Elixys Client-Server JSON

Table of Contents

[Overview 3](#_Toc304148164)

[Data structures 4](#_Toc304148165)

[Time and date 4](#_Toc304148166)

[Button constants 4](#_Toc304148167)

[Button 4](#_Toc304148168)

[Tab 4](#_Toc304148169)

[Table header 5](#_Toc304148170)

[Configuration 5](#_Toc304148171)

[User 6](#_Toc304148172)

[Sequence metadata 6](#_Toc304148173)

[Sequence component 7](#_Toc304148174)

[Reagent 7](#_Toc304148175)

[Pressure regulator state 8](#_Toc304148176)

[Reagent robot state 8](#_Toc304148177)

[Reactor state 9](#_Toc304148178)

[Server state 10](#_Toc304148179)

[Prompt state 11](#_Toc304148180)

[Sequence components 12](#_Toc304148181)

[Cassette 12](#_Toc304148182)

[Add 13](#_Toc304148183)

[Evaporate 14](#_Toc304148184)

[Transfer 15](#_Toc304148185)

[Elute 15](#_Toc304148186)

[React 16](#_Toc304148187)

[Prompt 17](#_Toc304148188)

[Install 17](#_Toc304148189)

[Comment 18](#_Toc304148190)

[Validation 18](#_Toc304148191)

[GET /configuration 19](#_Toc304148192)

[GET /state 19](#_Toc304148193)

[HOME 19](#_Toc304148194)

[SELECTSEQUENCE 20](#_Toc304148195)

[VIEWSEQUENCE 21](#_Toc304148196)

[EDITSEQUENCE 22](#_Toc304148197)

[RUNSEQUENCE 23](#_Toc304148198)

[MANUALRUN 24](#_Toc304148199)

[GET /sequence/[*Sequence ID*] 25](#_Toc304148200)

[GET /sequence/[*Sequence ID*]/component/[*Component ID*] 25](#_Toc304148201)

[GET /sequence/[*Sequence ID*]/reagent/[*Reagent ID 1*].[*Reagent ID 2*] 25](#_Toc304148202)

[POST /[*state*] 25](#_Toc304148203)

[HOME 26](#_Toc304148204)

[SELECTSEQUENCE 26](#_Toc304148205)

[VIEWSEQUENCE 26](#_Toc304148206)

[EDITSEQUENCE 26](#_Toc304148207)

[RUNSEQUENCE 27](#_Toc304148208)

[MANUALRUN 27](#_Toc304148209)

[PROMPT 27](#_Toc304148210)

[POST /sequence/[*Sequence ID*] 27](#_Toc304148211)

[POST /sequence/[*Sequence ID*]/component/[*Component ID*] 28](#_Toc304148212)

[POST /sequence/[*Sequence ID*]/component/[*Component ID*]/[*Insert ID*] 28](#_Toc304148213)

[POST /sequence/[*Sequence ID*]/reagent/[*Reagent ID*] 28](#_Toc304148214)

[DELETE /sequence/[*Sequence ID*]/component/[*Component ID*] 28](#_Toc304148215)

# 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 current state from the server by calling:

**GET /state**

This is called at regular intervals while the client is running and more frequently 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 one or more reagents and update a specific reagent using the following calls:

**GET\_/sequence/[*Sequence\_ID*]/reagent/[*Reagent\_ID 1*].[Reagent ID 2].[Reagent ID N]**

**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 MySQL timestamp format:

YYYY-MM-DD 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"**

**]**

**}**

## User

Describes the current user and consists of the following:

1. User name – The name of the current user for display purposes.
2. First name – The user’s first name.
3. Last name – The user’s last name.
4. 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":"hsimpson",**

**“firstname”:”Homer”,**

**“lastname”:”Simpson”,**

**"useraccesslevel":"Administrator"**

**}**

## Sequence metadata

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

1. Name – Sequence name.
2. Timestamp – The date and time the sequence was created.
3. Comment – Any comment associated with the sequence.
4. ID – Unique ID that is sent from the client to the server when the user selects the sequence.
5. Creator – Name of the user that created the sequence.
6. Components – Number of components in this sequence.

