Compere JSON External Control Protocol

User Guide

Trademark Information

Copyright Information

M710-21
www.7thsense.one
info@7thsense.one

Document Revision

Date	Document edition	Software version	Revision details	Author/Editor
22/12/2021	1		New release	Steve Warder, Nigel Scott
19/01/2022	2		Updates after review: • Added glossary • Updated sequence diagrams • Expanded protocol section • Added message encoding example • Tidied JSON examples and descriptions	Nigel Scott
04/02/2022	3		• Fixed JSON in "status" response example. • Added "unsolicited-response" details • Added details of basic and extended NAKs	Nigel Scott
15/08/2022	4	API v.4	• Added remaining commands • Renamed "status" to "get-system-status" • Updated paths to reflect JSON dot notation • Removed redundant Actions and Monitoring Data sections	Nigel Scott
04/01/2023	5	API v.5	• Added new messages • Added new result codes • Updated example paths • Update failure details for "set-role" and "join-project-group" • Add details of direct addressing	Nigel Scott
06/02/2023	6	API v.6	• Add details of object and property alias messages	Nigel Scott
24/02/2023	7	API v.7	• Added "max-response-length" argument • Added new result code definitions • Added new batch update messages • Replaced "Set Timeline Playhead Time" with "Set Timeline Position" and clarified time argument • Added new timeline marker messages • Added new remove object message • Extended Actor stats control message • Added SMPTE format for time specifications • Added details of input buffer overflow handling • Added new get available project groups message • Added "offline" as a role option • Clarified "save-project" functionality	Nigel Scott
22/03/2023	8	API v.8	• Removed "Get Asset Group" message as it is no longer required • Changed "Rivermax" references to generic "ST 2110"	Nigel Scott

			• Added new result code definitions:
			  o Duplicate UUID
			  o Failed to load clone
			  o Not yet implemented
			  o Failed to add timeline marker
			• Removed "unknown result" warning for messages that are now handled asynchronously:
			  o "Open Project"
			  o "Load Project"
			• Modified description of "Set Timeline Resource Duration" duration property to indicate correct format.	

26/05/2023	9	API v.9	• Changed "sdp-filepath" parameter to "sdp-file-path".
			• Replace references to "Feed Manager" with "Input Feeds" and "Output Feeds" as appropriate.
			• Changed conflicting "type" parameter name to "object-type" in:
			  o "Add Input To Input Feeds"
			  o "Add Output To Output Feeds"
			• Add missing "object-type" parameter in example response to "Add Output To Output Feeds".
			• Clarify that "linked-object-path" parameter is optional in:
			  o "Add Input To Input Feeds"
			  o "Add Output To Output Feeds"
			• Add "-" in parameters referencing ST 2110.
			• Clarify relative time formats are specified using SMPTE in:
			  o "Add Resource To Timeline"
			  o "Set Timeline Resource Start Time"
			  o "Trim Timeline Resource Begin"
			  o "Trim Timeline Resource End"
			  o "Set Timeline Position"
			  o "Add Timeline Marker"
			  o "Set Timeline Resource Duration"
			• Change "path" to "uuid" and clarify in "Add Resource To Timeline".
			• Move "Remove Object" next to add and clone object messages.
			• Add "Rewind Timeline" message.
			• Remove "Add Output To Output Feeds" message.
			• Remove "Remove Output From Output Feeds" message.
			• Change "time" argument to "duration" and clarify description for "Time Timeline Resource Begin" and "Trim Timeline Resource End" messages.
			• Added new result codes:
			  o Invalid device
			  o Failed to save project
				Nigel Scott

			○ Failed to remove input ○ Object is not an input ○ Object is not an output ○ Failed to remove output ○ Invalid object type ○ Object is not a viewport ○ Failed to remove viewport ○ Failed to add viewport to input ○ Object is not a timeline ○ Failed to remove object ○ Failed to trim timeline resource ○ Failed to add resource to timeline ○ Invalid duration specified • Remove scheduled time arguments for: ○ "Play Timeline" ○ "Pause Timeline" ○ "Stop Timeline" ○ "Buffer Timeline" • Update "Get Asset Group List" message: ○ Removed unused "path" argument ○ Updated example response • Remove "Buffer Timeline" command.	
16/06/2023	10	API v.10	• Added "Failover Pool Control" message. • Added "Trigger Failover Election" message. • Removed "Add Viewport To Output" message. • Removed "Remove Viewport From Output" message. • Added "Generate Status Notification" message.	Nigel Scott
26/07/2023	11	API v.11	• Fixed example for "Remove Viewport From Input" message • Added missing "uuid" to "Add Resource To Timeline" example • Added new result codes: ○ Invalid combination of parameters specified ○ Invalid marker name specified ○ Invalid frame number specified ○ Preselected failover election winner is not valid ○ Object name in path resolves to multiple objects ○ Failed to add input to input feeds ○ Failed to add output to output feeds ○ Failed to connect to WatchDog ○ WatchDog responded ○ Asset Logistics error • Added note to explain use of "direct-addresses" parameter	Nigel Scott

			• Extended Paths section to cover name based paths and direct addressing using UUIDs, names or aliases. • Updated message examples to use name based paths. • Removed unused "path" from "Add Resource To Timeline" example. • Fixed errors in Presets and Clones path sections. • Removed redundant section on Asset Database paths.	

08/09/2023	12	API v.12	• Added Juggler Control Messages section with:
			  o "Juggler Command"
			  o "Reset Juggler GPIO Trigger"
			• Added "Set Property Over Time" message.
			• Added "Clear Pending Set Property Over Time" message.
			• Deprecated "Failed to add output to output feeds" result code.
			• Added details and examples of drop frame and non drop frame SMPTE timecode formats.
			• Added section describing value property value formats for use when setting properties.
				Nigel Scott

15/09/2023	13	API v.13	• Added new 7thSense Internal section with details of "Tree Handling Control" message.	Nigel Scott
29/09/2023	14	API v.14	• Add section on setting the value of Selectable String properties.	Nigel Scott
04/12/2023	15	API v.15	• Removed "Set Role" message. • Added "Set Offline" message. • Added "Set Online" message. • Updated "Join Project Group" message (removed role argument). • Updated message examples in "Get Current Project Group" and "Get Project Group Details" message sections. • Added "Set Network Settings" message. • Added "Set TCP Connection Port" message. • Marked "Invalid Discovery Data" result code as deprecated. • Added "Failed to Update Network Settings" result code. • Added "execute-at-time" arguments to "Play Timeline" and "Stop Timeline" message examples. • Fixed "uuid" argument in "Add Resource To Timeline" message example.	Nigel Scott
01/01/2024	16	API v.16	• Added list of resettable render stats. • Added new result code: o Invalid Time Separator Error	Nigel Scott
24/01/2024	17	API v.17	• Added "Generate System Report" message. • Added "Configure Time Client" message.	Nigel Scott

18/03/2024	18	API v.18	• Fixed message name in example for "Configure Time Client". • Fixed "compere-discovery-port" argument name in "Failover Pool Control" message. • Added note that "Failover Pool Control" can be forwarded. • Document that "forward-request-to" is now mandatory in "set-property-over-time" messages. • Added new result code: o Invalid offset specified	Nigel Scott
16/04/2024	19	API v.19	• Added optional "layer-name" argument to "Add Resource To Timeline" message.	Nigel Scott
23/04/2024	20	API v.20	• Updated command in example for "Set Network Settings" • Updated "Logging Control" to correct "apply-to-logging-topics" setting.	Nigel Scott
21/08/2024	21	API v.21	• Clarify use of result codes (removed success and failure sections as these are not clear cut). • Fixed "broadcast-logging-port" argument name in "Broadcast Logging Control" message example. • Added section describing message logging. • Add "important" to list of available log levels in "Logging Control". • Fixed command name in "add-property-alias'' message example. • Added "forward-request-to" in "set-property-over-time" example. • Document the ".part" file extension behaviour when generating system reports.	Nigel Scott

Contents

1 Introduction

The purpose of this document is to provide an outline of the Compere JSON External Control Protocol for use with Compere. This document is targeted at engineers with basic JSON knowledge.

Aspects of this document are yet to be defined and will be updated in future revisions.

Note: "…" is used in the message examples below to indicate that further properties may be present. The properties shown are for example only.

Abbreviations

ACK: Acknowledged

JSON: JavaScript Object Notation

MAC: Media Access Control

NAK: Not Acknowledged

TCP: Transmission Control Protocol

UDP: User Datagram Protocol

Glossary

Netstring

This is a formatting method for strings which encodes the length of the string first. See https://en.wikipedia.org/wiki/Netstring for more information.

2 Logging

For development and debugging it can be useful to view messages being sent and received via JSON External Control. When enabled, the Netstring encoded messages are recorded in the Compere log file.

To enable logging, the minimum log level for the logging topic "ExternalControl" should be set to "verbose" using the "logging control" message (see 7.5.2).

