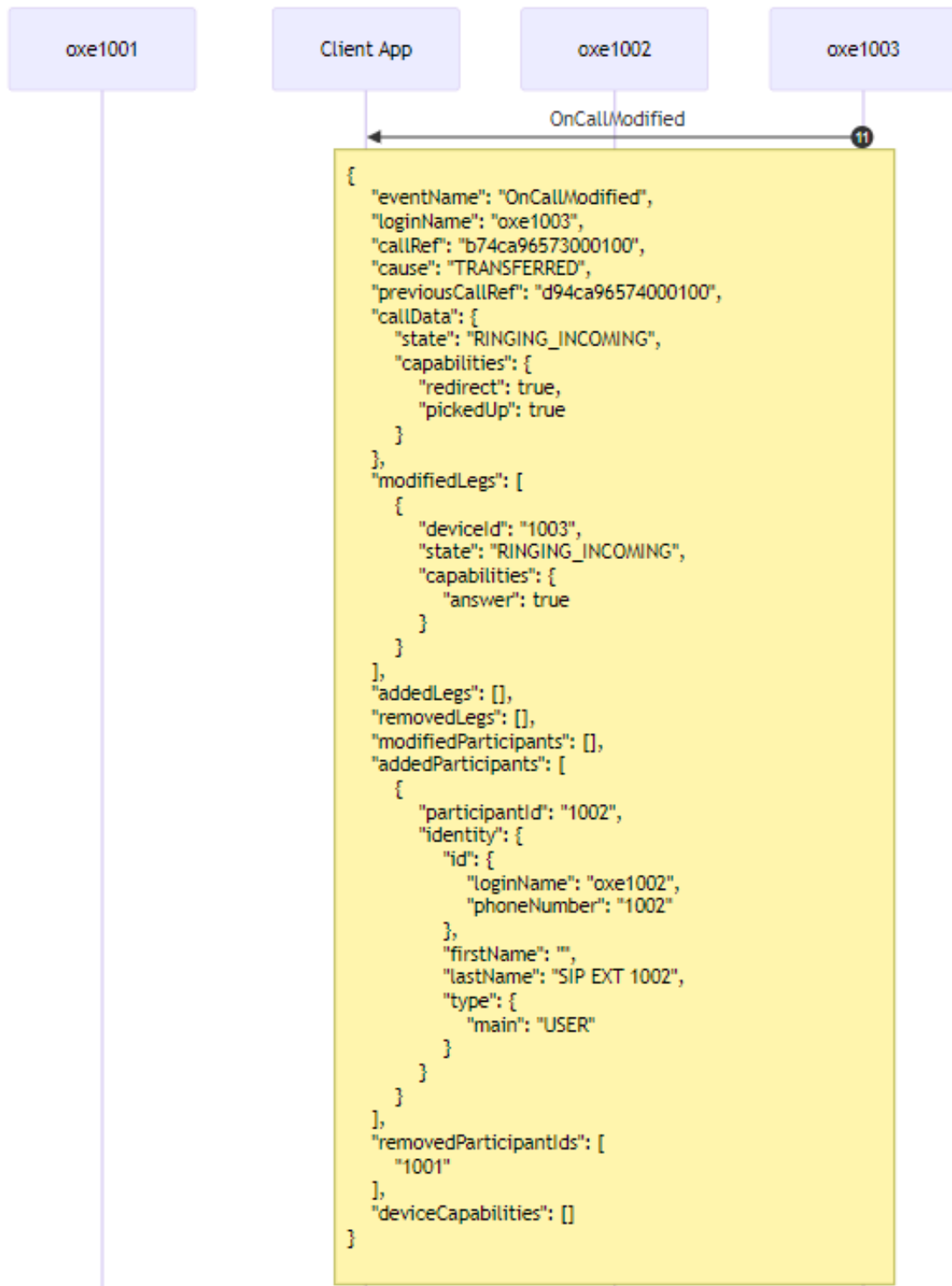


