Elixys Client-Server JSON

Table of Contents

[Overview 3](#_Toc295679694)

[Data structures 4](#_Toc295679695)

[Time and date 4](#_Toc295679696)

[Button constants 4](#_Toc295679697)

[Button 4](#_Toc295679698)

[Tab 4](#_Toc295679699)

[Table header 5](#_Toc295679700)

[Configuration 5](#_Toc295679701)

[User 5](#_Toc295679702)

[Sequence metadata 6](#_Toc295679703)

[Sequence component 6](#_Toc295679704)

[Reagent 7](#_Toc295679705)

[Pressure regulator state 7](#_Toc295679706)

[Reagent robot state 8](#_Toc295679707)

[Reactor state 9](#_Toc295679708)

[Server state 10](#_Toc295679709)

[Prompt state 11](#_Toc295679710)

[Sequence components 12](#_Toc295679711)

[Cassette 13](#_Toc295679712)

[Add 13](#_Toc295679713)

[Evaporate 14](#_Toc295679714)

[Transfer 15](#_Toc295679715)

[Elute 15](#_Toc295679716)

[React 16](#_Toc295679717)

[Prompt 17](#_Toc295679718)

[Move 17](#_Toc295679719)

[Install 18](#_Toc295679720)

[Comment 18](#_Toc295679721)

[Activity 19](#_Toc295679722)

[Validation 19](#_Toc295679723)

[GET /configuration 20](#_Toc295679724)

[GET /state 20](#_Toc295679725)

[HOME 20](#_Toc295679726)

[SELECTSEQUENCE 21](#_Toc295679727)

[VIEWSEQUENCE 22](#_Toc295679728)

[EDITSEQUENCE 23](#_Toc295679729)

[RUNSEQUENCE 24](#_Toc295679730)

[MANUALRUN 25](#_Toc295679731)

[GET /sequence/[*Sequence ID*] 26](#_Toc295679732)

[GET /sequence/[*Sequence ID*]/component/[*Component ID*] 27](#_Toc295679733)

[GET /sequence/[*Sequence ID*]/reagent/[*Reagent ID*] 27](#_Toc295679734)

[POST /[*state*] 27](#_Toc295679735)

[HOME 27](#_Toc295679736)

[SELECTSEQUENCE 28](#_Toc295679737)

[VIEWSEQUENCE 28](#_Toc295679738)

[EDITSEQUENCE 28](#_Toc295679739)

[RUNSEQUENCE 28](#_Toc295679740)

[MANUALRUN 29](#_Toc295679741)

[PROMPT 29](#_Toc295679742)

[POST /sequence/[*Sequence ID*] 29](#_Toc295679743)

[POST /sequence/[*Sequence ID*]/component/[*Component ID*] 29](#_Toc295679744)

[POST /sequence/[*Sequence ID*]/component/[*Component ID*]/[*Insert ID*] 30](#_Toc295679745)

[POST /sequence/[*Sequence ID*]/reagent/[*Reagent ID*] 30](#_Toc295679746)

[DELETE /sequence/[*Sequence ID*]/component/[*Component ID*] 30](#_Toc295679747)

# Overview

This document describes the messages passed between the Flash client and Python server in the Elixys system. The system is designed such that the client is thin and contains little to no state. The primary purpose of the client is to display the state that the server dictates and relay any significant user input to the server. The server will maintain the state of each client in its database. All aspects of user access control are handled on the server, from dictating the options the client displays to the user to preventing rouge clients from performing actions if they do not have sufficient privileges.

All interactions between the client and server occur over HTTP. Below is a brief overview of these interactions:

* The client can load details about the Elixys system configuration from the server by calling:

**GET /configuration**

This is done immediately after the client connects to load the system version and supported functions.

* The client can obtain the initial state by calling:

**GET /initialstate**

This is done immediately after the client connects and will return either the Home, Run Sequence or Manual Run screens depending on the current state of the Elixys system.

* The client can obtain the current state from the server by calling:

**GET /state**

This is called once per second while the client is running and several times per second during a synthesis to get the latest state of the system.

* The client can load the sequence associated with a given sequence ID and save sequence details by calling:

**GET\_/sequence/[*Sequence ID*]**

**POST\_/sequence/[*Sequence ID*]**

These calls are used when the user is viewing, editing or running a sequence. The POST call returns the new state.