**{**

**“type”:“sequencemetadata”,**

**“name”:“FAC (high temp)”,**

**“timestamp”:“2012-05-01 08:00:12”,**

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

**“id”:452,**

**“creator”:”devel”,**

**“components”:17**

**}**

## Sequence component

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

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. Available – Flag that indicates if a reagent is available in this position.
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. Position – The reagent position in the cassette.
5. Name – The short name of the reagent.
6. Name validation – Contains a string describing the name validation.
7. Description – The long description of the reagent.
8. Description validation – Contains a string describing the description validation.

**{**

**“type”:”reagent”,**

**“available”:true,**

**“reagentid”:928,**

**“componentid”:1,**

**“position”:”3”,**

**“name”:”F-18”,**

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

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

**(K2CO3) in acetonitrile (MeCN)”,**

**“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.
2. Pressure – The actual pressure in PSI.

**{**

**“type”:”pressureregulatorstate”,**

**“name”:”Gas”,**

**“pressure”:4.3**

**}**

## Reagent robot state

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

1. Position – The current position of the reagent robot:
   1. Cassette – Number of the cassette the robot is over or zero if indeterminate.
   2. Reagent position – Number of the reagent position the robot is over or zero if not over a reagent position.
   3. Delivery position – Number of the delivery position the robot is over or zero if not over a delivery position.
2. Actuator – String that specifies the state of the actuator. Possible values are “up”, “down” and “indeterminate”.
3. Gripper – String that specifies the state of the gripper. Possible values are “open”, “closed” and “indeterminate”. Note that “closed” indicates a vial is between the fingers while “indeterminate” indicates no vial present.

**{**

**“type”:”reagentrobotstate”,**

**"position":{"cassette":1,**

**"reagent":3,**

**"delivery":0},**

**“actuator”:”up”,**

**“gripper”:”open”**

**}**

## Reactor state

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

1. Number – The reactor number.
2. Temperature – The actual temperature of the reactor in Celsius.
3. Position – The reactor position which consists of:
   1. Horizontal – String describing the horizontal position, e.g. “Add”, “Evaporate”.
   2. Vertical – String describing the vertical position. Possible values are “up”, “down” and “indeterminate”.
4. Activity – The last know radiation activity level of the reactor in millicuries.
5. Activity time – The time the activity was measured.
6. Evaporation – Boolean value that indicates if the evaporation values are open.
7. Transfer valve – Boolean value that indicates if the transfer valve is open.
8. Transfer position – String that describes the transfer stopcock position. Possible values are “Waste” and “Out”.
9. Reagent 1 transfer valve – Boolean value that indicates if the first reagent transfer valve is open.
10. Reagent 2 transfer valve – Boolean value that indicates if the second reagent transfer valve is open.
11. Stir speed – Speed of the stir motor.
12. Video – URL of the video stream.

Reactor 1 will have the following additional fields:

1. Column position – String that describes the position of the column stopcocks. Possible values are “Load” and “Elute”.
2. F18 transfer valve – Boolean value that indicates if the F18 transfer valve is open.
3. Eluent transfer value – Boolean value that indicates if the eluent transfer valve is open.

**{**

**"type":"reactorstate",**

**"number":1,**

**"temperature":165.0,**

**"position":{"horizontal":"React1",**

**"vertical":"Up"},**

**"activity":"645.1",**

**"activitytime":"18:23.31",**

**"evaporation":False,**

**"transfer":False,**

**"transferposition":"waste",**

**"reagent1transfer":False,**

**"reagent2transfer":False,**

**"stirspeed":500,**

**“video”:”rtmp://192.168.1.100/Elixys/mp4:reactor1.mp4”,**

**"columnposition":"Load",**

**"f18transfer":False,**

**"eluenttransfer":False**

**}**

## Prompt state

The current state of the Prompt modal dialog box:

1. Show – Boolean value that specifies if the prompt dialog is shown.
2. Title – Title of the prompt dialog.
3. Text 1 – The first text to display to the user.
4. Edit 1 – True if the first edit box is to be displayed.
5. Edit 1 default – Default text to display in the first edit box.
6. Edit 1 validation – Validation string for the text entered by the user in the first edit box.
7. Text 2 – The second text to display to the user or empty if not used.
8. Edit 2 – True if the second edit box is to be displayed.
9. Edit 2 default – Default text to display in the second edit box.
10. Edit 2 validation – Validation string for the text entered by the user in the second edit box.
11. Buttons – Array of buttons to display to the user.

**{**

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

**“show”:”true”,**

**“title”:”Abort”,**

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

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

**“edit1default:”:””,**

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

**"text2":"",**

**"edit2":”false",**

**“edit2default:”:””,**

**“edit2validation”:””,**

**"buttons":**

**[**

**{Buttons are described above}**

**]**

**}**

## Run state

The current run state of the server:

1. Status – String describing the current system status. Possible values are “Idle”, “Running” and “Paused”.
2. Username – Name of the user that is operating the system.
3. Sequence ID – ID of the sequence that the system is running.
4. Component ID – ID of the component that the system is currently on.
5. Prompt – Describes any prompt that is being displayed in association with the run.

**{**

**“type”:”runstate”,**

**“status”:”Running”,**

**“username”:”hsimpson”,**

**“sequenceid”:4,**

**“componentid”:6,**

**“prompt”:{Prompt state is described above}**

**}**

## Hardware state

The current hardware state of the server:

1. Cooling – Specifies if the cooling system is on.
2. Vacuum – The actual vacuum in kPa.
3. Pressure regulators – Array of pressure regulators.
4. Reagent Robot – Details of the reagent robot.
5. Reactors – Array of reactors.

**{**

**“type”:”hardwarestate”,**

**“cooling”:False,**

**“vacuum”:-43.5,**

**“pressureregulators”:**

**[**

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

**]**

**“reagentrobot”:{Reagent robot state is described above},**

**“reactors”:**

**[**

**{Reactor state is described above}**

**]**

## }

## Server state

The current state of the server:

1. Run state – The current run state of the server.
2. Hardware state – The current hardware state of the server

**{**

**“type”:”serverstate”,**

**“runstate”:{Run state is described above},**

**“hardwarestate”:{Hardware state is described above}**

**}**

## 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-number – An enumeration of number values specified in the “values” field. Presented to the user as a list of options 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. 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 component

Used by the server to describe an individual component of a sequence to the client. Contains the following base fields plus additional component-specific detail documented in the sections below:

1. Component Type:
   1. CASSETTE
   2. ADD
   3. EVAPORATE
   4. TRANSFER
   5. ELUTE
   6. REACT
   7. PROMPT
   8. INSTALL
   9. COMMENT
2. ID – Unique ID that is used by the client to refer to the component when communicating with the server.
3. Name – The display name of the component.
4. Validation error – Flag that indicates that one of the fields of this component are invalid.

## Cassette

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

1. Reactor – The reactor associated with this component.
2. Available flag – Boolean value that indicates if this cassette is available.
3. Reagents – Array of reagents.

**{**

**“type”:”component”,**

**“componenttype”:”CASSETTE”,**

**“id”:1,**

**“name”:”Cassette 1”,**

**"validationerror": false,**

**“reactor”:1,**

**“available”:”true”,**

**“reagents”:**

**[**

**(Reagents are described above)**

**]**

**}**

## Add

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Reagent – The reagent to add to the reactor.
4. Reagent validation – Contains a string describing the reagent validation. Included in this string are the IDs of the other reagents that the user can choose from in Edit mode.
5. Delivery position – Reagent delivery position. Possible values are 1 and 2.
6. Delivery position validation – Contains a string describing the delivery position validation.
7. Delivery time – Time to deliver the reagent in seconds. This value will override the default if set or ignored if zero.
8. Delivery time validation – Contains a string describing the delivery time validation.
9. Delivery pressure – Pressure in PSI to use when delivering the reagent. This value will override the default if set or ignored if zero.
10. Delivery pressure validation – Contains a string describing the delivery pressure validation.

