



Exastro★

OASE Operation Autonomy Support Engine

Base [Practice]

※In this document, “Operation Autonomy Support Engine” will be written as “OASE”.

Exastro Operation Autonomy Support Engine Version 1.3.1
Exastro developer

Table of contents

1. Introduction
 - 1.1 Base 【Practice】
 2. Scenario description
 - 2.1 Scenario
 3. Preparation
 - 3.1 Create group
 - 3.2 Create and log in as new user
 - 3.3 Pay out Token
 - 3.4 Action settings (ITA driver)
 - 3.5 Create decision table
 4. Operation
 - 4.1 Create decision table file
 - 4.2 Register rules (Upload and test request)
 - 4.3 Judge rules (Send request via curl command)
 - 4.4 Check the Action results
- A Appendix
Sample 1

1. Introduction



1.1 Base [Practice] (1/3)

Preface

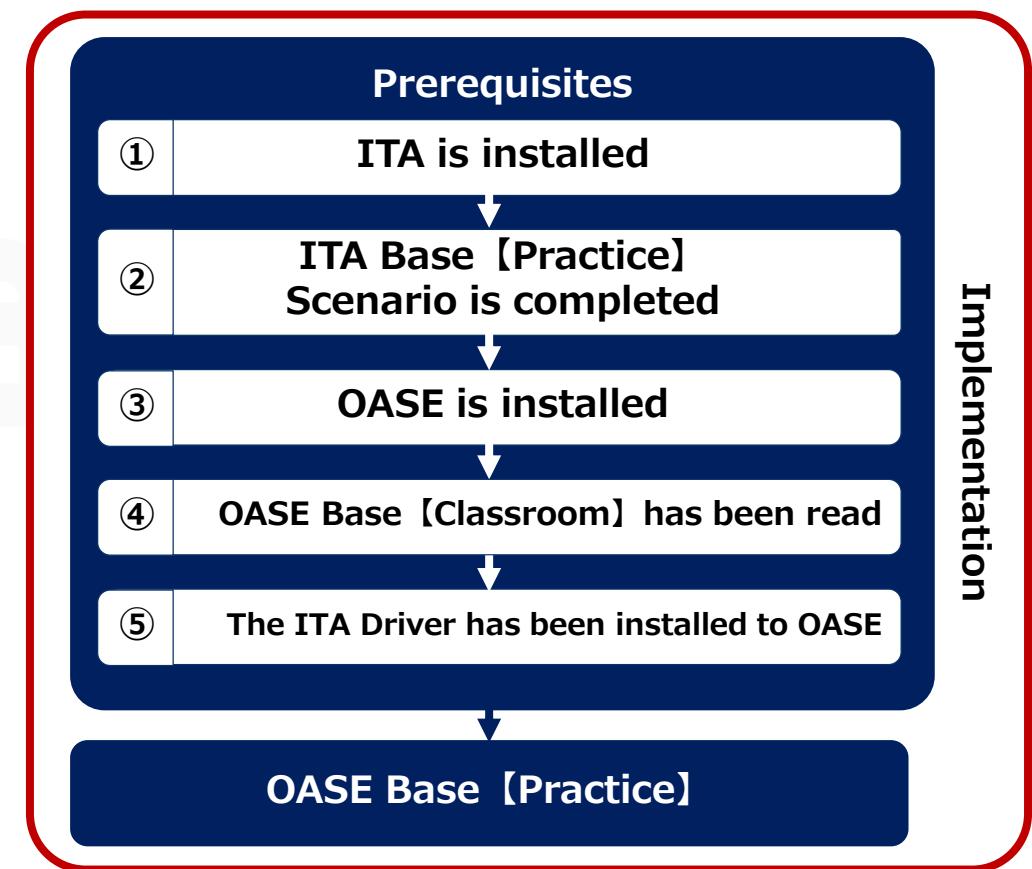
- This document uses Exastro Operation Autonomy Support Engine (OASE) to help users better understand the software's basic functions.
- If you want to learn more about the overview of OASE, we recommend that you see the [Exastro OASE Base \[Classroom\]](#) document.
- For documents with more comprehensive information, please see the official Exastro OASE manual page, [OASE docs](#).

1.1 Base [Practice] (2/3)

Prerequisites

- This document will have the user run ITA as an action, and will therefore assume that the following are installed/implemented.

- ① Please see the following document for installing ITA.
 - <[ITA Online install manual](#)>
- ② Please see the following document for the IT Automation BASE Practice document
 - <[IT Automation BASE \[Practice\]](#)>
 - Make sure to follow all the slides that comes after the "Scenario" slide, as we will use the settings from them.
- ③ Please see the following document for installing OASE.
 - <[OASE Online install manual](#)>
 - Make sure to include the mail server settings, as we will need them when we are creating a new user.
- ④ Please see the following document for OASE Base (Classroom)
 - <[OASE Base \[lecture\]](#)>
- ⑤ Please see the following document for installing the ITA Driver.
 - <[Environment construction manual -Install drivers->](#)

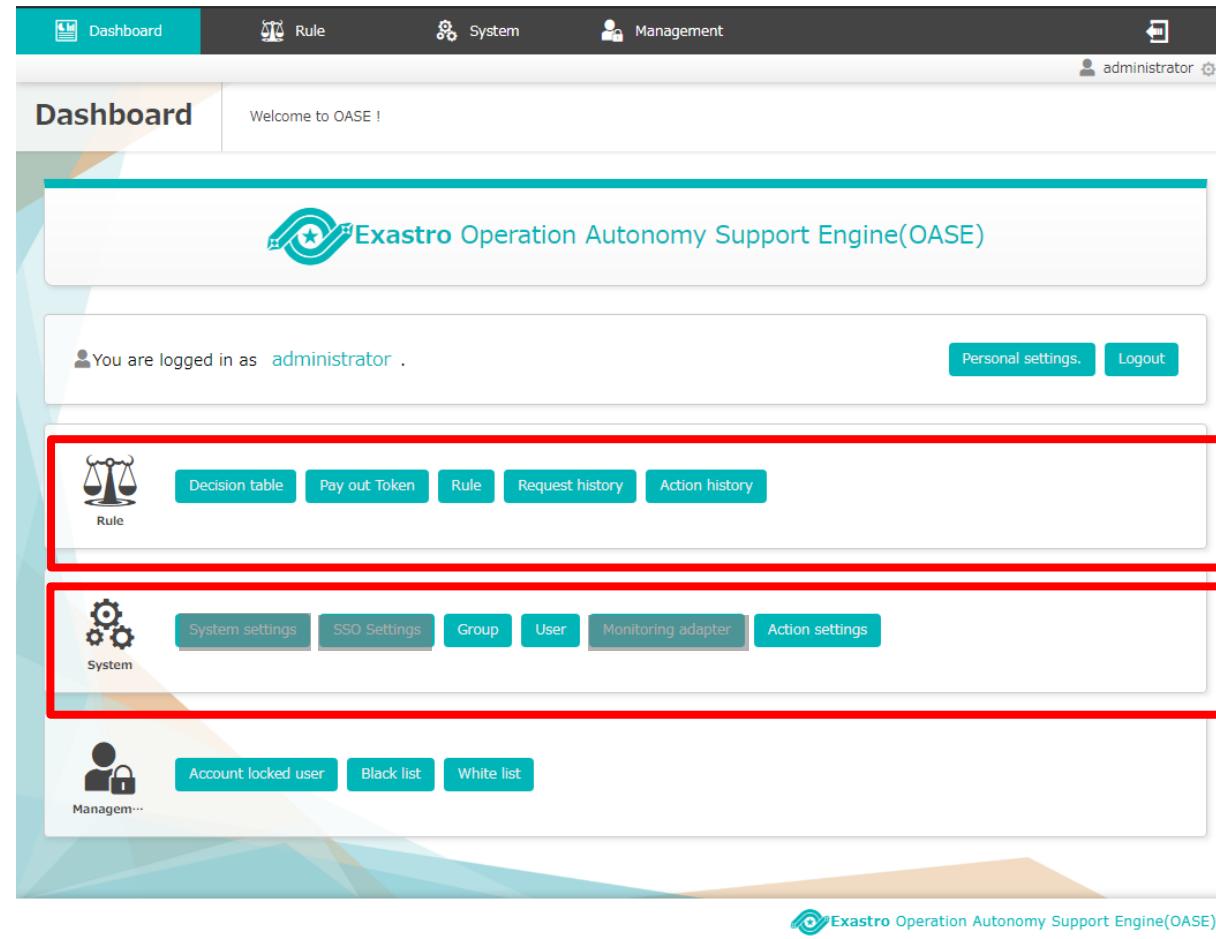


※In this document, IT Automation will be written as "ITA"