* The client can get a specific sequence component (i.e. cassette or unit operation), update, move or delete an existing component and insert a new component using the following calls:

**GET\_/sequence/[*Sequence\_ID*]/component/[*Component\_ID*]** **POST\_/sequence/[*Sequence\_ID*]/component/[*Component\_ID*]** **POST\_/sequence/[*Sequence\_ID*]/component/[*Component\_ID*]/[*Insert Index*]’**

**DELETE\_/sequence/[*Sequence\_ID*]/component/[*Component\_ID*]**

These calls are used when the user is viewing, editing or running a sequence. The POST and DELETE calls return the new state.

* The client can get or update a specific sequence reagent using the following calls:

**GET\_/sequence/[*Sequence\_ID*]/reagent/[*Reagent\_ID*]** **POST\_/sequence/[*Sequence\_ID*]/reagent/[*Reaagent\_ID*]**

The number of reagents is fixed by the server based on the cassettes in use. The POST call returns the new state.

* The client informs the server of any significant user action by calling:

**POST /[*state*]**

This call returns the new state.

# Data structures

The data structures in this section are used throughout this document.

## Time and date

JSON does not dictate a standard format for times and dates, so the Elixys system will use the following standard:

1. Dates will be in the format “DD/MM/YYYY”.
2. Times will be in the format “HH:MM.SS”.

## Button constants

The following buttons are hardcoded into the client in certain screens because there is no foreseeable need for the server to dictate their presence. The following button IDs are recognized by the server on these screens:

1. BACK
2. NEXT
3. PREVIOUS
4. TRASH

## Button

Sent from the server to the client and describes a button that will be displayed to the user:

1. Text – The button text.
2. ID – Unique ID that is sent from the client to the server when the user clicks on the button.

**{**

**“type”:“button”,**

**“text”:”Create”,**

**“id”:”CREATE”**

**}**

## Tab

Sent from the server to the client and describes a tab in a tab control that will be displayed to the user:

1. Text – The tab text.
2. ID – Unique ID that is sent from the client to the server when the user clicks on the tab.

**{**

**“type”:“tab”,**

**“text”:”Saved Sequences”,**

**“id”:”SAVED”**

**}**

## Table header

Sent from the server to the client and describes a column header in a table:

1. Display text – The text to display at the top of the column.
2. Field name – The name of the data field.

**{**

**“type”:“tableheader”,**

**“text”:”User Comment”,**

**“name”:”comment”,**

**}**

## Configuration

Sent from the server to the client and contains the Elixys system configuration:

1. Name – System name.
2. Version – System version.
3. Debug – Flag that indicates if the server is running in debug mode.
4. Supported operations – Array of unit operation names that are supported by the underlying hardware.

**{**

**“type”:“configuration”,**

**“name”:“Mini cell 3”,**

**“version”:“2.0”,**

**“debug”:”false”,**

**"supportedoperations":**

**[**

**"Add",**

**"Evaporate",**

**"Transfer",**

**"Elute",**

**"React",**

**"Prompt",**

**"Install",**

**"Comment",**

**"Activity"**

**]**

**}**

## User

Describes the current user and consists of the following:

1. User name – The name of the current user for display purposes.
2. User access level – The current user’s access level in the system (e.g. “Tech”). This is for display purposes only as all permissions are enforced on the server side.

**{**

**“type”:”user”,**

**"username":"Homer",**

**"useraccesslevel":"Administrator"**

**}**

## Sequence metadata

Sent from the server to the client and contains the sequence metadata:

1. Name – Sequence name.
2. Date – The date the sequence was created.
3. Time – The time the sequence was created.
4. Comment – Any comment associated with the sequence.
5. ID – Unique ID that is sent from the client to the server when the user selects the sequence.
6. Creator – User that created the sequence.
7. Operations – Number of operations.

**{**

**“type”:“sequencemetadata”,**

**“name”:“FAC (high temp)”,**

**“time”:“8:00”,**

**“date”:“05/01/2012”,**

**“comment”:“Experimental FAC synthesis using high temperatures”,**

**“id”:“452”,**

**“creator”:”devel”,**

**“operations”:”17”**

**}**

## Sequence component

Sent from the server to the client and contains the sequence component metadata:

1. Component name – Short display name for this component.
2. Component ID – Unique ID that the client uses to refer to the component.
3. Component type – The type of unit operation.
4. Validation error – Boolean value that specifies if the component has a validation error.