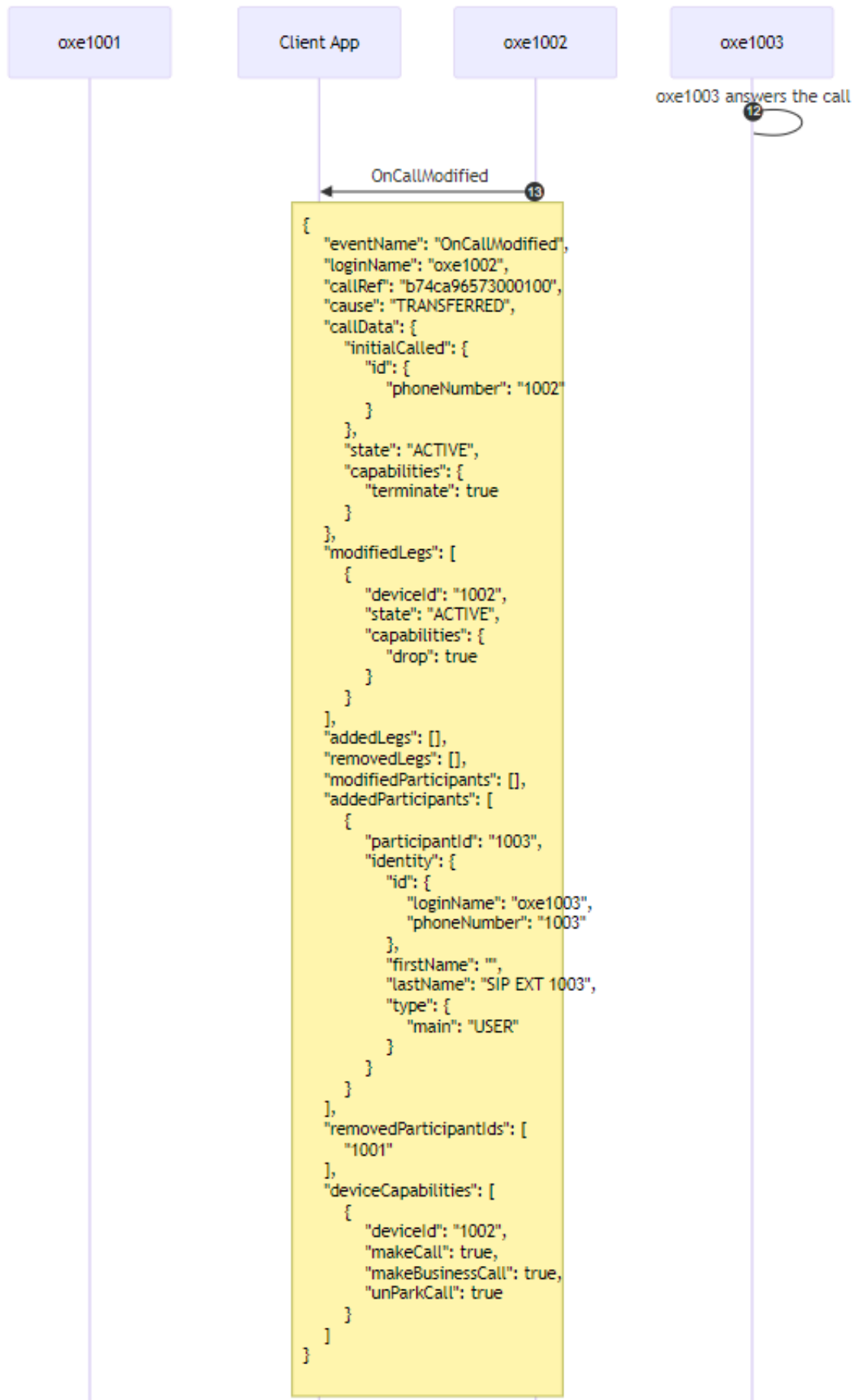