To disable logging, set the minimum log level back to "note" (or higher).

3 Protocol

The Compere JSON External Control Protocol uses a simple text-based JSON formatted messaging protocol over TCP and UDP connections. Compere will ACK or NAK all messages received. Specific success or failure responses will then be sent for all acknowledged messages.

When using TCP, the default port for connections is 5584. When using UDP, the default send port is *TBD* and the default receive port is *TBD*. The ports can be configured in Compere's preferences.

To ensure that complete messages are parsed, they are encoded as Netstrings. This is a very simple protocol which prefixes the message content with its length. Each External Control message is a JSON object literal, encoded as a single Netstring. Data is read from the connection and buffered until a complete Netstring is received. The Netstring is then decoded as JSON and processed. If the Netstring is not valid, it will be ignored, a basic NAK will be generated, and the buffer will be searched for the start of another Netstring.

Sequences

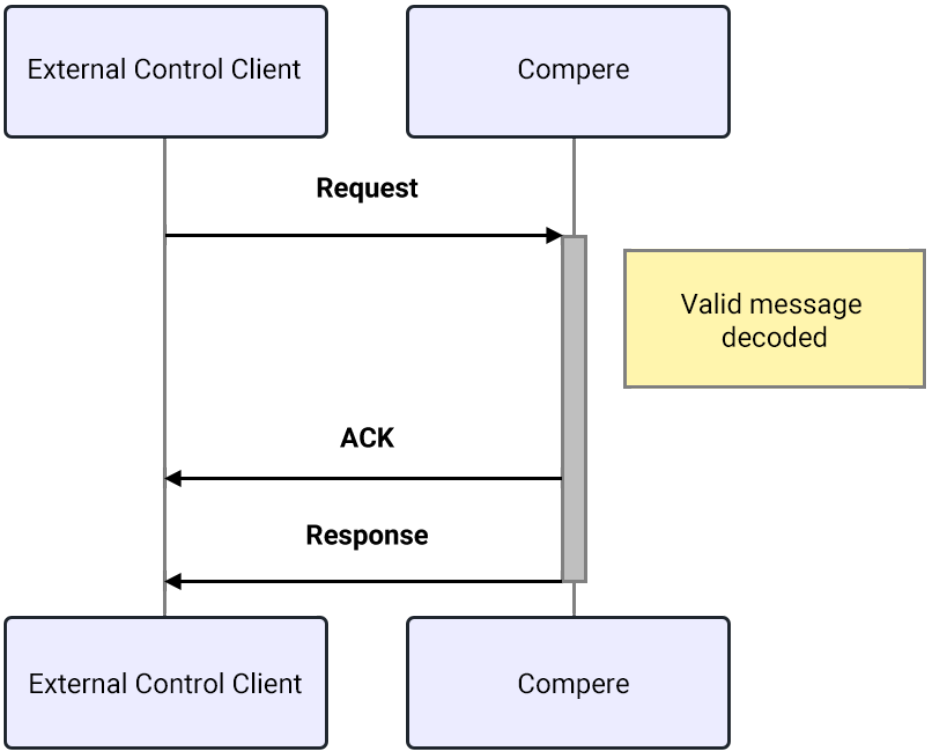

Figure 1 – Successful request handling

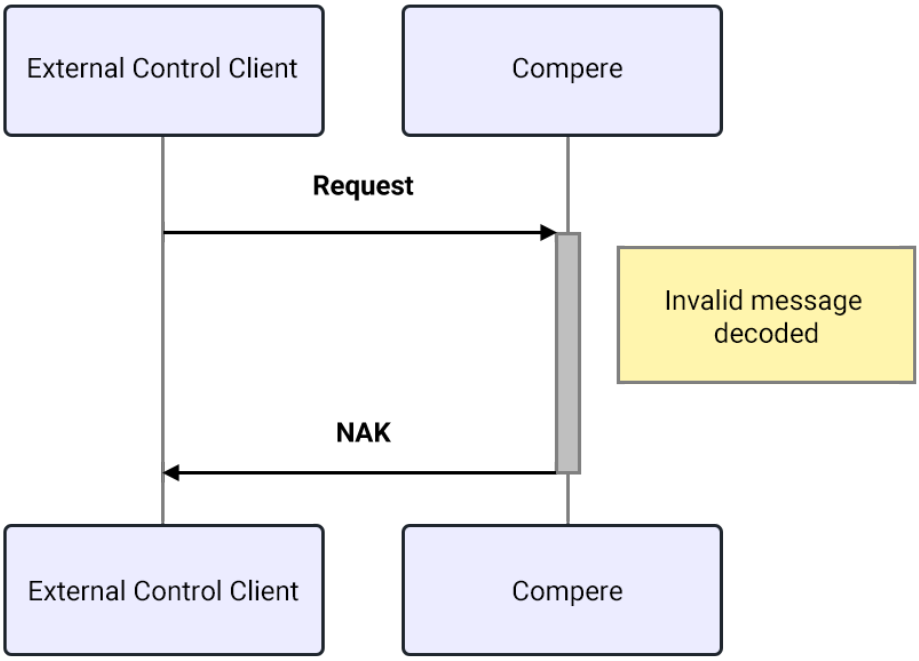

Figure 2 – Failed request handling

Message encoding example

JSON:

```
{"type":"request","cookie":1,"caller-id":"callerId","command":"get-system-status"}
```

Netstring:

```
82:{"type":"request","cookie":1,"caller-id":"callerId","command":"get-system-
status"},
```

Bytes transmitted (hexadecimal):

```
38 32 3a 7b 22 74 79 70 65 22 3a 22 72 65 71 75 65 73 74 22 2c 22 63 6f 6f 6b 69 65
22 3a 31 2c 22 63 61 6c 6c 65 72 2d 69 64 22 3a 22 63 61 6c 6c 65 72 49 64 22 2c 22
63 6f 6d 6d 61 6e 64 22 3a 22 67 65 74 2d 73 79 73 74 65 6d 2d 73 74 61 74 75 73 22
7d 2c
```

Note: The External Control interface has a limited input buffer (32kb). If this buffer is exceeded before messages can be processed, it will be cleared and a NAK will be generated. This should never happen in normal usage but is implemented as a safeguard.

4 Requests

All request messages in the Compere JSON External Control Protocol are specified using the following JSON format:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "<commandType>",
  "path": "path/to/object/in/tree",
  "params": {
    "property-1": "value",
    "property-2": "value"
    …
  },
}
```

Type	**String**	**Always "request"**
cookie	Integer	This is a message identifier that is returned in responses to messages. The uniqueness and range of this identifier is not policed. It is only used so that responses can be matched to originating messages. This will also be included in any multi-part responses.
Caller-id	String	This identifies the device that Compere is communicating with. It is used to identify the source of incoming messages and will be included in all responses.
Command	String	This is the command identifier for the message and must correspond to one of the commands listed in section 6.
Path[1]	String	A path to an object or property within the current project tree or status tree. Example: `"tree/GroupSet/Group/Juggler 10.100.150.94"` Other types of paths can also be specified (see 3.1).

params	Object	An object containing property/value pairs which identify properties and new values to set. Values are string-formatted representations.
		Example:
		`"params": {`
		`  "opacity": "90.0",`
		`  "fullscreen": "1"`
		`}`

direct-addresses[2]	Boolean	Some requests can generate responses which contain paths to objects within the Compere tree. This argument can be used to return these paths with direct addresses (UUIDs) instead of a full name path (see 3.1.1.2).
max-response-length	Integer	This can be used to specify the maximum size of the response message. If the response to a request is larger than this value, an error response will be generated instead, with details in the message argument.

[1] Some commands use an alternative main parameter instead of path, for example, "role" or "project-group".

[2] This parameter should be specified in the "params" block.

4.1 Paths

4.1.1 Object and Property Paths

Objects and properties within Compere are primarily addressed using name based paths.

The following sections of this document make use of this example partial data set (shown as XML for convenience):

```
<?xml version="1.0" encoding="UTF-8"?>
<tree UUID="956c9943237d415f81c0f34a83e0676e"
instantiatedBy="838db8e4f80540fc999eff5d31ff9368">
  <name value="tree"/>
  <TreeName UUID="16045eb32a7e4aae846876b2168969e4">
    <name value="My Project"/>
    <aliases value="0|"/>
  </TreeName>
  <GroupSet UUID="bbf9dbabfdcb4eb39be4ad32a76a931a">
    <name value="GroupSet"/>
    <aliases value="0|"/>
    <Group UUID="dff65ce661324aa3928431746436a9d9">
      <name value="Group"/>
      <aliases value="0|"/>
      <TestExternal UUID="adb29a4237ec4ca2a5bb68823de6f6c2">
        <name value="TestExternal 1"/>
        <aliases value="1|8:my-alias,"/>
        <enabled value="1"/>
        <state value="0"/>
        <direction value="3"/>
        …
```

4.1.1.1 Full Addressing

Objects and properties within Compere can be referenced by specifying a slash separated full path using the names of each parent object, terminated with either an object name or a property name.

The full path to the "TestExternal" object (highlighted in green in the example above) is:

```
"tree/GroupSet/Group/TestExternal 1"
```

The full path to the "direction" property (highlighted in yellow) is:

```
"tree/GroupSet/Group/TestExternal 1/direction"
```

Notes:

- *The names given to each object (highlighted in cyan) are used, **not** the type of the object.*
- *Objects must have names to be used in full addresses. Objects without names (where the name is blank/empty) must be directly addressed.*
- *Full addresses must resolve to a single item. Where a path resolves to multiple items, either object names must be changed so that paths are unique, or direct addressing must be used.*

- *Full addresses returned as part of responses to commands may not resolve to a single item and therefore cannot be guaranteed to be suitable for use in subsequent requests. Where a path resolves to multiple items, either object names must be changed so that paths are unique, or direct addressing must be used.*

4.1.1.2 Direct Addressing Using Object UUID

If the UUID of an object in the tree is known, this can be used to reference an object directly. This reduces the size of the required path and means the location of the object within the tree structure no longer needs to be considered.

The direct address of the "TestExternal" object (highlighted in green in the example above) using a UUID is:

```
"adb29a4237ec4ca2a5bb68823de6f6c2"
```

The direct address of the "direction" property (highlighted in yellow) of the object, using a UUID is:

```
"adb29a4237ec4ca2a5bb68823de6f6c2/direction"
```

4.1.1.3 Direct Addressing Using Names

If an object in the tree has a unique name (unique across the entire project), this can be used to reference the object directly. This reduces the size of the required path and means the location of the object within the tree structure no longer needs to be considered.

The direct address of the "TestExternal" object (highlighted in green in the example above) using its name is:

```
"TestExternal 1"
```

The direct address of the "direction" property (highlighted in yellow) of the object, using its name is:

```
"TestExternal 1/direction"
```

4.1.1.4 Direct Addressing Using Aliases

If an object in the tree has a registered alias, this can be used to reference the object directly. This reduces the size of the required path and means the location of the object within the tree structure no longer needs to be considered.

The direct address of the "TestExternal" object (highlighted in pink in the example above) using its alias is:

```
"my-alias"
```

The direct address of the "direction" property (highlighted in yellow) of the object, using its alias is:

```
"my-alias/direction"
```

4.1.2 Project File Paths

In some commands, the "path" argument can be used to refer to project files. In these cases, the value is a file path relative to the "projects" folder.

Project file names always use the extension ".prj" and this will be applied automatically.

Note: The root projects folder may depend on the current user profile.

4.1.3 Preset File Paths

In some commands, the "path" argument can be used to refer to preset files. In these cases, the value is a file path relative to the "presets" folder.

Preset file names always use the extension ".pre" and this will be applied automatically.

Note: The root presets folder may depend on the current user profile.

4.1.4 Clone File Paths

In some commands, the "path" argument can be used to refer to clone files. In these cases, the value is a file path relative to the "clones" folder.

Clone file names always use the extension ".clo" and this will be applied automatically.

Note: The root clones folder may depend on the current user profile.

4.2 Forwarded Requests

Some messages can be forwarded to other instances of Compere, using the "forward-request-to" argument. When messages are forwarded, result codes will always indicate success.

The following values can be used:

`""`	Targets all instances of Compere in the project group (including the instance receiving the command).
`"W612DNSC"`	Hostname – targets all instances of Compere running on the machine with the specified hostname.
`"01-23-45-67-89-AB"`	MAC address – targets all instances of Compere running on the machine with the specified MAC address.
`"2a7a8e9f-44c6-4d7c-bff3-f754ef7b6d1e"`	Application UUID – targets the specific instance of Compere.

4.3 Time Specifications

4.3.1 ISO8601

Some requests require times to be specified in ISO8601 format.

Examples:

`"2022-07-26T16:22:34Z"`	Full date and time to seconds accuracy, in extended format (with separators).
`"T143010.000005"`	Time to microseconds accuracy, in basic format (without separators).

4.3.2 SMPTE

Some requests require times to be specified in SMPTE format.

4.3.2.1 Drop Frame

Where a fractional frame rate is being used, SMPTE format timecodes must be specified using drop frame format, i.e., semicolon separated values.

Examples:

`"23;59;59;29"`	"HH;MM;SS;FF", i.e. 24 hour time, followed by a frame number, separated by semicolons.

4.3.2.2 Non Drop Frame

Where a integer frame rate is being used, SMPTE format timecodes must be specified using non drop frame format, i.e., colon separated values.

Examples:

`"23:59:59:29"`	"HH:MM:SS:FF", i.e. 24 hour time, followed by a frame number, separated by colons.

4.4 Property Value Formats

When setting properties (see 6.3.5), it is important that the correct string format is used in order that the value is parsed correctly. All values must be specified in quotes (as strings).

4.4.1 Numeric Types

The following numeric types are supported:

- int
- unsigned int (*can't be negative*)
- long long

- float
- double

Examples:

```
"-1234"
```

```
"3.1415"
```

```
"-0.002"
```

4.4.2 Boolean

Boolean values should be either "true" or "false".

4.4.3 Strings

No extra formatting is required for string values.

4.4.4 Numeric Vectors

Numeric vector property types are specified with the number of values and the list of values separated by a vertical bar. The list of values is semicolon separated.

Examples:

```
"3|1;2;3"
```

```
"5|0.2;-0.2;1.5;-3.0;6.0;999.99"
```

4.4.5 String Vectors

String vectors are specified with the number of values and the list of values separated by a vertical bar. The list of values comprises each string value converted to a Netstring concatenated with no extra separators.

Example:

```
"4|3:red,5:green,4:blue,6:yellow,"
```

4.4.6 Selectable Strings

Selectable Strings are specified with the current value and the property containing the alternative values separated by a vertical bar. The current value is specified as a Netstring. The property is specified by the UUID of the object that contains the property, and the property name separated by a forward slash.

If the referenced property is contained by the same object on which the property is being set, the UUID can be null (all zeros).

Note: The current value does not need to be in the list of alternative values.

Examples:

```
"6:orange,|c7690c07a3ac4b9683acee4bf3ba2958/string-vector-property"
```

i.e., set the current value to "orange" and get alternative values from the property "string-vector-property" in the object with UUID "c7690c07a3ac4b9683acee4bf3ba2958".

```
"3:red,|00000000000000000000000000000000/colours"
```

i.e., set the current value to "red" and get alternative values from the property "colours" in the same object.

4.4.6.1 Setting the current value only of a Selectable String property

When setting the value of a Selectable String, it is possible to modify just the current value of the property without changing the existing reference to the String Vector that provides the options. If the provided value does not contain a complete definition (as described above) of a serialized Selectable String, it will be treated as a new current value.

Note: When setting a Selectable String this way, the value should not be encoded as a Netstring.

Example:

Given an existing value of

```
"6:orange,|c7690c07a3ac4b9683acee4bf3ba2958/string-vector-property"
```

setting the value using the "set" command with a new value of

```
"red"
```

sets the value to

```
"3:red,|c7690c07a3ac4b9683acee4bf3ba2958/string-vector-property"
```

5 Message Acknowledgement

For every client request, Compere sends an ACK or NAK response.

An ACK is sent if the request is valid JSON format and the specified command is recognised, otherwise a NAK is sent.

Two types of NAK can be generated. A basic NAK is generated when an invalid Netstring is received, invalid JSON is decoded, or the required properties "caller-id" and "cookie" are missing. An extended NAK is generated when the request fails due to other missing parameters or other failures to process the request internally. The extended NAK includes the "caller-id" and "cookie" properties from the original request.

5.1 ACK

```
{
  "type": "ack",
  "cookie": 1,
  "caller-id": "<callerId>"
}
```

type	**String**	**Always "ack".**
Cookie	Integer	The cookie supplied in the originating message.
Caller-id	String	The caller-id supplied in the originating message.

5.2 Basic NAK

```
{
  "type": "nak",
  "message": "Invalid JSON."
}
```

type	**String**	**Always "nak".**
Message	String	A description of why the message has not been acknowledged. This would normally be due to an invalid message format or missing required properties.

5.3 Extended NAK

```
{
  "type": "nak",
  "cookie": 1,
  "caller-id": "<callerId>",
  "message": "Unsupported command."
}
```

type	**String**	**Always "nak".**
Cookie	Integer	The cookie supplied in the originating message.
Caller-id	String	The caller-id supplied in the originating message.
Message	String	A description of why the message has not been acknowledged. This would normally be due to an invalid message format or unknown command type.

6 Responses

Once a request has been successfully validated and acknowledged, Compere will try to perform the specified command. This will result in either one or more success response messages or an error response.

Initial responses to the originating request have a "type" property set to "response". Unsolicited responses (generated after a "register" request) have a "type" property set to "unsolicited-response".

All responses and unsolicited responses will include the "caller-id" and "cookie" properties that were supplied with the originating message.

Responses have the following format:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "next-sequence-id": 2,
  "ack-required": false,
  "result-code": 0,
  "message": "<descriptive message>",
  "data": {
    "path/to/object/in/tree": {
      "property-1": "1",
      "property-2": "true",
      "property-3": "value",
      …
    }
  }
}
```

type	**String**	**Always "response".**
Cookie	Integer	The cookie supplied in the originating message.
Caller-id	String	The caller-id supplied in the originating message.
Sequence-id	Integer	The identifier of this response. This allows the client to detect duplicate or missing responses within a determined sequence of responses.
Next-sequence-id	Integer	An optional identifier for the next response in a sequence of responses. If the next-sequence-id is not supplied, then this is the final response in the sequence.
Ack-required	Boolean	This indicates when an acknowledgement from the client is required.
Result-code	Integer	A predefined result code indicating the result of the request (see 5.1).
Message	String	A short message giving, for example, a description of the action performed or failure details.
Data	Object	A JSON object containing data relevant to the request.