**{**

**“type”:”sequencecomponent”,**

**"name":"Add F-18",**

**"id":"100",**

**“componenttype”:”ADD”,**

**"validationerror":"false"**

**}**

## Reagent

Describes a reagent and consists of the following:

1. Used – Flag that indicates if this reagent position is used in this cassette.
2. Reagent ID – Unique ID that the client uses to describe the reagent to the server.
3. Component ID – The unique ID of the cassette where this reagent resides.
4. Sequence ID – The unique ID of the sequence associated with this reagent.
5. Position – The reagent position in the cassette.
6. Name – The short name of the reagent.
7. Name description – Description of the name field.
8. Name validation – Contains a string describing the name validation.
9. Description – The long description of the reagent.
10. Description description – Description of the description field.
11. Description validation – Contains a string describing the description validation.

**{**

**“type”:”reagent”,**

**“used”:”true”,**

**“reagentid”:”928”**

**“componentid”:”1”,**

**“sequenceid”:”14”,**

**“position”:”3”,**

**“name”:”F-18”,**

**“namedescription”:”Short name of this reagent”,**

**“namevalidation”:”type=string; required=true”,**

**“description”:”[18F]F-, 10 mg Kryptofix (K222) and 1.0 mg potassium carbonate**

**(K2CO3) in acetonitrile (MeCN)”,**

**“descriptiondescription”:”Long name of this reagent”,**

**“descriptionvalidation”:”type=string”**

**}**

## Pressure regulator state

Describes the state of a pressure regulator and consists of the following:

1. Name – String describing the pressure regulator (e.g. “Main value pressure”).
2. Set pressure – The target pressure in millimeters of mercury (preferred units?).
3. Actual pressure – The actual pressure in millimeters of mercury.

**{**

**“type”:”pressureregulatorstate”,**

**“number”:”1”,**

**“setpressure”:”40”,**

**“actualpressure”:”43”**

**}**

## Reagent robot state

Describes the state of the reagent robot and consists of the following:

1. Position – Name of the current robot position (e.g. “Reactor1.ReagentDelivery1”).
2. Raw X – Gives the raw X position of the robot in millimeters.
3. Raw Y – Gives the raw Y position of the robot in millimeters.
4. Actuator – Specifies the state of the actuator. Possible values are “up”, “down” and “indeterminate”.
5. Gripper – Specifies the state of the robot gripper. Possible values are “open” and “closed”.

**{**

**“type”:”reagentrobotstate”,**

**“position”:”Reactor1.ReagentDelivery1”,**

**“rawx”:”42.4”,**

**“rawy”:”8.5”,**

**“actuator”:”up”,**

**“gripper”:”open”**

**}**

## Reactor state

Describes the state of a reactor and consists of the following:

1. Number – The reactor number.
2. Set temperature – The set temperature of the reactor in degrees Celsius.
3. Actual temperature – The actual temperature of the reactor in degrees Celsius.
4. Position – The reactor position.
5. Vial – The vial state. Possible values are “up”, “down” and “indeterminate”.
6. Activity – The last know radiation activity level of the active reactor in millicuries.
7. Activity time – The time the activity was last measured.
8. Video – URL of the video stream of the active reactor.
9. Evaporation valves – The state of the evaporation values (nitrogen and vacuum). Possible values are “open” and “closed”.
10. Transfer valve – The state of the transfer valve.
11. Reagent 1 transfer valve – The state of the first reagent transfer valve.
12. Reagent 2 transfer valve – The state of the second reagent transfer valve.
13. Stopcock 1 valve – The state of the first stopcock valve.
14. Stopcock 2 valve – The state of the second stopcock valve.
15. Stopcock 3 valve – The state of the third stopcock valve.