1.1 Base [Practice] (3/3)

This guide will use the following OASE functions

- Dashboard screen



Category : rule

Screen
Decision table
Pay out Token
Rule
Request History
Action History

Category : System

Screen
Group
User
Action settings

2. Scenario

2.1 Scenario description (1/2)

The following illustrates the scenario of this document (After installing OASE)

[Preparation]

Various settings

- 1 Create group**
- 2 Create and log in as new user.**
- 3 Pay out Token**
- 4 Configure Action (ITA driver)**
- 5 Create decision table**

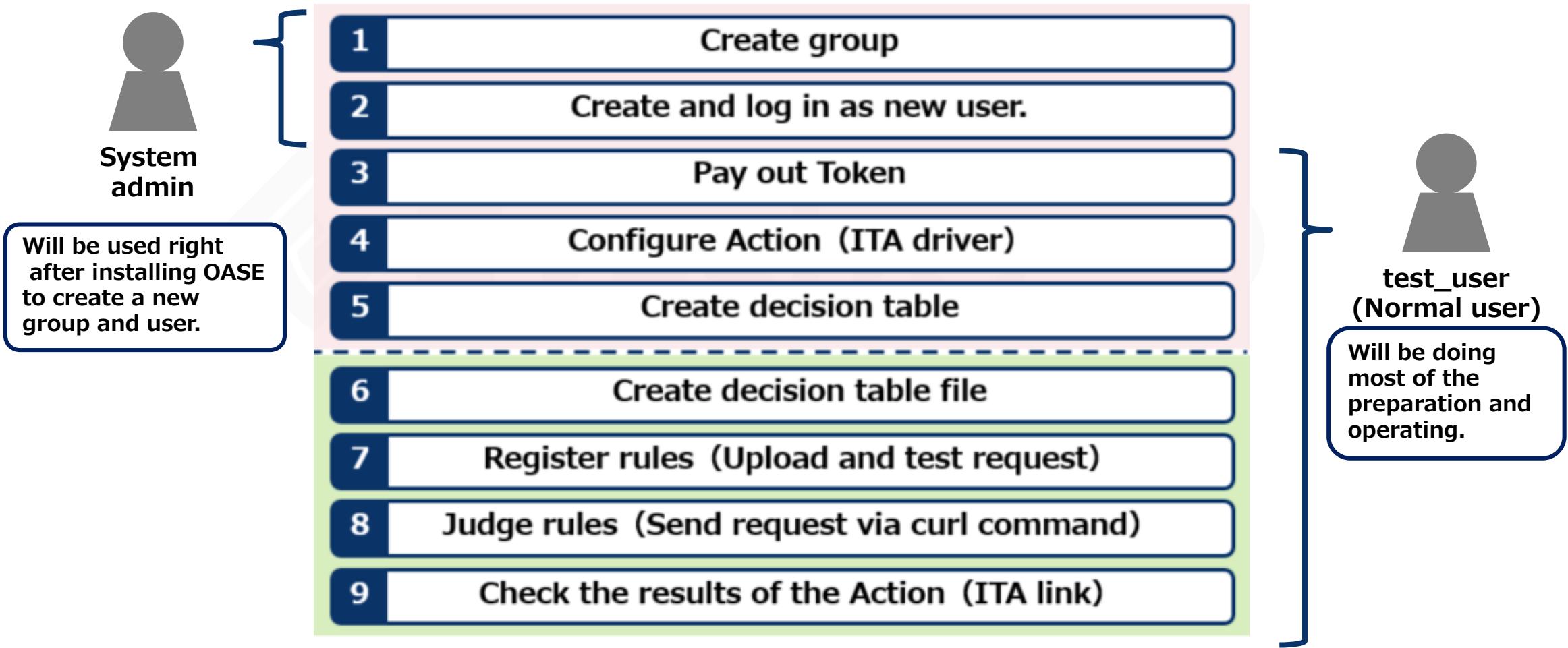
[Operation]

Creating and registering rules
Insert message
Rule matching
and
Executing

- 6 Create decision table file**
- 7 Register rules (Upload and test request)**
- 8 Judge rules (Send request via curl command)**
- 9 Check the results of the Action (ITA link)**

2.1 Scenario description (2/2)

In this scenario, the user will start logged in as the System admin and will then proceed to create a new group and user (test_group, test_user) that will be used for the rest of the scenario.



3. Preparation

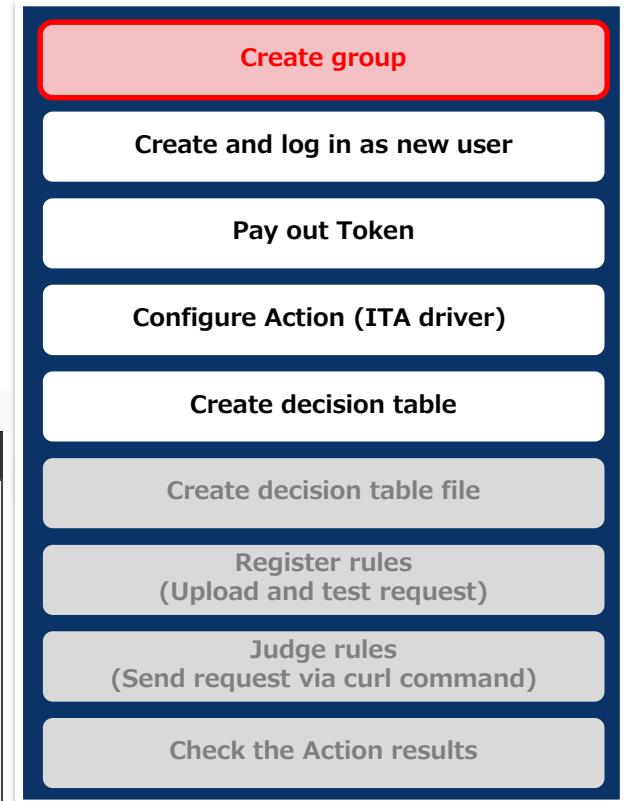


3.1 Create group

Create a new group called “test_group”

- ① Log in as the system administrator
- ② Move to the “group” screen and create a new group called “test_group”
- ③ Configure the “test_group”s access permission settings.

The screenshot shows the Exastro UI interface. At the top, there is a navigation bar with tabs: Dashboard, Rule, System (which is selected), and Management. On the right side of the header, there is a user dropdown labeled "administrator" with a gear icon, which is circled in red with the number "1". Below the header, the main content area has a title "Group" and a sub-section "Overview". A table row is highlighted with a green background, containing the text "test_group" in a white box, which is circled in red with the number "2". To the right of the table, a modal window titled "+ Access permission" is open. This modal has a sub-section "Setting permissions" with a table showing "Group name: test_group". The main part of the modal is a grid table with columns: Category, Screen, Function, No permission (radio button), Reference only (radio button), and Can perform update (radio button). The rows are categorized into Rule and System. Under Rule, "Decision table" and "Pay out Token" have "Can perform update" checked. Under System, "System settings", "SSO Settings", "Group", "User", and "Action settings" all have "Can perform update" checked. A callout box with a red border and the text "Input the following values" points to the "Group name" entry in the table. Another callout box with a red border and the text "Input the following values" points to the "Access permission" entry in the table, with the note "“Can perform update” for everything." A red circle with the number "3" points to the bottom right corner of the "Access permission" table.



3.2 Create and log in as new user

Create a new user

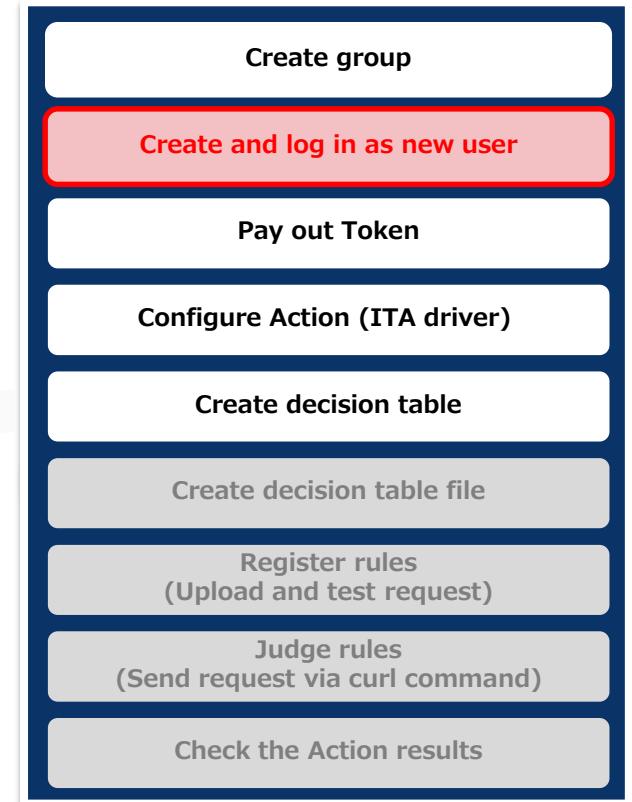
- ① Move to the “user” screen while logged in as the system admin
- ② Create a new user called “test_user”
- ③ Log out from the system admin account and log in as “test_user”

The screenshot shows the 'User' management screen. A red box highlights the input fields for 'User name', 'Login ID', 'Email address', and 'Group'. A red circle labeled '1' points to the 'administrator' user in the top right corner. A red circle labeled '2' points to the 'Input the following values' table below.