Note: When an acknowledgement is required, and none is received, Compere will resend the response up to 3 times, with a delay of 3000ms between each message.

6.1 Result Codes

The following result codes are defined:

0	No error (see 5.1)
1	Request forwarded
2	Request blocked
3	Cannot be forwarded
4	Invalid path
5	Unknown property
6	Client not previously registered
7	Unknown Project
8	Failed to get script status
9	Failed to get system status
10	Unknown ID
11	System error
12	Partial success
13	Missing parameter
14	Max response length exceeded
15	Invalid tree
16	Not descendant tree
17	Failed to set properties
18	File already exists
19	Unknown task type
20	~~Invalid discovery data~~ *(deprecated)*
21	Failed to apply preset to objects
22	Failed to get available projects
23	Invalid time specified
24	Failed to get available presets
25	Disabled parameter
26	Invalid parameter value
27	Failed to add object alias
28	Failed to remove object alias
29	Invalid reboot type
30	Failed to add property alias
31	Unknown object
32	Alias already exists

33	Alias not found
34	Duplicate property
35	Name is not a valid identifier
36	Name is not valid XML
37	Failed to remove property alias
38	Failed to create new project
39	Failed to open project
40	Duplicate UUID
41	Failed to load clone
42	Not yet implemented
43	Failed to add timeline marker
44	Invalid device
45	Failed to save project
46	Failed to remove input
47	Object is not an input
48	Object is not an output
49	Failed to remove output
50	Invalid object type
51	Object is not a viewport
52	Failed to remove viewport
53	Failed to add viewport to input
54	Object is not a timeline
55	Failed to remove object
56	Failed to trim timeline resource
57	Failed to add resource to timeline
58	Invalid duration specified
59	Invalid combination of parameters specified
60	Invalid marker name specified
61	Invalid frame number specified
62	Preselected failover election winner is not valid
63	Object name in path resolves to multiple objects
64	Failed to add input to input feeds
65	~~Failed to add output to output feeds~~ (*deprecated*)
66	Failed to connect to WatchDog
67	WatchDog responded
68	Asset Logistics error

69	Failed to update network settings
70	Invalid time separator error
71	Invalid offset specified

7 Messages

7.1 Project and Application Messages

7.1.1 New Project

The "new-project" command is used to create a new project file on the current server's file system.

The "path" argument is treated as a project filename (see 3.1.2).

The optional "overwrite" flag can be used to force a new project file to overwrite an existing project file.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "new-project",
  "path": "my-projects/new-project.prj",
  "params": {
    "overwrite": "true"
  }
}
```

7.1.2 Open Project

The "open-project" command is used to load a project file from the current server's file system.

The "path" argument is treated as a project filename (see 3.1.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "open-project",
  "path": "my-projects/project.prj"
}
```

7.1.3 Save Project

The "save-project" command is used to save a project via the server. This command is a request that may not be completed (see 5.1).

The optional "path" argument is treated as a project filename (see 3.1.2). If the "path" argument is not provided, the project file will be saved with its existing name.

If the "path" argument is provided and the file already exists, the command will fail, unless the optional "overwrite" flag is specified.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "save-project",
  "path": "my-projects/current-project.prj",
  "params": {
    "overwrite": "true"
  }
}
```

7.1.4 Export Project

The "export-project" command is used to request a targeted instance of the Compere to export the project to a local file. This command is a request that may not be completed (see 5.1).

The "path" argument is treated as a project filename (see 3.1.2).

If the file already exists, the command will fail, unless the optional "overwrite" flag is specified.

The "exclude" argument is used to specify a comma separated list of tree nodes which should be excluded from the export.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "export-project",
  "path": "my-projects/exported-project.prj",
  "params": {
    "forward-request-to": "W612DNSC",
    "overwrite": "true",
    "exclude": "Selection"
  }
}
```

7.1.5 Set Offline

The "set-offline" command is used to take an instance (or instances) of Compere temporarily out of a project group.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Note: Setting an instance offline will prevent any further indirect external control with the application.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-offline",
  "params": {
    "forward-request-to": "8c492218-c484-4186-823b-c061b8bf87da"
  }
}
```

7.1.6 Set Online

The "set-online" command is used to bring an instance of Compere that is offline, back into the project group.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Note: Although forwarding this message is supported, in practice it is not currently possible to bring an offline instance of Compere back online in this way.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-online",
  "params": {
    "forward-request-to": "8c492218-c484-4186-823b-c061b8bf87da"
  }
}
```

7.1.7 Join Project Group

The "join-project-group" command is used to move an instance of Compere into another project group.

The "project-group" argument is used to specify the project group to join.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "join-project-group",
  "project-group": "my-project",
  "params": {
    "forward-request-to": "A8-64-F1-6A-01-08"
  }
}
```

7.1.8 Get Current Project Group

The "get-current-project-group" command retrieves the details of the current project group.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-current-project-group"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "name": "Bank A"
  }
}
```

7.1.9 Get Project Group Details

The "get-project-group-details" command retrieves host information about a selected project group.

The optional "project-group" argument can be used to retrieve details of a specific project group. If not, specified, the details of the current project group are returned.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-project-group-details",
  "project-group": "my-project"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "name": "My project",
    "children": [
      {
        "member": {
          "hostname": "W652DNSC",
          "ip-address": "192.168.0.1",
          "operating-system": "Windows 10",
          "role": "server",
          "uuid": "04bfeb83-4775-4186-8e6d-b2b4afb52d27",
          "instance-type": "GUI",
          "colour": "ff438200",
          "application-version": "v1.0.7 B (52a9dce)",
          "selected-nic": "Ethernet 2",
          "online": "true",
          "score": "10"
        }
      },
      {
        "member": {
          "hostname": "LN1233JH",
          "ip-address": "192.168.0.2",
          "operating-system": "Linux",
          "role": "client",
          "uuid": "e47ca628-8fcb-42f7-8c26-37b5bac416ee",
          "instance-type": "Juggler",
          "colour": "ff438200",
          "application-version": "v1.0.7 B (52a9dce)",
          "selected-nic": "eth1",
          "online": "true",
          "score": "0"
        },
        ...
      ]
    }
  }
}
```

7.1.10 Close

The "close" command closes an instance of Compere.

The optional "reason" argument can be used to provide a textual reason for the closure, which may be logged before the application is shut down.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "close",
  "params": {
    "reason": "Testing"
  }
}
```

7.1.11 Reboot

The "reboot" command performs a hard or soft reboot of a machine hosting an instance of Compere.

The "reboot-type" argument is used to specify the type of reboot:

- "hard" (typically a power reset)
- "soft" (typically a software restart)

The optional "reason" argument can be used to provide a textual reason for the reboot, which may be logged before the device is rebooted.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "reboot",
  "reboot-type": "hard",
  "params": {
    "reason": "Testing"
  }
}
```

7.1.12 Clear Tasks

The "clear-tasks" command can be used to clear all or failed tasks from the task manager task list.

The optional "task-type" argument can be used to specify the type of tasks to be cleared:

- "all"
- "failed"

If not specified, the command will apply to all tasks.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "clear-tasks",
  "params": {
    "task-type": "failed"
  }
}
```

7.1.13 Script

The "script" command runs a script on the server. Specified properties are passed to the script. If the script is already running the properties are updated.