**{**

**“type”:”reactorstate”,**

**“number”:”1”,**

**“settemperature”:”165.0”,**

**“actualtemperature”:”75.3”,**

**“position”:”4”,**

**“vial”:”up”,**

**“activity”:”43.5”,**

**“activitytime”:”08:43”,**

**“stirspeed”:”500”,**

**“video”:”rtmp://192.168.0.200:1935/Elixys/mp4:reactor1camera.mp4”,**

**“evaporationvalves”:”open”,**

**“transfervalve”:”closed”,**

**“reagent1transfervalve”:”closed”,**

**“reagent2transfervalve”:”closed”,**

**“stopcock1valve”:”open”,**

**“stopcock2value”:”closed”,**

**“stopcock3valve”:”closed”**

**}**

## Server state

The current state of the server:

1. Run mode – Mode of the instrument. Possible values are “IDLE”, “RUNSEQUENCE” and “MANUALRUN”.
2. Status – String describing the current system status.
3. Sequence ID – The ID of the sequence that is currently running.
4. Active reactor – The active reactor number.
5. Username – The username of the person operating the system.
6. Cooling – Specifies the state of the cooling system. Possible values are “on” and “off”.
7. Vacuum – Specifies the state of the vacuum system. Possible values are “on” and “off”.
8. Door – Specifies the state of the main door. Possible values are “closed” and “open”.
9. Pressure regulators – Array of pressure regulators.
10. Reagent Robot – Details of the reagent robot.
11. Reactors – Array of reactors.

**{**

**“type”:”serverstate”,**

**“runmode”:”RUNSEQUENCE”,**

**“status”:”Reacting, 8:23 minutes”,**

**“sequenceid”:100,**

**“activereactor”:1,**

**“username”:”hsimpson”**

**“cooling”:”on”,**

**“vacuum”:”on”,**

**“door”:”closed”,**

**“pressureregulators”:**

**[**

**{(Pressure regulator state is described above)}**

**]**

**“reagentrobot”:**

**{**

**(Reagent robot state is described above)**

**},**

**“reactors”:**

**[**

**{(Reactor state is described above)}**

**]**

**}**

## Prompt state

The current state of the Prompt modal dialog box:

1. Show – Boolean value that specifies if the prompt dialog is shown.
2. Text 1 – The first text to display to the user.
3. Edit 1 – True if the first edit box is to be displayed.
4. Text 2 – The second text to display to the user or empty if not used.
5. Edit 2 – True if the second edit box is to be displayed.
6. Buttons – Array of buttons to display to the user.

**{**

**"type":"promptstate",**

**“show”:”true”,**

**"text1":"Are you sure you want to abort the current operation?",**

**"edit1":"false",**

**“edit1validation”:”type=string; required=true”,**

**"text2":"",**

**"edit2":”false",**

**“edit2validation”:””,**

**"buttons":**

**{**

**(Buttons are described above)**

**}**

**}**

## Sequence components

Describes an individual component of a sequence:

1. Component Type:
   1. CASSETTE
   2. ADD
   3. EVAPORATE
   4. TRANSFER
   5. ELUTE
   6. REACT
   7. PROMPT
   8. MOVE
   9. INSTALL
   10. COMMENT
   11. ACTIVITY
2. Name – The display name of the component.
3. Component ID – Unique ID that is used by the client to refer to the component when communicating with the server.
4. Sequence ID – Unique ID of the sequence that this component is associated with.
5. Reactor – The reactor associated with this component.
6. Reactor description – Description of the reactor field.
7. Reactor validation – Contains a string describing the reactor validation.
8. Additional details – All of the above component types contain additional information as documented in the next sections.

### Cassette

Describes the configuration of a cassette. Contains the following in addition to the base data members above:

1. Used flag – Boolean value that indicates if this cassette is used.
2. Reagents – Array of reagent IDs.

**{**

**“type”:”component”,**

**“componenttype”:”CASSETTE”,**

**“name”:”Cassette 1”,**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor associated with this cassette”,**

**“reactorvalidation”:””,**

**“used”:”true”,**

**“reagents”:**

**[**

**”847”,**

**”848”,**

**”849”,**

**”850”**

**]**

**}**

### Add

Describes the reagent addition unit operation. Contains the following in addition to the base data members above:

1. Reagent – The reagent ID.
2. Reagent description – Description of the reagent field.
3. Reagent validation – Contains a string describing the reagent validation.