Item	Setting value
User name	test_user
Login ID	test_user
Mail address	(A mail address that can receive mail)
Group	test_group

POINT

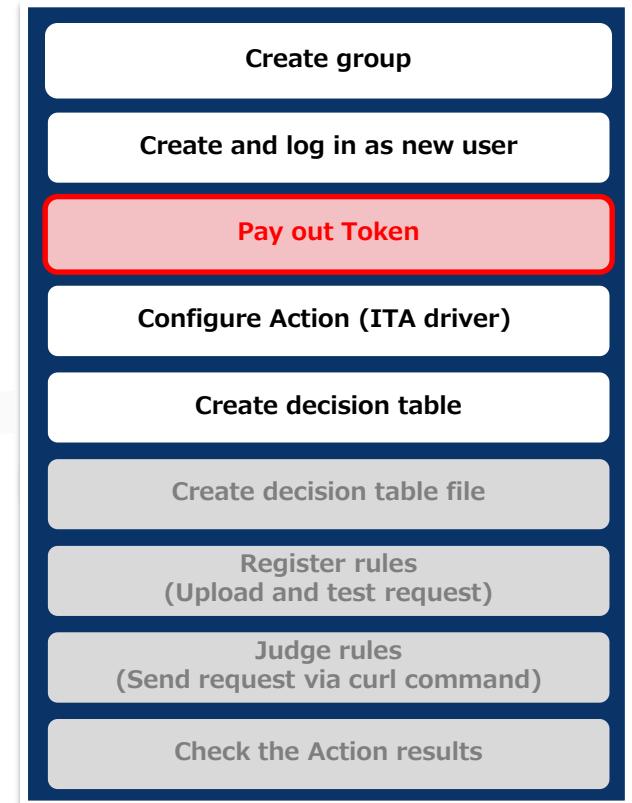
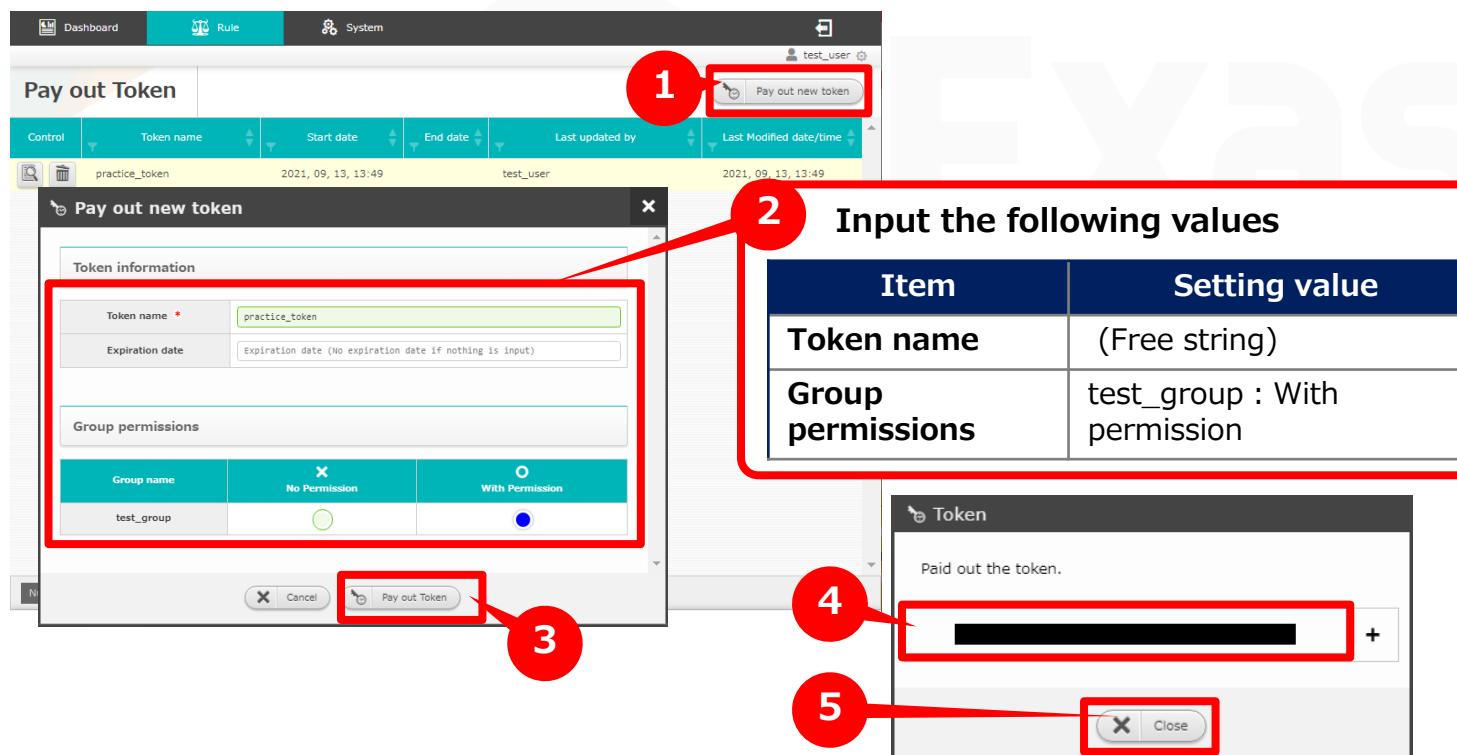
The user will receive two different mails when registering a new user in OASE. “Password notification” and “Login ID Notification”. Use the login ID and password written in said mails to log in to OASE.



3.3 Pay out Token

Pay out a new Token

- ① Press the “Pay out new token” button
- ② Input all the required information
- ③ Press the “Pay out Token” button
- ④ Copy the token displayed on the screen
- ⑤ Press the “Close” button



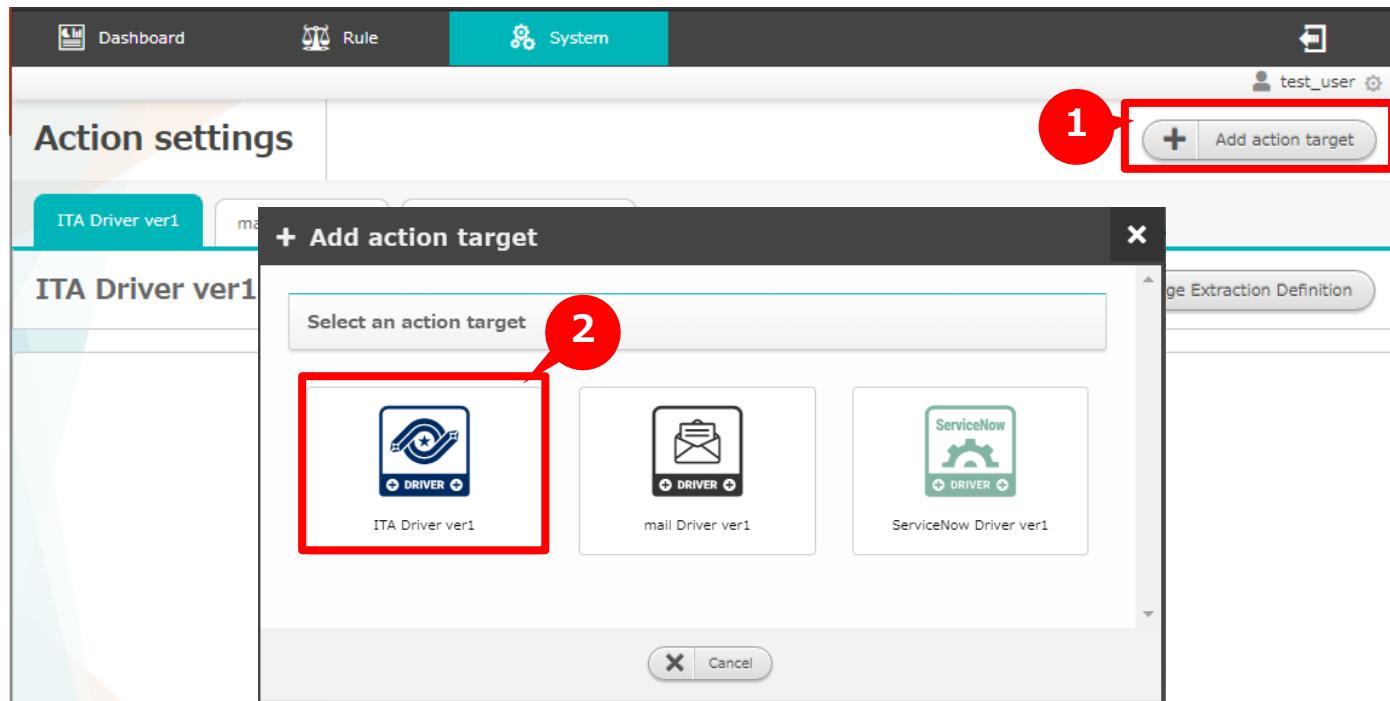
POINT

We will need a token when we are configuring the settings in chapter "4.3 Judge rules (Send request via curl command)".