**{**

**“type”:”component”,**

**“componenttype”:”ADD”,**

**“id”:1,**

**“name”:”Add K222”,**

**"validationerror": false,**

**“reactor”:1,**

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

**“reagent”:(Reagent details are described above),**

**“reagentvalidation”:”type=enum-reagent; values=2,3,4,5; required=true”**

**"deliveryposition":2,**

**“deliverypositionvalidation”:”type=enun-number; values=1,2; required=true”,**

**"deliverytime":0,**

**“deliverytimevalidation”:”type=number; min=0; max=10”,**

**"deliverypressure":0,**

**“deliverypressurevalidation”:”type=number; min=0; max=15”**

**}**

## Evaporate

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Duration – The length of the evaporation in seconds.
4. Duration validation – Contains a string describing the reactor validation.
5. Evaporation temperature – The evaporation temperature in Celsius.
6. Evaporation temperature validation – Contains a string describing the evaporation temperature validation.
7. Final temperature – The final temperature in Celsius.
8. Final temperature validation – Contains a string describing the final temperature validation.
9. Stir speed – The stir speed in arbitrary units.
10. Stir speed validation – Contains a string describing the stir speed field.
11. Evaporation pressure – Nitrogen pressure in PSI to use when evaporating. This value will override the default if set or ignored if zero.
12. Evaporation pressure validation – Contains a string describing the evaporation pressure validation.

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

**“reactor”:1,**

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

**"duration":300,**

**"durationvalidation":"type=number; min=0; max=7200; required=true",**

**"evaporationtemperature":165.0,**

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

**"finaltemperature":35.0,**

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

**"stirspeed":500,**

**“stirspeedvalidation”:”type=number; min=0; max=5000; required=true”,**

**“evaporationpressure”:10,**

**“evaporationpressurevalidation”:”type=number; min=0; max=25”**

**}**

## Transfer

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Target – The target reagent.
4. Target validation – Contains a string describing the target validation.

**{**

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

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

**“id”:15,**

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

**"validationerror": false,**

**“reactor”:1,**

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

**"target":(Reagent details are described above),**

**"targetvalidation":”type=enum-target; values=11; required=true”,**

**}**

## Elute

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Reagent – The elution reagent.
4. Reagent validation – Contains a string describing the reagent validation.
5. Target – The target reagent.
6. Target validation – Contains a string describing the target validation.

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

**“reactor”:1,**

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

**"reagent":(Reagent details are described above),**

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

**"target":(Reagent details are described above),**

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

**}**

## React

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Position – The react position.
4. Position validation – Contains a string describing the position validation.
5. Duration – The length of the reaction in the time format.
6. Duration validation – Contains a string describing the reactor validation.
7. Reaction temperature – The reaction temperature in Celsius.
8. Reaction temperature validation – Contains a string describing the reaction temperature validation.
9. Final temperature – The final temperature in Celsius.
10. Final temperature validation – Contains a string describing the final temperature validation.
11. Stir speed – The stir speed in arbitrary units.
12. Stir speed validation – Contains a string describing the stir speed field.

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

**“reactor”:1,**

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

**"position":1,**

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

**"duration":270,**

**"durationvalidation":"type=number; min=0; max=7200; required=true",**

**"reactiontemperature":165.0,**

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

**"finaltemperature":35.0,**

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

**"stirspeed":500,**

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

**}**

## Prompt

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

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

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

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

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

**}**

## Install

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

1. Reactor – The reactor associated with this component.
2. Reactor validation – Contains a string describing the reactor validation.
3. Message – Text to display to the user.
4. Message validation – Contains a string describing the message validation.