**{**

**“type”:”component”,**

**“componenttype”:”ADD”,**

**“name”:”Add F-18”,**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor where the reagent will be added”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**“reagent”:”344”,**

**“reagentdescription”:”Reagent to add to the reactor”,**

**“reagentvalidation”:”type=enum-reagent; values=211,212,213; required=true”**

**}**

### Evaporate

Describes the evaporation unit operation. Contains the following in addition to the base data members above:

1. Duration – The length of the reaction in the time format.
2. Duration description – Describes the duration field.
3. Duration validation – Contains a string describing the reactor validation.
4. Evaporation temperature – The evaporation temperature in Celsius.
5. Evaporation temperature description – Describes the evaporation temperature field.
6. Evaporation temperature validation – Contains a string describing the evaporation temperature validation.
7. Final temperature – The final temperature in Celsius.
8. Final temperature description – Describes the final temperature field.
9. Final temperature validation – Contains a string describing the final temperature validation.
10. Stir speed – The stir speed in rotations per minute.
11. Stir speed description – Describes the stir speed field.
12. Stir speed validation – Contains a string describing the stir speed field.

**{**

**"type":"component",**

**"componenttype":"EVAPORATE",**

**"name":"Evaporate",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor where the reagent will be added”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"duration":"00:05.00",**

**“durationdescription”:”Evaporation duration after the target temperature is reached”,**

**"durationvalidation":"type=time; min=00:00.00; max=02:00.00; required=true",**

**"evaporationtemperature":"165.0",**

**“evaporationtemperaturedescription”:”Evaporation temperature in degrees Celsius”,**

**"evaporationtemperaturevalidation":"type=temperature, min=20; max=200; required=true",**

**"finaltemperature":"35.0",**

**"finaltemperaturedescription":"Final temperature after evaporation in degrees Celsius",**

**“finaltemperaturevalidation”:”type=temperature; min=20; max=200; required=true”,**

**"stirspeed":"500",**

**"stirspeeddescription":"Speed of the stir bar in rotations per minute"**

**“stirespeedvalidation”:”type=speed; min=0; max=5000; required=true”**

**}**

### Transfer

Describes the transfer unit operation. Contains the following in addition to the base data members above:

1. Target – The target ID.
2. Target description – Description of the target field.
3. Target validation – Contains a string describing the target validation.

**{**

**"type":"component",**

**"componenttype":"TRANSFER",**

**"name":"Transfer",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor whose contents will be transferred”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"target":"321",**

**“targetdescription”:”Target where the reactor contents will be transferred”,**

**"targetvalidation":”type=enum-target; values=321; required=true”,**

**}**

### Elute

Describes the elution unit operation. Contains the following in addition to the base data members above:

1. Reagent – The reagent ID.
2. Reagent description – Description of the reagent field.
3. Reagent validation – Contains a string describing the reagent validation.
4. Target – The target ID.
5. Target description – Description of the target field.
6. Target validation – Contains a string describing the target validation.

**{**

**"type":"component",**

**"componenttype":"ELUTE",**

**"name":"Elute",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor where the reagent will be eluted”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"reagent":"12",**

**"reagentdescription":”Reagent used for elution”,**

**“reagentvalidation”:”type=enum-reagent; values=12,23,34; required=true”,**

**"target":"321",**

**“targetdescription”:”Target through which the eluent will be passed”,**

**"targetvalidation":”type=enum-target; values=321; required=true”,**

**}**

### React

Describes the reaction unit operation. Contains the following in addition to the base data members above:

1. Position – The react position.
2. Position description – Describes the position field.
3. Position validation – Contains a string describing the position validation.
4. Duration – The length of the reaction in the time format.
5. Duration description – Describes the duration field.
6. Duration validation – Contains a string describing the reactor validation.
7. Reaction temperature – The reaction temperature in Celsius.
8. Reaction temperature description – Describes the reaction temperature field.
9. Reaction temperature validation – Contains a string describing the reaction temperature validation.
10. Final temperature – The final temperature in Celsius.
11. Final temperature description – Describes the final temperature field.
12. Final temperature validation – Contains a string describing the final temperature validation.
13. Stir speed – The stir speed in rotations per minute.
14. Stir speed description – Describes the stir speed field.
15. Stir speed validation – Contains a string describing the stir speed field.