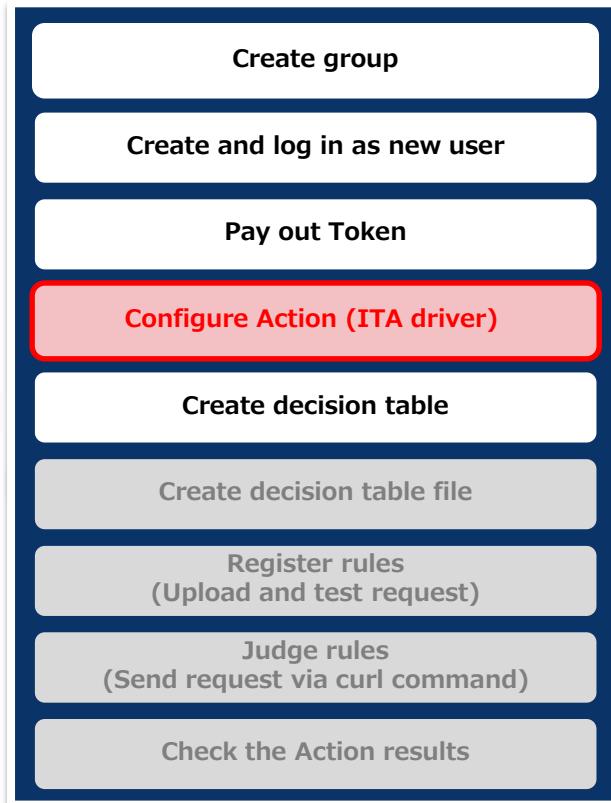
3.4 Action settings (ITA driver) (1/4)

Add Action destination

- ① Go to the “Action settings” screen and press the “Add new action destination” button.
- ② Select ITA Driver ver1.



※ The screen above will not display if you don't have any drivers installed.



POINT

Make sure to read the following manual and install the Mail driver in advance.
[Environment construction manual - Install drivers ->](#)

3.4 Action settings (ITA driver) (2/4)

Configure Action destination

- ① Input all the necessary information in the ITA Driver ver1 window.
- ② Press the “Save” button.

+ ITA Driver ver1

Name *	exastro-ita
Version *	1.7.0
Protocol *	http
Host / IP *	xxx.xxx.xxxx
Port *	80
User name *	administrator
Password *

Set permissions

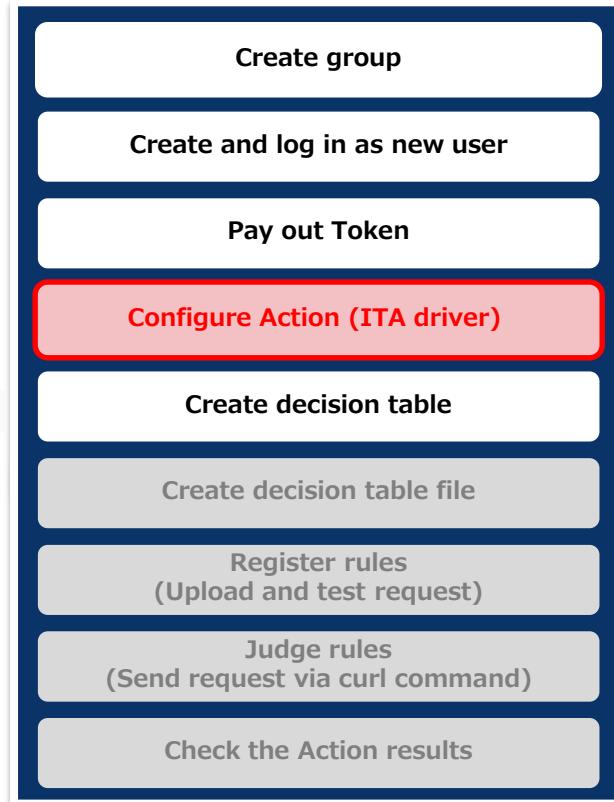
test_group

Permissions	No permission	Reference only	Can perform update
Register	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

X Cancel Back Save Register

1 Input the following values

Item	Setting value
Name	(Free string)
Version	1.7.1
Protocol	http
Host/IP	(IT Host name or IP Address)
Port	80
user name	administrator
Password	(ITA user password)
Permission settings	test_group : "Can update"



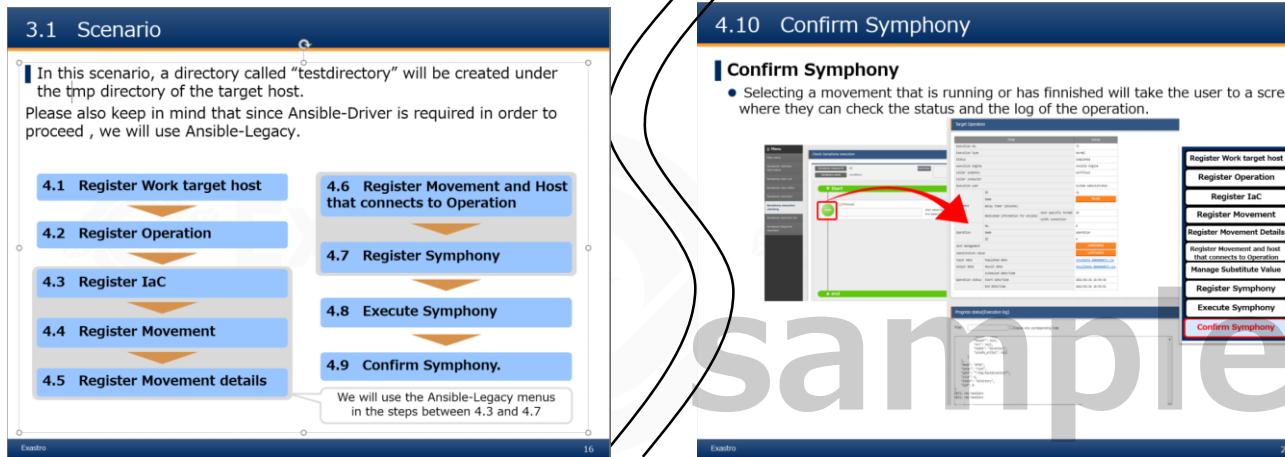
POINT

The "Name" will be used later in the <[Create decision table file](#)> when we are specifying the destination of the action.

3.4 Action settings (ITA driver) (3/4)

Change the ITA registration settings

- ① Log in to ITA and follow the <[Learn ITA BASE 【Practice】](#)>document from the “Scenario” slide to the “Symphony confirmation” slide.
※The chapter name and pages might change.



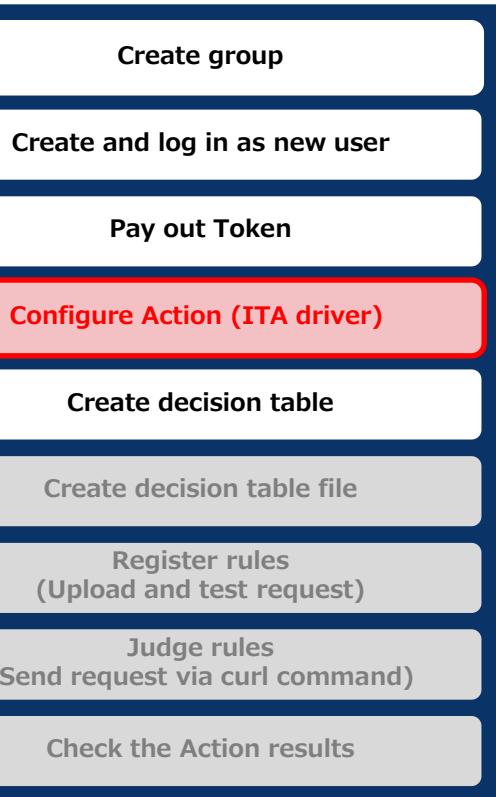
- ② Go to the ITA “Ansible-Legacy” menu group > “Substitution value list” menu > “List/Update” submenu and change the “specific value”



2

Input the following values

Item	Setting value
Specific value	(Free string)



POINT

The string input for the specific value will be the name for the new directory that will be created.