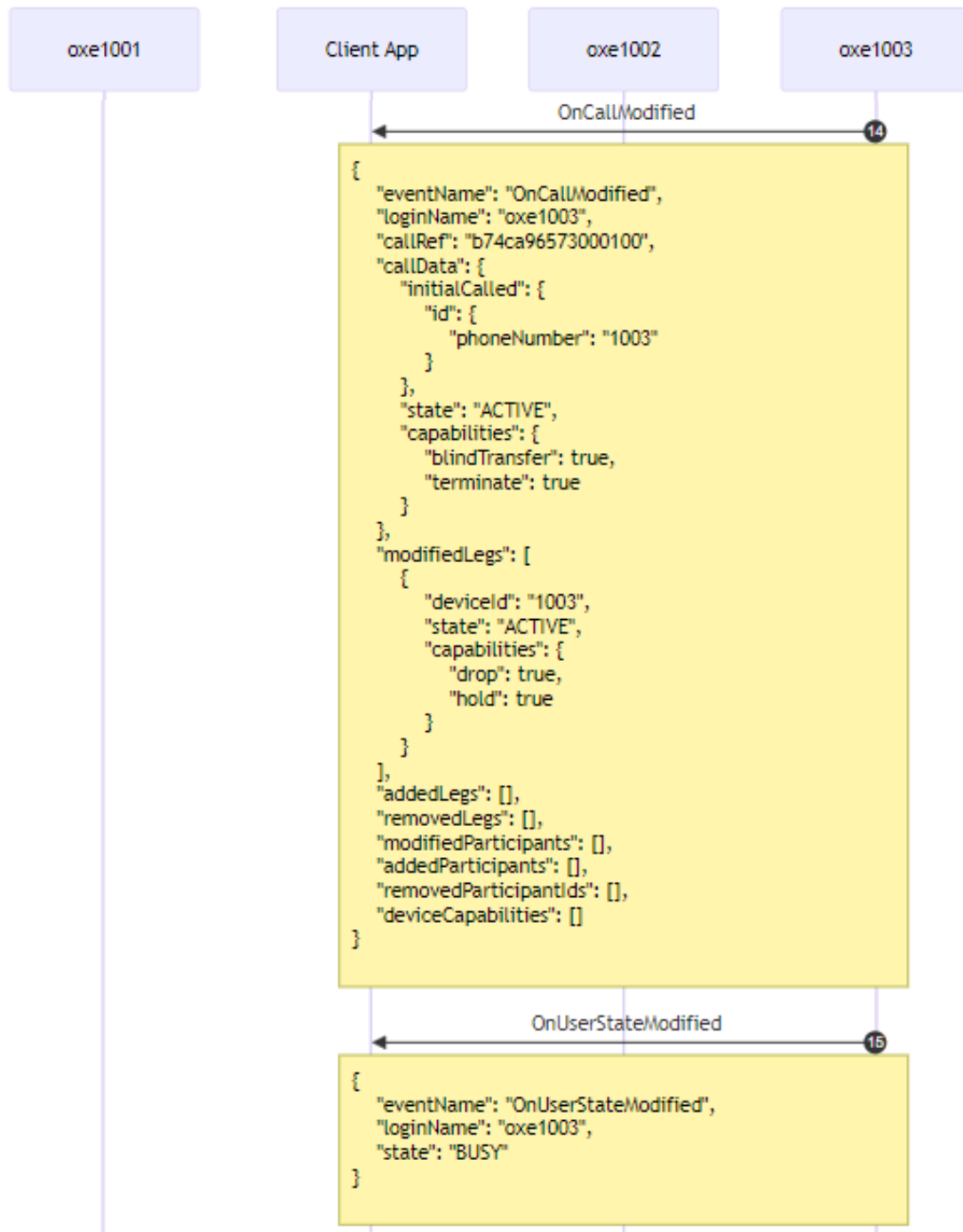