**{**

**"type":"component",**

**"componenttype":"REACT",**

**"name":"React",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor where the reagent will be added”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"position":"1",**

**"positiondescription":”Position where the reaction will take place”,**

**“positionvalidation”:”type=enum-literal; values=1,2; required=true”,**

**"duration":"00:04.30",**

**“durationdescription”:”Evaporation duration after the target temperature is reached”,**

**"durationvalidation":"type=time; min=00:00.00; max=02:00.00; required=true",**

**"reactiontemperature":"165.0",**

**“reactiontemperaturedescription”:”Reaction temperature in degrees Celsius”,**

**"reactiontemperaturevalidation":"type=temperature; min=20; max=200; required=true",**

**"finaltemperature":"35.0",**

**"finaltemperaturedescription":"Final temperature after evaporation in degrees Celsius",**

**“finaltemperaturevalidation”:”type=temperature; min=20; max=200; required=true”,**

**"stirspeed":"500",**

**"stirspeeddescription":"Speed of the stir bar in rotations per minute"**

**“stirespeedvalidation”:”type=speed; min=0; max=5000; required=true”**

**}**

### Prompt

Describes the prompt unit operation. Contains the following in addition to the base data members above:

1. Message – Text to display to the user.
2. Message description – Contains a string describing the message field.
3. Message validation – Contains a string describing the message validation.

**{**

**"type":"component",**

**"componenttype":"PROMPT",**

**"name":"Prompt",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:””,**

**“reactordescription”:””,**

**“reactorvalidation”:””,**

**"message":"Please take a sample for analysis",**

**“messagedescription”:”This will be displayed to the user”,**

**"messagevalidation":"type=string; required=true",**

**}**

### Move

Describes the move unit operation. Contains the following in addition to the base data members above:

1. Position – The react position.
2. Position description – Description of the position field.
3. Position validation – Contains a string describing the position validation.
4. State – The reactor state.
5. State description – Description of the state field.
6. State validation – Contains a string describing the state validation.

**{**

**"type":"component",**

**"componenttype":"MOVE",**

**"name":"Move",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor that will be moved”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"position":"1",**

**"positiondescription":”Target position where the reactor will be moved”,**

**“positionvalidation”:”type=enum-literal; values=1,2; required=true”,**

**“state”:”Open”,**

**“statedescription”:”State of the reactor after move”,**

**“statevalidation”:”type=enum-literal; values=Open,Close; required=true”,**

**}**

### Install

Describes the install unit operation. Contains the following in addition to the base data members above:

1. Message – Text to display to the user.
2. Message description – Contains a string describing the message field.
3. Message validation – Contains a string describing the message validation.

**{**

**"type":"component",**

**"componenttype":"INSTALL",**

**"name":"Install",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor that will be moved to the install position”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**"message":"Please take a sample for analysis",**

**“messagedescription”:”This will be displayed to the user”,**

**"messagevalidation":"type=string; required=true",**

**}**

### Comment

Describes the comment unit operation. Contains the following in addition to the base data members above:

1. Comment – User-specified comment.
2. Comment description – Description of the comment field.
3. Comment validation – Contains a string describing the comment validation.

**{**

**"type":"component",**

**"componenttype":"COMMENT",**

**"name":"Comment",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:””,**

**“reactordescription”:””,**

**“reactorvalidation”:””,**

**"comment":"Bromination and cytosine coupling",**

**“commentdescription”:”Enter a comment”,**

**"commentvalidation":"type=string"**

**}**

### Activity

Describes the radiation activity measurement unit operation. This unit operation contains no additional parameters.

**{**

**"type":"component",**

**"componenttype":"ACTIVITY",**

**"name":"Measure activity",**

**“componentid”:”1923”,**

**“sequenceid”:”15”,**

**“reactor”:”1”,**

**“reactordescription”:”Reactor where the radioactivity will be measured”,**

**“reactorvalidation”:”type=enum-literal; values=1,2,3; required=true”,**

**}**

## Validation

Each user-editable field in the state is accompanies by a validation string. This string is semicolon-delimited and can consist of the following options:

1. Type – Describes the type of field. Possible types and the required fields that must accompany each are:
   1. enum-literal – An enumeration of literal values specified in the “values” field. Presented to the user as a list of strings to choose from.
   2. enum-reagent – An enumeration of reagents IDs specified in the “values” field. Presented to the user as a list of reagent short names to choose from.
   3. enum-target – An enumeration of target IDs specified in the “values” field. Presented to the user as a list of target names to choose from.
   4. time – Specifies a time value. May be accompanies by the “min” and “max” fields.
   5. temperature – Specifies a temperature in degrees Celsius. May be accompanied by the “min” and “max” fields.
   6. speed – Specifies a speed in rotations per minute. May be accompanies by the “min” and “max” fields.
   7. string – Specifies a string.
2. Values – Describes a fixed set of comma-separated values. The manner in which the values are interpreted depends on the type.
3. Minimum – The minimum allowed value. The manner in which this is interpreted depends on the type.
4. Maximum – The maximum allowed value. The manner in which this is interpreted depends on the type.
5. Required – Flag that determines if this field is required.