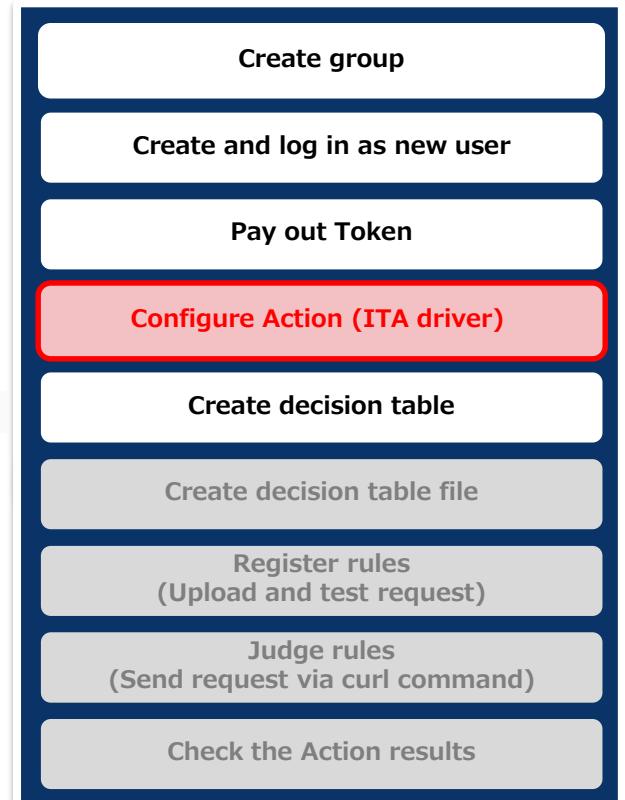
3.4 Action settings (ITA driver) (4/4)

In order to link ITA and OASE, we will need to configure the following settings:

- Go to "Management console" Menu group> "Role Menu link list" Menu > "List/Update" Sub menu and revive "Symphony link movement list".

The screenshot shows the Exastro Management Console interface. The left sidebar has a 'Role - Menu link list' item highlighted with a red box. The main area has two tabs: 'Description' and 'List/Update'. The 'List/Update' tab is active, showing a table with columns: History, Update, Discard, Item No., Role, Menu group, ID, Name, ID, Name, Associate, Last update date/time, and Last updated by. One row in the table is highlighted with a red box and labeled 'Movement associated with Symphony list'. A red arrow points from the 'Restore' button in the table to this row. Another red arrow points from the text 'Press the "Restore" button' to the 'Restore' button.

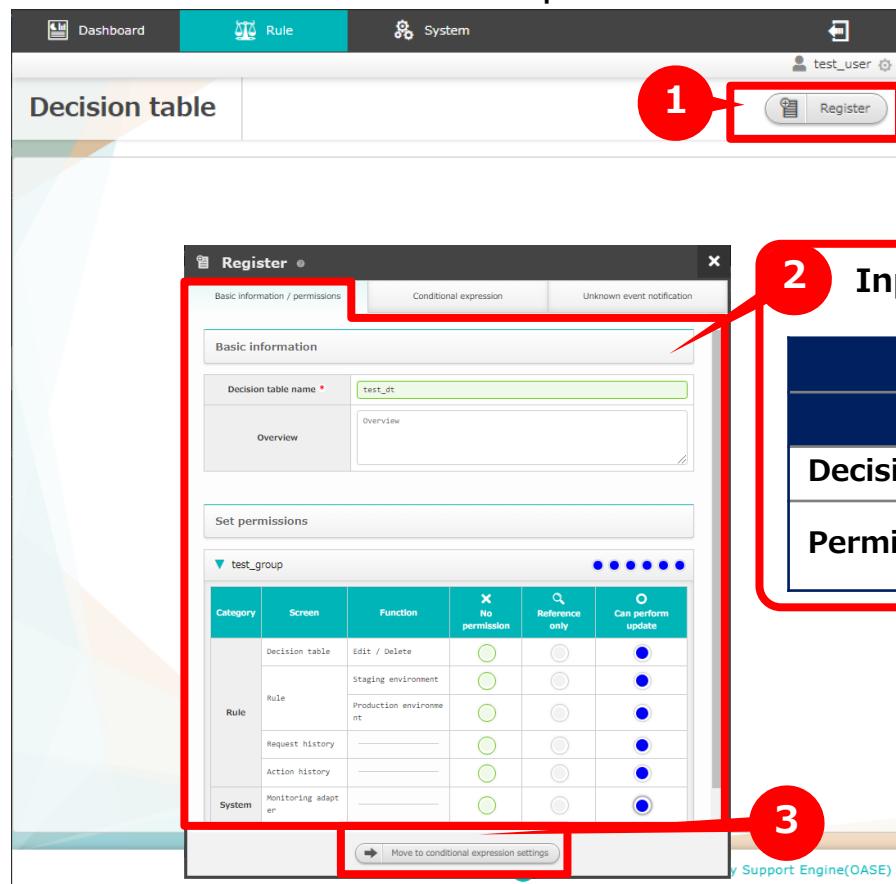
History	Update	Discard	Item No.	Role	Menu group	ID	Name	ID	Name	Associate	Last update date/time	Last updated by
History	Update	Restore	2100100001	System Administrator	Symphony	2100100001	Movement associated with Symphony list	2100000311	View only	2015/04/01 10:00:00	System Administrator	



3.5 Create decision table (1/2)

Create decision table

- ① Go to the "Decision table" screen and press "Create new" button
- ② Input all the necessary information in the "Create new" screen-> "Basic information/ permission" tab.
- ③ Press the "Condition expression" button



2 Input the following values

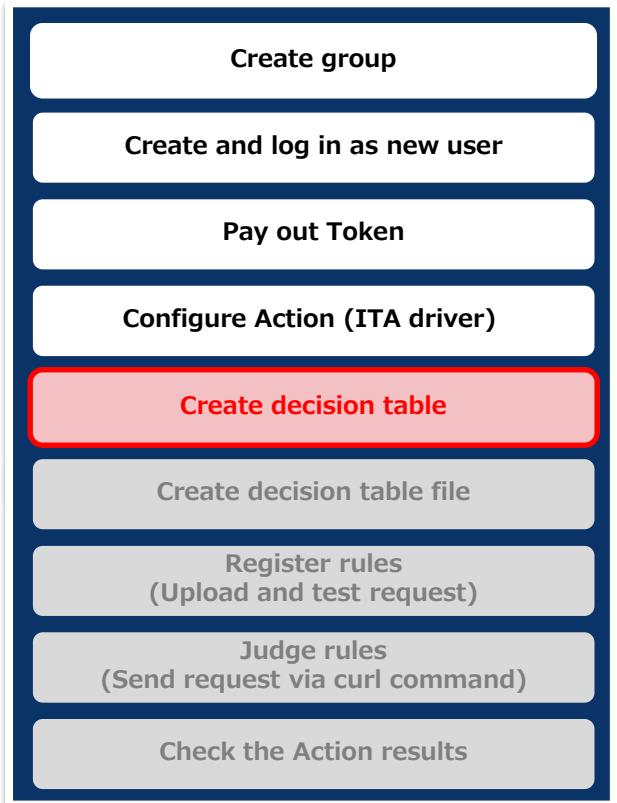
Basic information / permission tab

Item	Setting value
Decision table name	(Free string)
Permission settings	test_group : "Can perform update" for all.

3

POINT

Make sure to configure settings for at least one group in the "Permission settings". You will not be able to make changes to the Decision table if you don't.



3.5 Create decision table (2/2)

Create decision table

- ④ Input the necessary information in the “Create new” screen -> Conditional branch tab.
- ⑤ Press the “Unknown event notification” button
- ⑥ Input all the necessary information in the “Unknown event notification” tab
- ⑦ Press the “Save” button

The screenshot shows the 'Create new' screen with two tabs highlighted by red boxes: 'Conditional expression' (step 4) and 'Unknown event notification' (step 6). The 'Conditional expression' tab contains fields for 'Condition name' (Free string) and 'Conditional expression' (Pulldown selection). The 'Unknown event notification' tab contains fields for 'Unknown event notification' (Select "Notify by mail") and 'Notification E-mail address' (Mail address that can receive e-mails).