If the "path" argument is empty or not present, the status of any currently running scripts is returned.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "script",
  "path": "my-scripts/script",
  "params": {
    "property-1": "value1"
    "property-2": "value2"
    ...
  }
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "scriptName-1": "status1",
    "scriptName-2": "status2",
    "scriptName-3": "status3"
    ...
  }
}
```

7.1.14 Get Devices

The "get-devices" command returns a list of the devices in the current project.

For each device, the type, name, path and UUID is returned.

Responses to this command can contain direct addresses (see 3.1.1.2) if the optional "direct-addresses" argument is specified in the request.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-devices"
}
```

Example response

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "devices": {
    "children": [
      {
        "device": {
          "type": "Actor",
          "uuid": "adf86e48-2282-4259-8ef7-6f8e5b6e782e",
          "path": "tree/ResourceSet/Actor",
          "name": "Actor"
        }
      },
      {
        "device": {
          "type": "juggler",
          "uuid": "4c935173-f83b-41c2-87eb-31951d5f2766",
          "path": "tree/GroupSet/Group/Juggler 10.100.150.94",
          "name": "Juggler 10.100.10.94"
        }
      },
      {
        "device": {
          "type": "juggler",
          "uuid": "996f7320-2937-4aaa-a62b-40a6abcf5061",
          "path": "tree/ResourceSet/Virtual Juggler WarpBlend",
          "name": "Virtual Juggler WarpBlend"
        }
      }
    ]
  }
}
```

7.1.15 Get Available Projects

The "get-available-projects" command returns a list of the projects which can be opened by Compere.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-available-projects"
}
```

Example response

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "projects": {
    "children": [
      {
        "project": {
          "filename": "my-projects/test-project.prj"
        }
      },
      {
        "project": {
          "filename": "my-projects/my-latest-project.prj"
        }
      },
      {
        "project": {
          "filename": "default-project.prj"
        }
      }
    ]
  }
}
```

7.1.16 Get Available Project Groups

The "get-available-project-groups" command returns a list of the project groups which are available to Compere.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-available-project-groups"
}
```

Example response

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "project-groups": {
    "children": [
      {
        "project-group": {
          "name": "Default"
        }
      },
      {
        "project-group": {
          "name": "Test"
        }
      },
      {
        "project-group": {
          "name": "Temporary"
        }
      }
    ]
  }
}
```

7.1.17 Failover Pool Control

The "failover-pool-control" command allows settings related to failover to be modified. The following settings can be changed:

- "score" (an unsigned integer value)
- "heartbeat-cadence" (milliseconds)
- "election-duration" (milliseconds)
- "initial-grace-period" (seconds)
- "heartbeat-port" (a valid port number)
- "compere-discovery-port" (a valid port number)

This command can optionally be forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "failover-pool-control",
  "params": {
    "heartbeat-cadence": "2000"
    "score": "50"
  }
}
```

7.1.18 Trigger Failover Election

The "trigger-failover-election" command requests a leadership election in order to select a server for the current project group.

The optional "preselected-winner" argument can be used to explicitly choose a winner (which much be a member of the current project group).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "trigger-failover-election",
  "params": {
    "preselected-winner": "W98F3NSC"
  }
}
```

7.2 Status Messages

7.2.1 Get System Status

The "get-system-status" command is used to retrieve the overall system status. This includes overall project status, peer data and task status.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-system-status"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "task-summary": {
      "total": 6,
      "failed": 1,
      "pending": 3,
      "complete": 2,
      "percent-complete": 34
    },
    "tasks": [
      {
        "name": "task-1",
        "status": "failed",
        "percent-complete": 100
      },
      {
        "name": "task-2",
        "status": "complete",
        "percent-complete": 100
      },
      {
        "name": "task-3",
        "status": "in-progress",
        "percent-complete": 56
      },
      ...
    ],
    "project-status": {
      "project-path": "path/to/project/file.prj",
      "default-project": "default project name"
    },
    "network": {
      "comms-mode": "server",
      "peers": [
        {
          "name": "peer-1",
          "unique-id": "<id>",
          "aux-data": "<auxData>",
          "user-data": "<userData>"
        },
        {
          "name": "peer-2",
          "unique-id": "<id>",
          "aux-data": "<auxData>",
          "user-data": "<userData>"
        },
        ...
      ]
    }
  }
}
```

Note: "comms-mode" will be either "server", "client" or "off".

7.2.2 Get Status

The "get-status" command returns the summarized overall status of a device or devices (derived from the status property of the device(s) in the status tree).

Status will have one of the following values:

`"ok"`	No current issues
`"unknown"`	Non-responsive, or unable to determine status
`"critical"`	Loss of functionality
`"degraded"`	Issues detected, but still operational
`"warning"`	No operational issues, but potential issues have been detected
`"disabled"`	Not currently active (undergoing maintenance, for example)

The optional "depth" argument can be used to limit the number of levels which are reported from below the object referred to by the "path" argument.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-status",
  "path": " path/to/device/in/status/tree",
  "params": {
    "depth": "1"
  }
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "path": "path/to/device/in/status/tree",
    "status": "warning"
  }
}
```

7.2.3 Get Extended Status

The "get-extended-status" command returns the extended status of a device or devices.

Details are device specific.

The optional "depth" argument can be used to limit the number of levels which are reported from below the object referred to by the "path" argument.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-extended-status",
  "path": "path/to/device/in/status/tree",
  "params": {
    "depth": "3"
  }
}
```

7.3 Project Tree Messages

7.3.1 Add Object

The "add-object" command allows objects to be added to the project tree.

The "path" argument is used to specify the parent object for the new object.

The "type" argument specifies the type of the new object.

The "name" argument specified the name of the new object.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-object",
  "path": "path/to/parent/in/tree",
  "params": {
    "type": "juggler",
    "name": "My Juggler"
  }
}
```

7.3.2 Clone Object

The "clone-object" command is used to clone an object including its children.

The "path" argument is used to specify the object in the project tree to be cloned.

The "target" argument specifies the path of the parent object for the new object.

The optional "new-instance-name" and "new-instance-uuid" arguments can be used to explicitly set new values for the cloned object.

Note: Cloned objects will always have new UUIDs, unless specifically set using the "new-instance-uuid" argument.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "clone-object",
  "path": "path/to/existing/object/in/tree",
  "params": {
    "target": "path/to/parent/object/in/tree",
    "new-instance-name": "My Cloned Juggler",
    "new-instance-uuid": "362a951a-22ae-4155-ae40-14c0c399e0e3"
  }
}
```

7.3.3 Remove Object

The "remove-object" command allows objects to be removed from the project tree.

The "path" argument specifies the path to the new object.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-object",
  "path": "path/to/object/in/tree"
}
```

7.3.4 Get

The "get" command message is a one time request of a value tree at a given path.

The optional "recurse" Boolean argument can be used to specify whether the response should also include all child nodes of the value tree. When not specified, children are included.

Responses to this command can contain direct addresses (see 3.1.1.2) if the optional "direct-addresses" argument is specified in the request.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get",
  "path": "path/to/object/in/tree",
  "params": {
    "recurse": "false"
  }
}
```

7.3.5 Set

The "set" command message is a request to set new values for a value tree path.

The "params" argument should include one or more properties for the object specified by the "path" argument.

The format of a property value depends on its type. See for further details.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set",
  "path": "path/to/object/in/tree",
  "params": {
    "opacity": "1.0",
    "fullscreen": "true"
    ...
  }
}
```

7.3.6 Register

The "register" command is used to register for notifications of value changes for a value tree within the current project.

On initial successful subscription, the client will receive an initial response that contains the current data value/s for the specified project path.

The client will then receive subsequent unsolicited responses whenever a value within the specified tree changes. These will continue until the client disconnects or the client deregisters using the "deregister" message.

Responses (including unsolicited responses) to this command can contain direct addresses (see 3.1.1.2) if the optional "direct-addresses" argument is specified in the request.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "register",
  "path": "path/to/object/in/tree"
}
```

Subsequent responses will always include the original message "caller-id" and "cookie".

Where responses are sent for descendant tree changes the data will only include the JSON representation of the modified descendant tree.