# GET /configuration

This operation returns the Elixys system configuration.

**{**

**(Configuration is described above)**

**}**

# GET /state

This request is sent from the client to the server to fetch the current state. The server responds with the following:

1. User – The current user.
2. Server state – The current state of the server.
3. Prompt state – Set if the server wants the client to prompt the user.
4. Client state – The page the user is currently viewing. Possible values are:
   1. HOME
   2. SELECTSEQUENCE
   3. VIEWSEQUENCE
   4. EDITSEQUENCE
   5. RUNSEQUENCE
   6. MANUALRUN
5. Client state details – Page-specific information is returned as documented in the sections below.

## HOME

The home page contains the following additional information:

1. Buttons – Array of buttons to display on the screen.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"HOME",**

**"buttons":**

**{**

**(Buttons are described above)**

**}**

**}**

## SELECTSEQUENCE

The Select Sequence page contains the following additional information:

1. Tabs – Array of tabs to show in the tab control.
2. Tab ID – The unique ID of the currently selected tab.
3. Table headers – Array of table headers to show on each tab.
4. Navigation buttons – Array of buttons to display in the upper right corner.
5. Option buttons – Array of buttons to display in the bottom center of the tab panel.
6. Sequences – Array of sequence metadata to display in the tab panel.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"SELECTSEQUENCE",**

**"tabs":**

**{**

**(Tabs are described above)**

**},**

**"tabid":"14",**

**“tableheaders”:**

**{**

**(Table headers are described above)**

**}**

**"navigationbuttons":**

**{**

**(Buttons are described above)**

**}**

**"optionbuttons":**

**{**

**(Buttons are described above)**

**}**

**"sequences":**

**{**

**(Sequence metadata are described above)**

**}**

**}**

## VIEWSEQUENCE

The View Sequence page contains the following additional information:

1. Navigation buttons – Array of buttons to display in the upper right corner.
2. Sequence ID – The unique ID of the sequence the user is currently viewing.
3. Component ID – The unique ID of the component in the sequence the user is currently viewing.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"VIEWSEQUENCE",**

**"navigationbuttons":**

**{**

**(Buttons are described above)**

**}**

**"sequenceid":"65",**

**"componentid":”422"**

**}**

## EDITSEQUENCE

The Edit Sequence page contains the following additional information:

1. Navigation buttons – Array of buttons to display in the upper right corner.
2. Sequence ID – The unique ID of the sequence the user is currently viewing.
3. Component ID – The unique ID of the component in the sequence the user is currently viewing.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"EDITSEQUENCE",**

**"navigationbuttons":**

**{**

**(Buttons are described above)**

**}**

**"sequenceid":"65",**

**"componentid":422"**

**}**

## RUNSEQUENCE

The Run Sequence page contains the following additional information:

1. Navigation button – Button to display in the upper right corner.
2. Sequence ID – The unique ID of the sequence the user is currently running.
3. Component ID – The unique ID of the component in the sequence the user is currently running.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"RUNSEQUENCE",**

**"navigationbuttons":**

**{**

**(Buttons are described above)**

**}**

**"sequenceid":"65",**

**"componentid":422"**

**}**

## MANUALRUN

The Manual Run page contains the following additional information:

1. Manual run step – Describes the current step in the manual run:
   1. CASSETTE – The user is configuring the cassettes prior to starting the run.
   2. SELECT – The user is prompted to select the next unit operation.
   3. CONFIGURE – The user is prompted to configure the unit operation.
   4. RUN – The unit operation is being performed.
2. Navigation buttons – Array of buttons to display in the upper right corner.
3. Sequence ID – The unique ID of the sequence the user is currently running.
4. Component ID – The unique ID of the component in the sequence. This component is the one the user must configure in the CONFIGURE step and the one the system is running in the RUN step.
5. Operation result – Boolean value that specifies if the last unit operation was successful. This value will be set when returning to the SELECT step after a RUN.