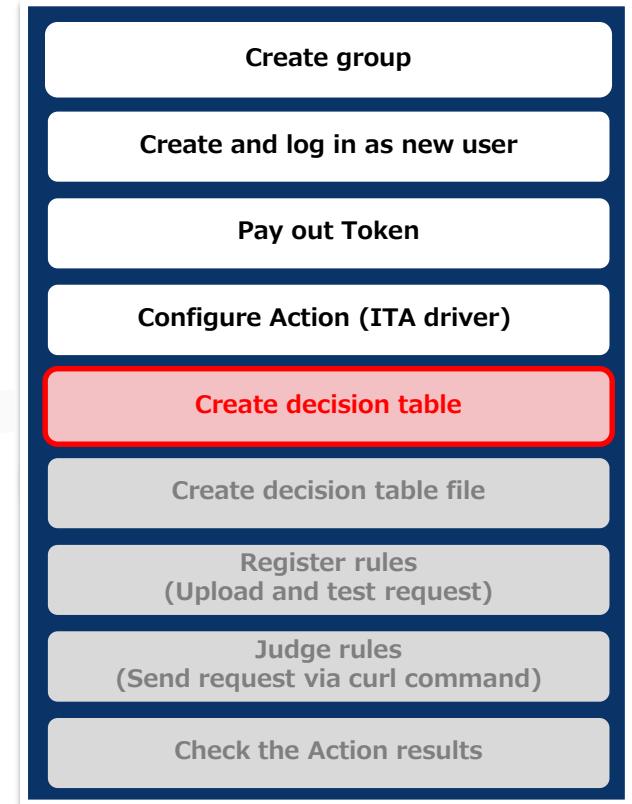
Conditional expression tab

Item	Setting value
Condition name	(Free string)
Conditional expression	Pulldown selection

Unknown event notification tab

Item	Setting value
Unknown event notification	Select "Notify by mail".
Notification E-mail address	(Mail address that can receive e-mails)

Red numbered circles indicate the steps: 4 (Conditional expression tab), 5 (Unknown event notification button), 6 (Unknown event notification tab), and 7 (Save button).



POINT

The configured conditional expression will be used later when we are setting specific values in the Decision table file's "Condition part".

4. Operation

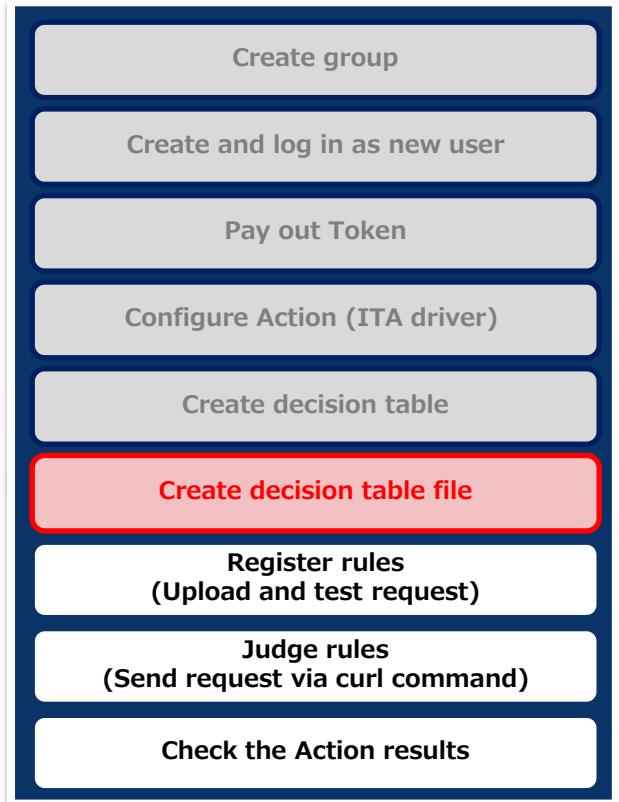


4.1 Create decision table file (1/3)

Download and create the decision table file

- Click the download button the right of the decision table we created in chapter 3.2 and download the decision table file

operation	Decision table name	Last updated by	Last Modified date/time
	test_dt	test_user	2021, 09, 13, 16:54



POINT

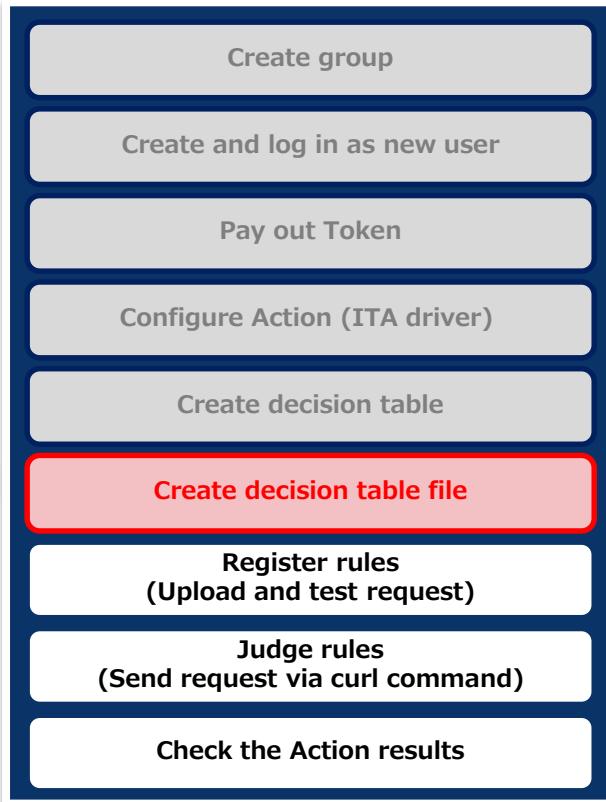
The name of the decision table file is automatically generated (E.g."id0000000000.xlsx"). Note that it will be different from the name of the decision table itself. In the next page, we will explain the contents of the file.

4.1 Create decision table file (2/3)

Fill in the Decision table file

For an exact copy of the following decision table file, please see [A. Appendix Sample1](#).

- | | |
|------------------------|--|
| ①Comment part | Used for comments and descriptions. Can also be blank |
| ②Condition part | Creates the condition that will be matched to the rules. |
| ③Action part | <ul style="list-style-type: none">Can set different actions for each rule name.The only type of "Action type" that can be used are the ones that have their drivers installed in the "Action settings" screen.Note that the writing format of the "Action parameter information" depends on the type of action.Can be used to perform an action or to send a validation mail. |
| ④Action condition part | Used to specify the validity period. Can also be blank |



POINT

For all the different values and how to correctly write them, please see the "example" sheet in the excel file.

After changing the Decision table file, you can change the name of the file name to your liking.

4.1 Create decision table file (3/3)

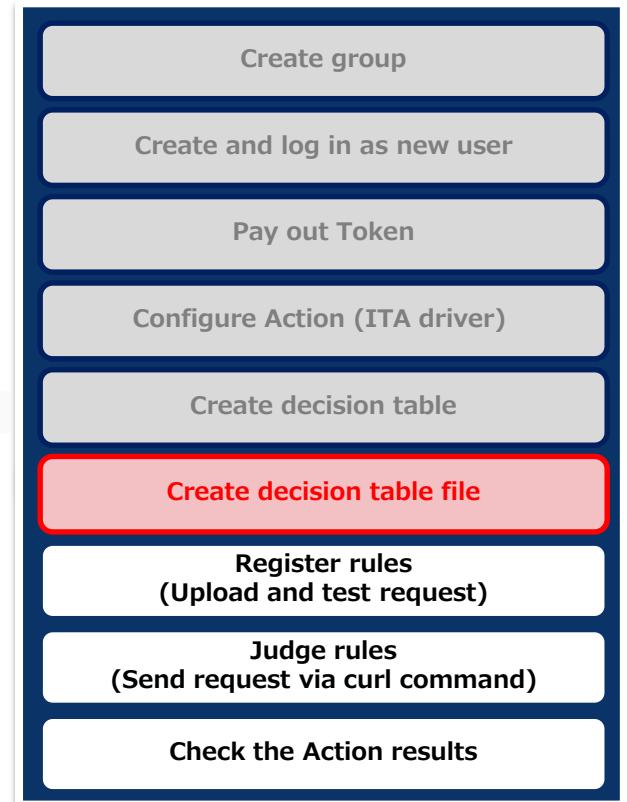
Input Action parameter information

- Use the table below and fill out the Action parameter Information

No.	Action parameter information	Setting value
1	ITA_NAME	The name registered in the "ITA Driver ver1" name tab.
2	SYMPHONY_CLASS_ID	The Symphony class ID (Is found in ITA's "Symphony" menu group -> "Symphony execution" menu -> "Symphony [list]")
3	OPERATION_ID	The Operation ID (Is found in ITA's "Symphony" menu group -> "Symphony execution" menu -> "Operation [list]")

The diagram illustrates the mapping of action parameters from the table to the ITA Driver and Symphony interfaces. Red arrows point from the table rows to specific fields in each interface.