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

**“reactor”:1,**

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

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

**"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 validation – Contains a string describing the comment validation.

**{**

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

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

**“id”:1,**

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

**"validationerror": false,**

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

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

**}**

# 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 1*].[*Reagent ID 2*]

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

**{**

**“type”:”reagents”,**

**“reagents”:**

**[**

**(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 component

Used by the client when updating or creating an individual sequence component on the server. Contains the following base fields plus additional component-specific detail documented in the sections below:

1. Component Type:
   1. CASSETTE
   2. ADD
   3. EVAPORATE
   4. TRANSFER
   5. ELUTE
   6. REACT
   7. PROMPT
   8. INSTALL
   9. COMMENT
2. ID – Unique ID of the component obtained from the server when updating an existing component or zero for a new component.
3. Name – The display name of the component. This field is optional and can be left blank.

## Cassette

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

1. Available flag – Boolean value that indicates if this cassette is available.

**{**

**“type”:”component”,**

**“componenttype”:”CASSETTE”,**

**“id”:1,**

**“name”:”Cassette 1”,**

**“available”:”true”**

**}**

## Add

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

1. Reactor – The reactor where the reagent will be added.
2. Reagent – The ID of the reagent to add to the reactor.
3. Delivery position – Reagent delivery position. Possible values are 1 and 2.
4. Delivery time – Time to deliver the reagent in seconds. This value will override the default if nonzero.
5. Delivery pressure – Pressure in PSI to use when delivering the reagent. This value will override the default if nonzero.

**{**

**“type”:”component”,**

**“componenttype”:”ADD”,**

**“id”:1,**

**“name”:”Add K222”,**

**“reactor”:1,**

**“reagent”:14,**

**"deliveryposition":2,**

**"deliverytime":0,**

**"deliverypressure":0**

**}**

## Evaporate

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

1. Reactor – The reactor that will be evaporated.
2. Duration – The length of the evaporation in seconds.
3. Evaporation temperature – The evaporation temperature in Celsius.
4. Final temperature – The final temperature in Celsius.
5. Stir speed – The stir speed in arbitrary units.
6. Evaporation pressure – Nitrogen pressure in PSI to use when evaporating. This value will override the default if nonzero.

**{**

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

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

**“id”:1,**

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

**“reactor”:1,**

**"duration":600,**

**"evaporationtemperature":165.0,**

**"finaltemperature":35.0,**

**"stirspeed":500,**

**“evaporationpressure”:10,**

**}**

## Transfer

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

1. Reactor – The reactor associated with this component.
2. Target – The target reagent ID.

**{**

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

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

**“id”:15,**

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

**“reactor”:1,**

**"target":43**

**}**

## Elute

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

1. Reactor – The reactor associated with this component.
2. Reagent – The elution reagent.
3. Target – The target reagent.

**{**

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

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

**“id”:1,**

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

**“reactor”:1,**

**"reagent":51,**

**"target":14,**

**}**

## React

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

1. Reactor – The reactor associated with this component.
2. Position – The react position.
3. Duration – The length of the reaction in seconds.
4. Reaction temperature – The reaction temperature in Celsius.
5. Final temperature – The final temperature in Celsius.
6. Stir speed – The stir speed in arbitrary units.

**{**

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

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

**“id”:1,**

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

**“reactor”:1,**

**"position":1,**

**"duration":300,**

**"reactiontemperature":165.0,**

**"finaltemperature":35.0,**

**"stirspeed":500**

**}**

## Prompt

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

1. Message – Text to display to the user.

**{**

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

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

**“id”:1,**

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

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

**}**

## Install

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

1. Reactor – The reactor associated with this component.
2. Message – Text to display to the user.

**{**

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

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

**“id”:1,**

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

**“reactor”:1,**

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

**}**

## Comment

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

1. Comment – User-specified comment.

**{**

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

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

**“id”:1,**

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

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

**}**

# 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.