Example:

If the client registers to receive updates to the path:

```
"path/to/object/in/tree"
```

and subsequent changes occur in:

```
"path/to/object/in/tree/child"
```

Example unsolicited response:

```
{
  "type": "unsolicited-response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "message": "Project data updated.",
  "data": {
    "path/to/object/in/tree/child": {
      "property-1": "1",
      "property-2": "true",
      "property-3": "value"
      ...
    }
  }
}
```

7.3.7 Deregister

The "deregister" command deregisters the client from tree value change notifications for a previously registered project path.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "deregister",
  "path": "path/to/object/in/tree"
}
```

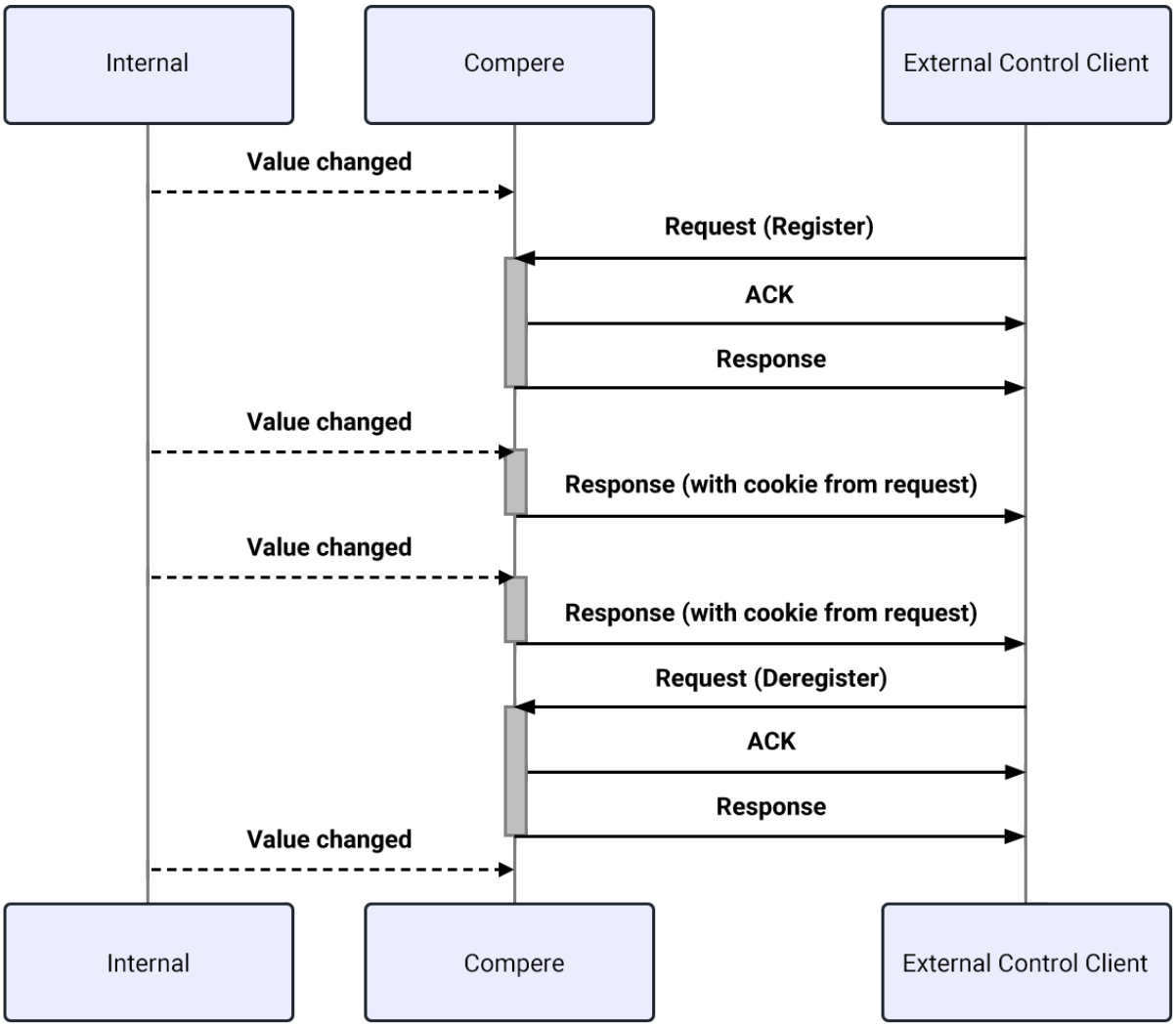

Figure 3 – Register and Deregister request handling

7.3.8 Recall Preset

The "recall-preset" command recalls a preset from the server filesystem. This command is a request that may not be completed (see 5.1).

The "path" argument is treated as a preset filename (see 3.1.3).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "recall-preset",
  "path": "my-presets/preset.pre"
}
```

7.3.9 Apply Preset To Objects

The "apply-preset-to-objects" command applies a preset file to one or more objects. This command is a request that may not be completed (see 5.1).

A typical usage is for copying the configuration of one Juggler to another.

The "path" argument is treated as a preset filename (see 3.1.3).

The "target" argument can be used to specify the path of a specific object to which the preset should be applied.

The "targets" argument can be used to specify a comma separated list of object names to apply the preset to. All objects which match a specified name will be considered as targets.

At least one of the "target" or "targets" arguments must be included in the request.

The command will fail if:

- The preset contains more than one root object.
- The preset object type does not match the target object type.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "apply-preset-to-objects",
  "path": "my-presets/preset.pre",
  "params": {
    "targets": "Juggler,Actor"
  }
}
```

7.3.10 Load Clone

The "load-clone" command creates new instances of objects defined in a clone file (and adds them to top level group object). This command is a request that may not be completed (see 5.1).

The "path" argument is treated as a clone filename (see 3.1.4).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "load-clone",
  "path": "my-clones/clone.clo"
}
```

7.3.11 Get Asset Group List

The "get-asset-group-list" command retrieves the asset group list for a given folder.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-asset-group-list"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "asset-groups": {
      "children": [
        {
          "entry":
            "name": "7thFlyAnimationHD422",
            "uuid": "6c9ffea8ada5bad59f53bbb3930024e9",
            "asset-count": "1",
            "asset-classes": "StillSequence",
            "duration": "1583333"
          }
        },
        {
          "entry": {
            "name": "dillama_2min",
            "uuid": "cc602af77fc2f00ff7d7ab7153001d31",
            "asset-count": "1",
            "asset-classes": "StillSequence",
            "duration": "3683333"
          }
        },
        {
          "entry": {
            "name": "testLoop7",
            "uuid": "cc602af77fc2f00ff7d7ab71530041bd",
            "asset-count": "1",
            "asset-classes": "StillSequence",
            "duration": "1000000"
          }
        }
      ]
    }
  }
}
```

7.3.12 Add Timeline

The "add-timeline" command adds a timeline to the project.

The optional "count" argument can be used to add more than one timeline if required.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-timeline",
  "params": {
    "count": "2"
  }
}
```

7.3.13 Remove Timeline

The "remove-timeline" command removes a timeline from the project.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-timeline",
  "path": "path/to/timeline/in/tree"
}
```

7.3.14 Add Layer

The "add-layer" command adds a layer to a timeline in the current project.

The optional "layer-index" argument can be used to add the layer at a specific position in the layer list. An index of "0" will insert the layer at the start of the list.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-layer",
  "path": "path/to/timeline/in/tree",
  "params": {
    "layer-index": "3"
  }
}
```

7.3.15 Move Layer

The "move-layer" command moves an existing layer within a timeline.

The "layer-index" argument is used to specify the new position of the layer in the timeline's layer list. An index of "0" will move the layer to the start of the list.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "move-layer",
  "path": "path/to/timeline/layer/in/tree",
  "params": {
    "layer-index": "2"
  }
}
```

7.3.16 Remove Layer

The "remove-layer" command removes an existing layer from a timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-layer",
  "path": "path/to/timeline/layer/in/tree"
}
```

7.3.17 Enable Layer

The "enable-layer" command marks a layer as active for timeline processing.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "enable-layer",
  "path": "path/to/timeline/layer/in/tree"
}
```

7.3.18 Disable Layer

The "disable-layer" command marks a layer as inactive for timeline processing.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "disable-layer",
  "path": "path/to/timeline/layer/in/tree"
}
```

7.3.19 Add Resource To Timeline

The "add-resource-to-timeline" command adds a resource to a timeline in the current project.

The "uuid" argument specifies the resource to add and refers to an item in the asset database.

The "timeline-path" argument is the path of the timeline to which the resource will be added.

The "time" argument specifies the timeline time at which the resource should be added and is specified in SMPTE format (see 3.3.2).

The optional "layer-name" argument can be used to specify the layer of the timeline that the resource will be added to. If not specified, a new layer will be created.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-resource-to-timeline",
  "uuid": "7c6395c6-1d64-49ae-82e9-5d2de7720600",
  "params": {
    "timeline-path": "path/to/timeline/in/tree",
    "time": "00:01:20:00",
    "layer-name": "layer-two"
  }
}
```

7.3.20 Remove Resource From Timeline

The "remove-resource-from-timeline" command is used to remove a resource from a timeline in the current project.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-resource-from-timeline",
  "path": "path/to/timeline/resource/in/tree"
}
```

7.3.21 Set Timeline Resource Start Time

The "set-timeline-resource-start-time" command sets the start time for a given timeline resource.

The "time" argument is used to specify the new timeline start time for the given resource and is specified using SMPTE format (see 3.3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-timeline-resource-start-time",
  "path": "path/to/timeline/resource/in/tree",
  "params": {
    "time": "00:00:10:00"
  }
}
```

7.3.22 Set Timeline Resource Duration

The "set-timeline-resource-duration" sets the duration for a given timeline asset (how long it will play).

The "duration" argument is used to specify the new timeline duration, up to the original duration of the given asset, and is specified using SMPTE format (see 3.3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-timeline-resource-duration",
  "path": "path/to/timeline/resource/in/tree",
  "params": {
    "duration": "00:00:25:00"
  }
}
```

7.3.23 Trim Timeline Resource Begin

The "trim-timeline-resource-begin" command trims the begin time of a specified asset to the given time.