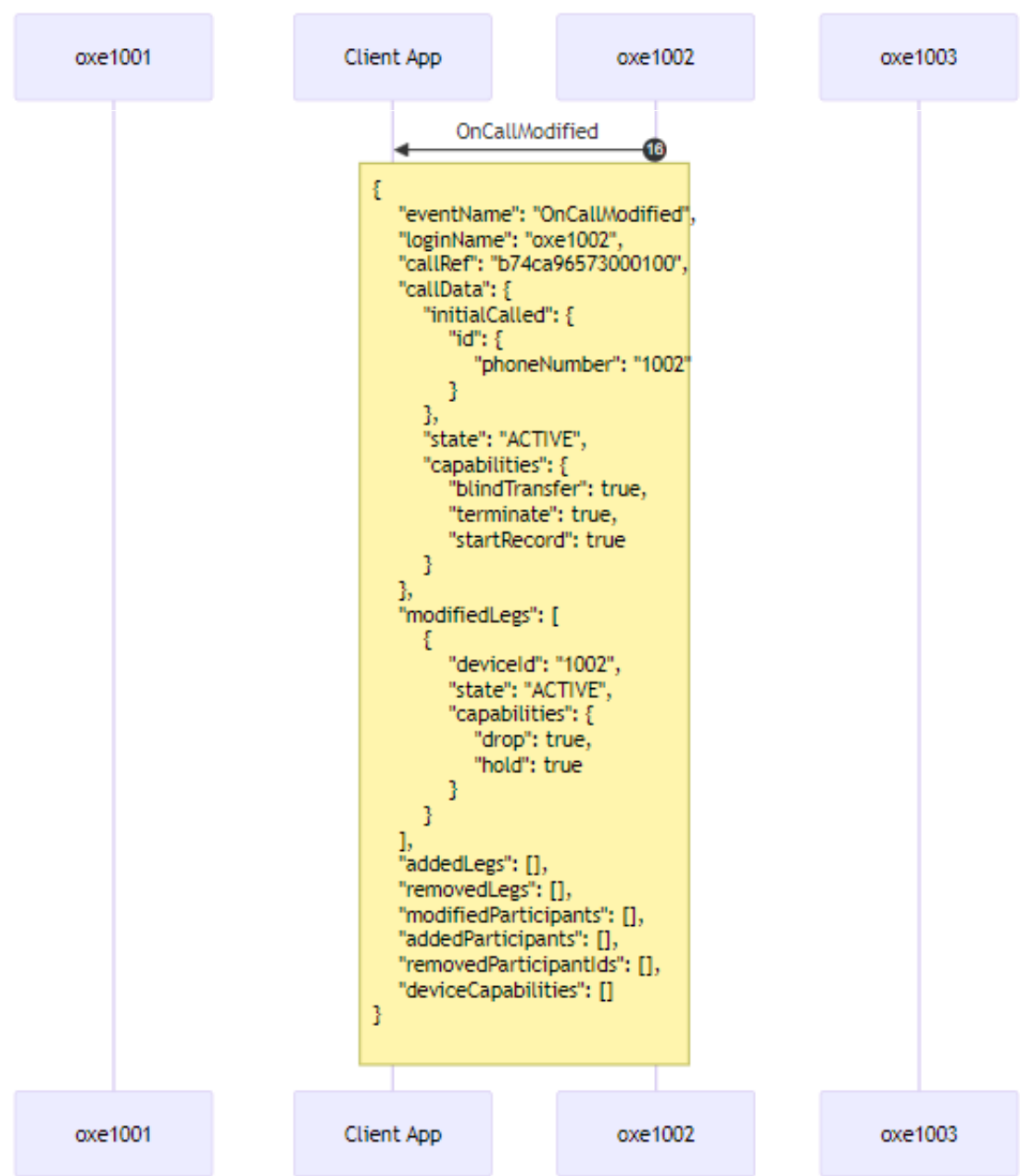












END OF DOCUMENT