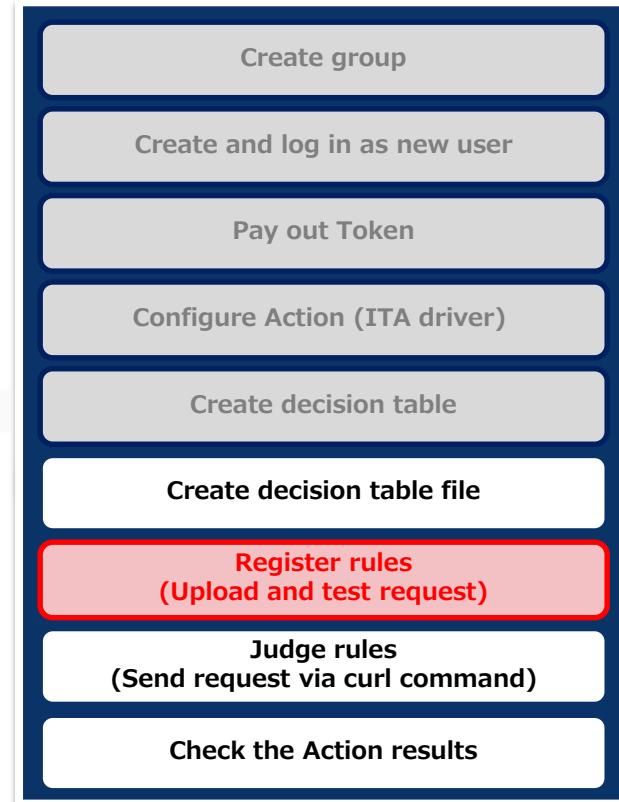
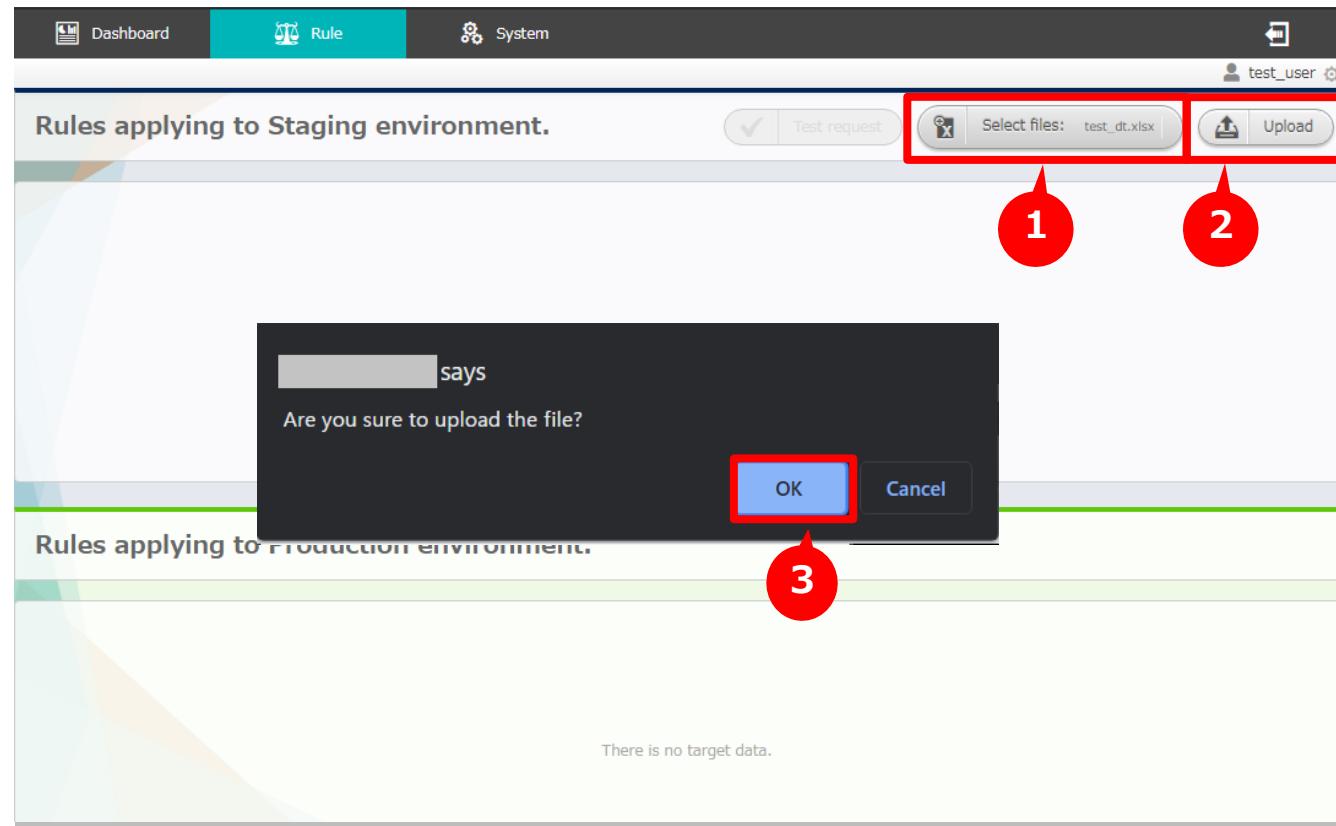
- Action parameter information (Required) row:** Points to the **ITA Driver ver1** interface. Red arrow ① points to the **Name** field where **exastro_ita** is entered. Red arrow ② points to the **Symphony** interface, specifically the **Symphony [List]** table where the **symphony class ID** is set to **2**.
- ITA_NAME=exastro_ita, SYMPHONY_CLASS_ID=2, OPERATION_ID=28 row:** Points to the **Symphony** interface. Red arrow ③ points to the **Operation [List]** table where the **operation ID** is set to **28**.