The "duration" argument is used to specify the amount of time to be removed from the start of the specified resource and is specified using SMPTE format (see 3.3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "trim-timeline-resource-begin",
  "path": "path/to/timeline/resource/in/tree",
  "params": {
    "duration": "00:00:01:10"
  }
}
```

7.3.24 Trim Timeline Resource End

The "trim-timeline-resource-end" command trims the end time of a specified resource to the given time.

The "duration" argument is used to specify the amount of time to be removed from the end of the specified resource and is specified using SMPTE format (see 3.3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "trim-timeline-resource-end",
  "path": "path/to/timeline/resource/in/tree",
  "params": {
    "duration": "00:00:02:00"
  }
}
```

7.3.25 Add ST 2110 Output To Device

The "add-st-2110-output-to-device" command adds an ST 2110 output to the given device.

The "sdp-file-path" is used to specify the local path of the SDP file used to configure the ST 2110 stream.

The optional "primary-source-ip-address", "secondary-source-ip-address", "primary-destination-ip-address" and "secondary-destination-ip-address" arguments can be used to specify IP addresses for the ST 2110 stream.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-st-2110-output-to-device",
  "path": "path/to/device/in/tree",
  "params": {
    "sdp-file-path": "/path/to/st-2110/config",
    "primary-source-ip-address": "192.168.0.1",
    "secondary-source-ip-address": "192.168.0.2",
    "primary-destination-ip-address": "192.168.0.3",
    "secondary-destination-ip-address": "192.168.0.4"
  }
}
```

7.3.26 Add Input To Input Feeds

The "add-input-to-input-feeds" command adds an input object to the input feeds.

The "object-type" argument is used to specify the object type to create for the input.

The optional "linked-object-path" argument can be used to specify the path to the object in the project tree that is linked to this input.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-input-to-input-feeds ",
  "object-type": "actor",
  "params": {
    "linked-object-path": "path/to/linked/object/in/tree"
  }
}
```

7.3.27 Remove Input From Input Feeds

The "remove-input-from-input-feeds" command removes an input object from the input feeds.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-input-to-input-feeds",
  "path": "path/to/input/object/in/tree"
}
```

7.3.28 Add Viewport To Input

The "add-viewport-to-input" command adds a viewport to the given input object.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-viewport-to-input",
  "path": "path/to/input/object/in/tree"
}
```

7.3.29 Remove Viewport From Input

The "remove-viewport-from-input" command removes a viewport from the given input object.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-viewport-from-input",
  "path": "path/to/viewport/in/tree"
}
```

7.3.30 Get Available Presets

The "get-available-presets" command returns a list of the presets which can be used by Compere.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-available-presets"
}
```

Example response

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "presets": {
    "children": [
      {
        "preset": {
          "filename": "my-presets/alpha.pre",
          "can-be-used-by": "recall-preset,apply-preset-to-objects"
        }
      },
      {
        "preset": {
          "filename": "my-presets/beta.pre",
          "can-be-used-by": "apply-preset-to-objects"
        }
      },
      {
        "preset": {
          "filename": "default-preset.pre"
          "can-be-used-by": "recall-preset"
        }
      }
    ]
  }
}
```

7.3.31 Add Object Alias

The "add-object-alias" command creates an alternative name for an object within Compere. This can then be used to interact with the object without needing to specify its path or UUID.

The "path" argument specifies the object which is to be aliased.

The "alias" argument specifies the name of the object alias. The alias must not already exist.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-object-alias",
  "path": "path/to/object/in/tree",
  "params": {
    "alias": "alias-name"
  }
}
```

7.3.32 Remove Object Alias

The "remove-object-alias" command removes an alias previously created with the Add Object Alias command (see 6.3.31).

The "path" argument specifies the object which has the alias.

The "alias" argument specifies the name of the alias to be removed.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-object-alias",
  "path": "path/to/object/in/tree",
  "params": {
    "alias": "alias-name"
  }
}
```

7.3.33 Add Property Alias

The "add-property-alias" command creates an alias of an object's property on another External object.

The "path" argument specifies the full path to the property which is to be aliased.

The "alias" argument specifies the name of the property alias.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-property-alias",
  "path": "path/to/property/of/object/in/tree",
  "params": {
    "alias": "path/to/alias/property/of/object/in/tree"
  }
}
```

7.3.34 Remove Property Alias

The "remove-property-alias" command removes a property alias previously created with the Add Property Alias command (see 6.3.33).

The "path" argument specifies the full path to the property alias.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-object-alias",
  "path": "path/to/alias/property/of/object/in/tree",
}
```

7.3.35 Start Batch

The "start-batch" command allows multiple updates to the project tree to be processed simultaneously. Actions from subsequent project tree messages are queued until a "release-batch" command is received.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "start-batch"
}
```

7.3.36 Release Batch

The "release-batch" command processes all project tree messages queued since a previous "start-batch" command was received.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "release-batch"
}
```

7.3.37 Set Property Over Time

The "set-property-over-time" command allows a property change to be made synchronously across the project group with an optional duration.

The "path" argument specifies the path to the property to be modified.

The "final-value" argument specifies the value of the property at the end of the change period.

The "offset" argument can be used to specify the scheduled time at which the property change should start and is specified using SMPTE format (see 3.3.2). If not specified, the next available slot will be used.

The optional "duration" argument sets the duration of the change and is specified using SMPTE format (see 3.3.2). If not specified, the final value will be applied immediately (after the offset time).

The "forward-request-to" argument is mandatory to avoid unexpected behaviour (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-property-over-time",
  "path": "path/to/property/in/tree",
  "params": {
    "final-value": "0.0",
    "duration": "00:00:20:30",
    "offset": "00:00:05:00",
    "forward-request-to": ""
 }
}
```

7.3.38 Clear Pending Set Property Over Time

The "clear-pending-set-property-over-time" command discards any scheduled property changes over time, created using "set-property-over-time" (see 6.3.37).

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "clear-pending-set-property-over-time"
}
```

7.4 Timeline Control Messages

7.4.1 Play Timeline

The "play-timeline" command submits a play command to the specified timeline.

The "execute-at-time" argument can be used to specify the time at which the play command should be actioned and should be specified in ISO8601 format (see 3.3.1).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "play-timeline",
  "path": "path/to/timeline/in/tree",
  "params": {
    "execute-at-time": "2023-11-23T10:33:00.000Z"
  }
}
```

7.4.2 Pause Timeline

The "pause-timeline" command submits a pause command to the specified timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "pause-timeline",
  "path": "path/to/timeline/in/tree"
}
```

7.4.3 Stop Timeline

The "stop-timeline" command submits a stop command to the specified timeline.

The "execute-at-time" argument can be used to specify the time at which the stop command should be actioned and should be specified in ISO8601 format (see 3.3.1).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "stop-timeline",
  "path": "path/to/timeline/in/tree",
  "params": {
    "execute-at-time": "2023-11-23T10:33:00.000Z"
  }
}
```

7.4.4 Pause Timeline On Next Frame

The "pause-timeline-on-next-frame" command submits a next-frame command to the given timeline and pauses the timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "pause-timeline-on-next-frame",
  "path": "path/to/timeline/in/tree"
}
```

7.4.5 Pause Timeline On Previous Frame

The "pause-timeline-on-previous-frame" command submits a previous-frame command to the given timeline and pauses the timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "pause-timeline-on-previous-frame",
  "path": "path/to/timeline/in/tree"
}
```

7.4.6 Set Timeline Position

The "set-timeline-position" command submits a stop command to the specified timeline.

One of "time", "frame" or "marker" must be specified.

The "time" argument can be used to specify the relative time that the playhead should be set to and is specified using SMPTE format (see 3.3.2).

The "frame" argument can be used to specify the frame number that the playhead should be set to.

The "marker" argument can be used to specify the name of the marker that the playhead should be set to.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-timeline-position",
  "path": "path/to/timeline/in/tree",
  "params": {
    "time": "01:00:00:00"
  }
}
```

7.4.7 Rewind Timeline

The "rewind-timeline" command submits a request to move the playhead to the start of the timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "rewind-timeline ",
  "path": "path/to/timeline/in/tree"
}
```

7.4.8 Add Timeline Marker

The "add-timeline-marker" command creates a new marker on the specified timeline.

The "name" argument is used to specify the name of the marker.

The "type" argument is used to specify the type of marker to add and should be one of "goto" or "stop".

The "time" argument is used to specify the time within the timeline where the marker should be placed and is specified using SMPTE format (see 3.3.2).