**{**

**"type":"state",**

**“user”:**

**{**

**(User is described above)**

**}**

**"serverstate":**

**{**

**(Server state is described above)**

**},**

**“promptstate”:**

**{**

**(Prompt state is described above)**

**}**

**"clientstate":"MANUALRUN",**

**"manualrunstep":"SELECT",**

**"navigationbuttons":**

**{**

**(Buttons are described above)**

**}**

**"sequenceid":"65",**

**"componentid":"422",**

**“operationresult”:true**

**}**

# GET /sequence/[*Sequence ID*]

This operation returns the following for the given sequence ID:

1. Sequence metadata – Details of the sequence.
2. Sequence components – Array of the sequence components.

**{**

**"type":"sequence",**

**"metadata":**

**{**

**(Sequence metadata are described above)**

**},**

**"components":**

**{**

**(Sequence components are described above)**

**}**

**}**

# GET /sequence/[*Sequence ID*]/component/[*Component ID*]

This operation returns the component details for the given sequence and component IDs.

**{**

**(Component details are described above)**

**}**

# GET /sequence/[*Sequence ID*]/reagent/[*Reagent ID*]

This operation returns the reagent details for the given sequence and reagent IDs.

**{**

**(Reagent details are described above)**

**}**

# POST /[*state*]

The client uses POST to inform the server of any significant user action on the page specified by *state*:

1. Action – The action performed by the user:
   1. Action – Describes the user action. Possible values are:
      1. BUTTONCLICK
      2. TABCLICK
   2. Target ID – ID of the target component the user acted on.
2. Additional information – Page-specific information as documented in the sections below.

The server always returns the new state of the client as a response to a POST.

## HOME

The Home page does not send any additional information.

**POST /HOME**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**}**

**}**

## SELECTSEQUENCE

The select sequence page sends the following in addition to the action:

1. Sequence ID – The unique ID of the currently selected sequence.

**POST /SELECTSEQUENCE**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**},**

**“sequenceid”:”9000”**

**}**

## VIEWSEQUENCE

The View Sequence page does not send any additional information. Each component in the sequence is displayed as a button. If the user clicks on one of these buttons then a BUTTONCLICK action will be sent to the server with the component ID for the button ID.

**POST /VIEWSEQUENCE**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**}**

**}**

## EDITSEQUENCE

The Edit Sequence page does not send any additional information. Each component in the sequence is displayed as a button. If the user clicks on one of these buttons then a BUTTONCLICK action will be sent to the server with the component ID for the button ID.

**POST /EDITSEQUENCE**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**}**

**}**

## RUNSEQUENCE

The Run Sequence page does not send any additional information.

**POST /RUNSEQUENCE**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**}**

**}**

## MANUALRUN

The Manual Run page does not send any additional information.

**POST /MANUALRUN**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**}**

**}**

## PROMPT

The Prompt modal dialog box contains the following additional information:

1. Edit 1 – The text from the first edit box.
2. Edit 2 – The text from the second edit box.

**POST /PROMPT**

**{**

**"action":**

**{**

**"type":"BUTTONCLICK",**

**"targetid":"BACK"**

**},**

**“edit1”:”New sequence name”,**

**“edit2”:””**

**}**

# POST /sequence/[*Sequence ID*]

This operation is used by the client to save sequence metadata to the server. Specify a sequence ID to update an existing sequence or leave it blank to create a new one.

**{**

**(Sequence details are described above)**

**}**

# POST /sequence/[*Sequence ID*]/component/[*Component ID*]

This operation is used by the client to update an existing unit operation within a sequence.

**{**

**(Component details are described above)**

**}**

# POST /sequence/[*Sequence ID*]/component/[*Component ID*]/[*Insert ID*]

This operation is used by the client to insert the given component in the sequence at the insertion ID. Insert a new component by setting the component ID to zero and passing the new component in the body of the request. Move an existing component by specifying the ID of the component to move in which case the server will ignore anything passed in the body of the request.

**{**

**(Component details are described above)**

**}**

# POST /sequence/[*Sequence ID*]/reagent/[*Reagent ID*]

This operation is used by the client to update reagent details for an existing reagent.

**{**

**(Reagent details are described above)**

**}**

# DELETE /sequence/[*Sequence ID*]/component/[*Component ID*]

This operation is used by the client to delete an existing unit operation from a sequence.