The "goto-time" argument is used when a "goto" marker is added and specifies the time within the timeline that the playhead should be moved to and is specified using SMPTE format (see 3.3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "add-timeline-marker",
  "path": "path/to/timeline/in/tree,
  "params": {
    "name": "my-marker",
    "time": "00:01:00:00",
    "type": "goto",
    "goto-time": "00:00:30:00"
  }
}
```

7.4.9 Remove Timeline Marker

The "remove-timeline-marker" command removes a marker from the specified timeline.

The "name" argument is used to specify the name of the marker.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "remove-timeline-marker",
  "path": "path/to/timeline/in/tree,
  "params": {
    "name": "my-marker"
  }
}
```

7.4.10 Get Timeline Markers

The "get-timeline-markers" command returns a list of defined markers for the specified timeline.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-timeline-markers",
  "path": "path/to/timeline/in/tree"
}
```

Example response

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "timeline-markers": {
    "children": [
      {
        "marker": {
          "name": "alpha",
          "time": "P40S",
          "type": "goto",
          "goto-time": "P20.5S"
        }
      },
      {
        "marker": {
          "name": "beta"
          "time": "P1M10S"
          "type": "stop"
        }
      }
    ]
  }
}
```

7.5 Maintenance Messages

7.5.1 Broadcast Logging Control

The "broadcast-logging-control" command controls features of Compere's UDP broadcast logging functionality.

Logging can be switched on and off via this interface.

All arguments within "params" are optional.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "broadcast-logging-control",
  "params": {
    "broadcast-logging-enabled": "true",
    "broadcast-logging-port": "5597",
    "broadcast-logging-network-interface": "enx00e04c682e3a"
  }
}
```

7.5.2 Logging Control

The "logging-control" command controls features of Compere's general logging functionality.

The minimum log level (severity) can be controlled using this interface. Setting it prevents Compere from generating log messages for levels below that specified.

The available log levels (in order of severity) are "verbose", "note", "warning", "important", "error" and "critical".

The optional "apply-to-logging-topics" argument can be used to set the log level for specific logging topics. If not specified, the log level is applied to all logging topics.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "logging-control",
  "params": {
    "minimum-log-level": "warning",
    "apply-to-logging-topics": "comms,network"
  }
}
```

7.5.3 Get Logging Settings

The "get-logging-settings" command allows the current logging settings to be retrieved from Compere.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-logging-settings"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "logging-settings": {
    "logging-topics": {
      "children": [
        {
          "General": {}
        },
        {
          "Audio": {}
        },
        {
          "Comms": {}
        },
        ...
      ]
    }
  }
}
```

7.5.4 Get Tree MD5 Sum

The "get-tree-md5-sum" command can be used to generate an MD5 hash value for part (or all) of the current project tree.

Responses to this command can contain direct addresses (see 3.1.1.2) if the optional "direct-addresses" argument is specified in the request.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "get-tree-md5-sum",
  "path": "path/to/object/in/tree"
}
```

Example response:

```
{
  "type": "response",
  "cookie": 1,
  "caller-id": "<callerId>",
  "sequence-id": 1,
  "ack-required": false,
  "result-code": 0,
  "data": {
    "path": "path/to/object/in/tree",
    "md5sum": "9e107d9d372bb6826bd81d3542a419d6"
  }
}
```

7.5.5 Generate Status Notification

The "generate-status-notification" command allows External Control users to add notifications to the targeted instance of Compere which will then be shown in the notification area of the status bar.

The "severity" argument specifies the importance of the notification and should be one of "info", "warning" or "error".

The "description" argument specifies the text that will be shown in the notification popup.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "generate-status-notification",
  "params": {
    "severity": "error",
    "description": "Something has gone wrong."
  }
}
```

7.5.6 Set Network Settings

The "set-network-settings" command allows the IP address, subnet mask and gateway to be set for an instance of Compere.

If any of the settings are not specified, the existing value will be retained.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Note: This message is currently only honoured by Juggler devices.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-network-settings",
  "params": {
    "ip-address": "192.168.0.10",
    "subnet-mask": "255.255.255.0",
    "gateway": "1.2.3.4"
  }
}
```

7.5.7 Generate System Report

The "generate-system-report" command requests that the targeted instance(s) of Compere generate a system report archive (zip) file containing a snapshot of the system at the current time, including:

- Preference file
- Machine ID file
- Tree Filter file
- Files in the Layout folder
- Files in the Local Render Settings folder
- Files in the Logging folder
- Files in the Projects folder
- Current Preferences tree (as XML)
- Current Project tree (as XML)
- Current Render Settings tree (as XML)
- Current Status tree (as XML)
- Details of currently known peers
- System status (version, hostname, instance ID, startup argument)

The archive will be generated in the "SystemReports" folder on the machine hosting the instance of Compere. Each instance of Compere can also be configured to generate the reports in an additional location, which could be a network share, for example.

While the archive file is being generated (which can take several minutes if large files are present) the generated filename will have the ".part" extension. Do not attempt to copy or move the file while it has this extension, or the archive will be invalidated. Once the archive file is complete, the extension will be removed and replaced with just ".zip", and the file can then be safely moved or copied.

This command can optionally be forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "generate-system-report",
  "params": {
    "forward-request-to": ""
  }
}
```

7.6 GUI Control Messages

7.6.1 Actor Render Panel Control

The "actor-render-panel-control" command can be used to control the state of the actor render panels in the Compere GUI.

The "fullscreen" argument is used to set the full screen state of a render panel.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "actor-render-panel-control",
  "params": {
    "fullscreen": "true"
  }
}
```

7.6.2 Actor Stats Control

The "actor-stats-control" command can be used to control the display of stats on Actor instances of the Compere GUI.

The "show" argument can be used to enable or disable the display of stats.

The "reset-stats" argument can be used to reset all or specific statistics (using a comma separated list).

This command can be optionally forwarded to other instances of Compere (see 3.2).

The following statistics can be reset:

- `repeats`
- `late-frames`
- `local`
- `dropped-media-frames`
- `dropped-graphics-frames`

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "actor-stats-control",
  "params": {
    "show": "true",
    "reset-stats": "dropped-media-frames,dropped-graphics-frames"
  }
}
```

7.7 Juggler Control Messages

7.7.1 Juggler Command

The "juggler-command" command allows commands to be sent to a Juggler.

The "forward-request-to" argument is mandatory as this command is always forwarded to other instances of Compere running on a Juggler (see 3.2).

The "juggler-command-name" argument is used to specify the command itself.

Optional parameters for the juggler command are specified in the "params" block.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "juggler-command",
  "params": {
    "juggler-command-name": "test-juggler-command",
    "forward-request-to": "juggler1",
    "test-parameter": "test-value"
  }
}
```

7.7.2 Reset Juggler GPIO Trigger

The "reset-juggler-gpio-trigger" command allows the GPIO trigger to be reset on a Juggler.

The "forward-request-to" argument is mandatory as this command is always forwarded to other instances of Compere running on a Juggler (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "reset-juggler-gpio-trigger",
  "params": {
    "forward-request-to": "juggler2"
  }
}
```

7.8 7thSense Internal Messages

Important: These messages should only be used by 7thSense employees or at the request of and under the direct supervision of 7thSense.

7.8.1 Tree Handling Control

The "tree-handling-control" command allows preferences associated with tree handling to be set for targeted instances of Compere.

This command can be optionally forwarded to other instances of Compere (see 3.2).

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "tree-handling-control",
  "params": {
    "forward-request-to": "",
    "open-comms-delay-ms": "10000",
    "new-tree-usage-delay-ms": "2000"
  }
}
```

7.8.2 Set TCP Connection Port

The "set-tcp-connection-port" command allows setting of the port number for the main TCP communications socket for targeted instances of Compere.

This command can be optionally forwarded to other instances of Compere (see 3.2). If the "forward-request-to" argument is not specified, it will be forwarded to the entire project group by default.

By convention, this value is normally set across an entire project group.

Note: A restart will be required for the new setting to be used.

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "set-tcp-connection-port",
  "params": {
    "forward-request-to": "",
    "port-number": "5555"
  }
}
```

7.8.3 Configure Time Client

The "configure-time-client" command modifies Time Client settings for the targeted instance(s) of Compere.

If any of the settings are not specified, the existing value will be retained.

This command can optionally be forwarded to other instances of Compere (see 3.2).

The following settings can be changed:

- `time-client-response-timeout-milliseconds`
- `time-client-request-interval-milliseconds`
- `time-client-num-failed-requests-before-reconnection`

Note: *A restart may be required for the new settings to be used.*

Example request:

```
{
  "type": "request",
  "cookie": 1,
  "caller-id": "<callerId>",
  "command": "configure-time-client",
  "params": {
    "time-client-num-failed-requests-before-reconnection": "10"
  }
}
```

E: info@7thsense.one
W: 7thsense.one

7thSense Design Ltd

2 The Courtyard Shoreham Road
Upper Beeding
Steyning
West Sussex
BN44 3TN
UK

T: +44 (0) 1903 812299

7thSense LLC

4207 Vineland Rd
Suite M1
Orlando, FL 32811
USA

T: +1 407 505 5200