4.2 Register rules (Upload and test request) (1/6)

Select the decision table you want to test

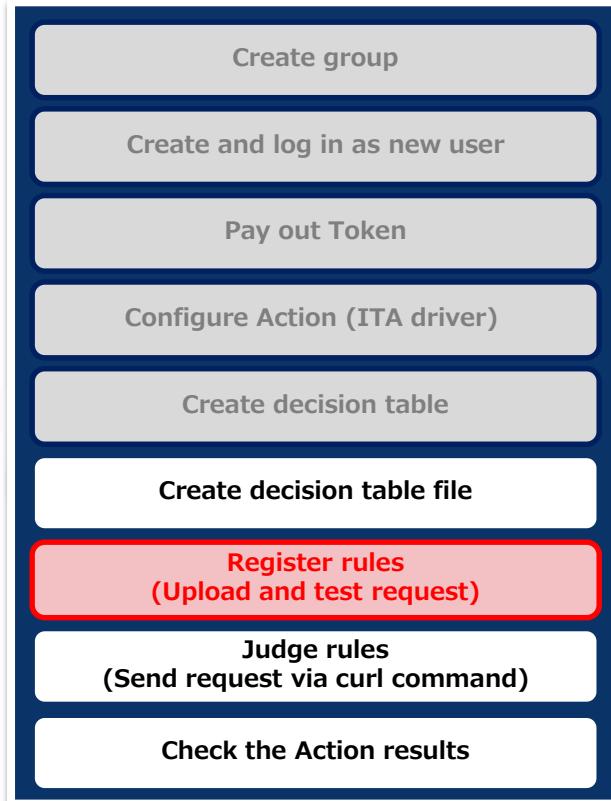
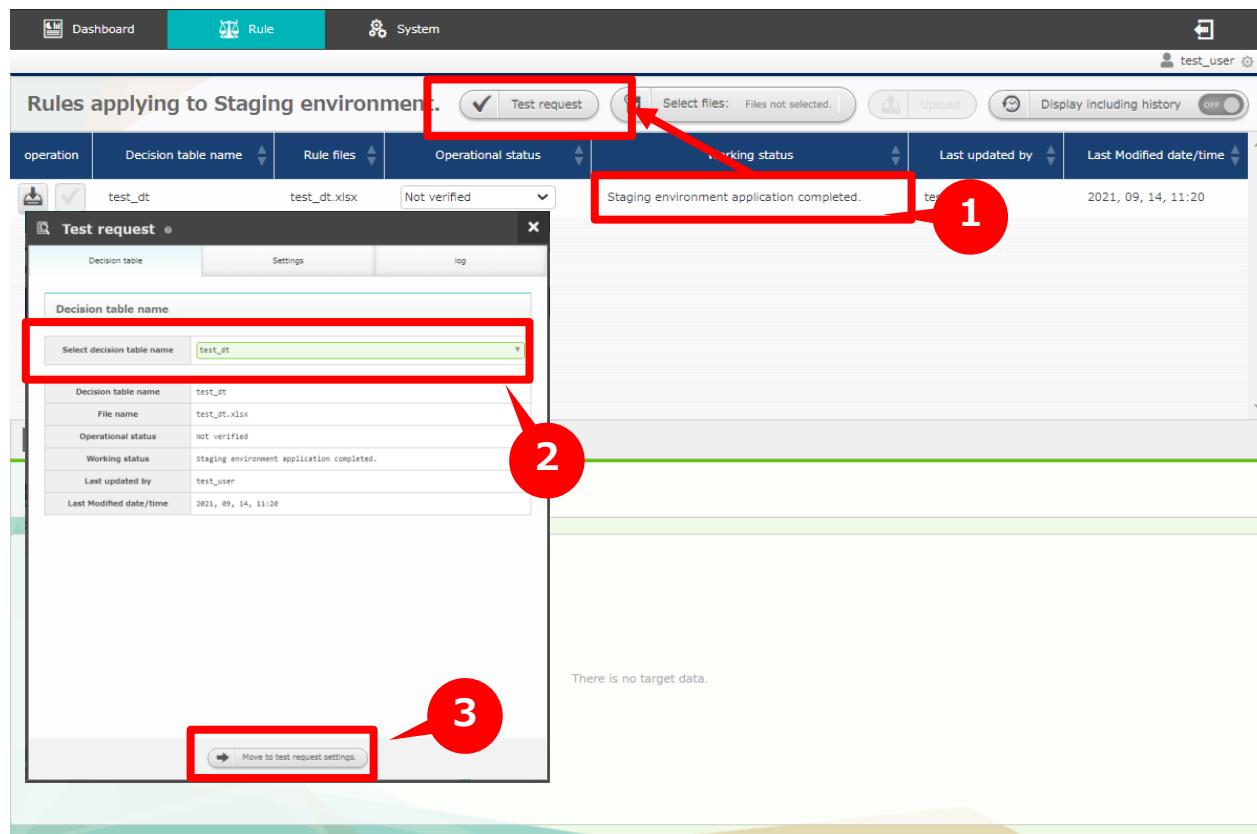
- ① Click the “Upload” button in the Rule screen and select the previously created decision table file
- ② Press the “Upload” button
- ③ Press “OK” on the browser dialogue box



4.2 Register rules (Upload and test request) (2/6)

Select test request target

- ① After the “Operation status” has changed to “Staging environment application completed”, press the “Test request” button.
- ② Select the decision table file you want to test.
- ③ Press the “Test request” button.



POINT

The Operation status changes automatically every 5 seconds. For more information regarding the different statuses, please refer to [<User manual - Rule screen- \(1\) Rule screen \(Staging\) >](#)

4.2 Register rules (Upload and test request) (3/6)

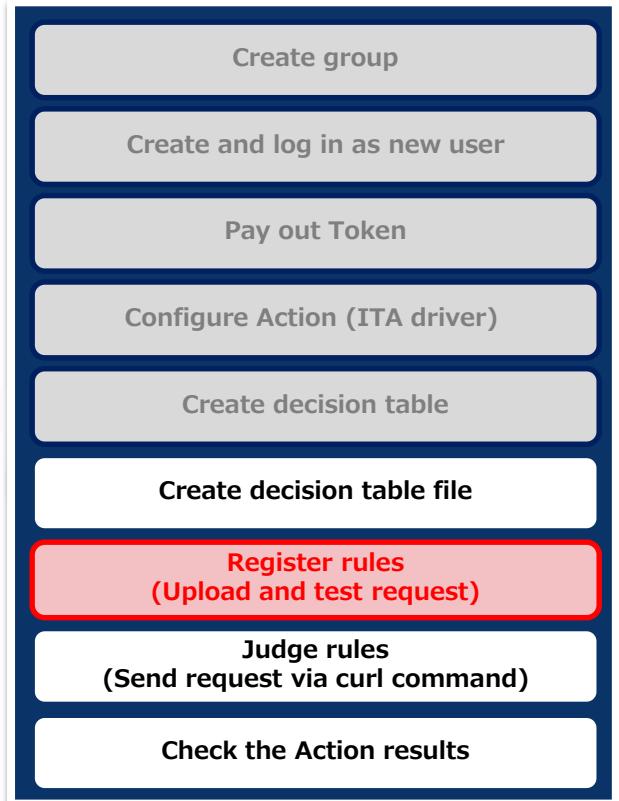
Input test values and run the test request

- ① In the settings tab, select “Bulk test”
- ② Press the “Download Excel files for bulk testing” button
- ③ In the file, input a value that matches the rule you previously created

The screenshot shows the 'Test request' interface with the 'Settings' tab selected. A red box highlights the 'Bulk test' button under 'Test request settings'. Another red box highlights the 'Download Excel files for bulk testing' button. A third red box highlights the Excel spreadsheet area where three rows of data are listed:

Request name	Event start date/time	Alert level	target
For rule 1	2019-5-17 1:20:30	Error:	HDD usage 80% over
For rule 2	2019-5-17 1:20:30	Warning:	memory usage 80% over
For rule 3	2019-5-17 1:20:30	[info]	HDD usage 20% over

A note at the bottom of the spreadsheet says: ↑ If you want to add more requests, copy 4th row and insert more.



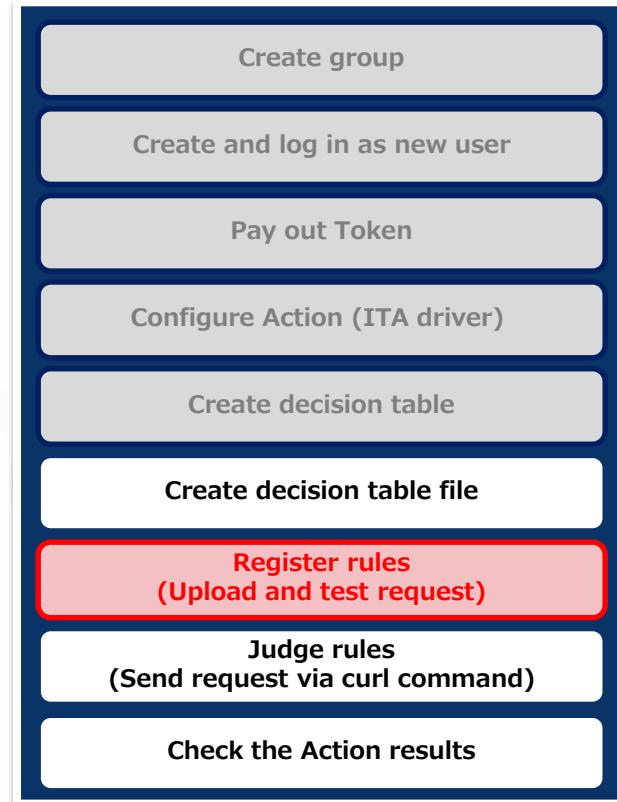
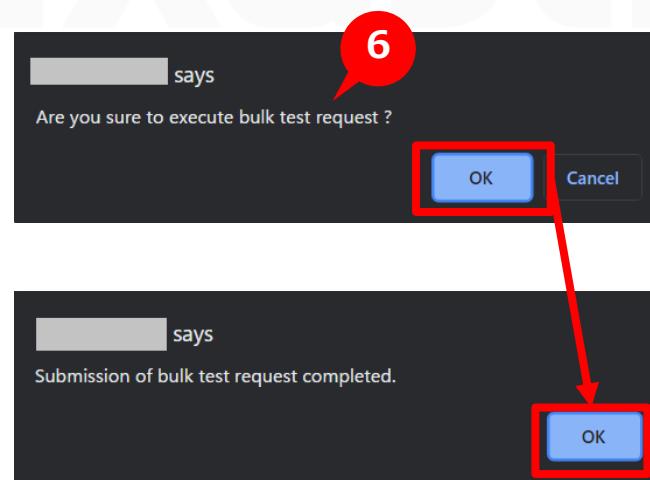
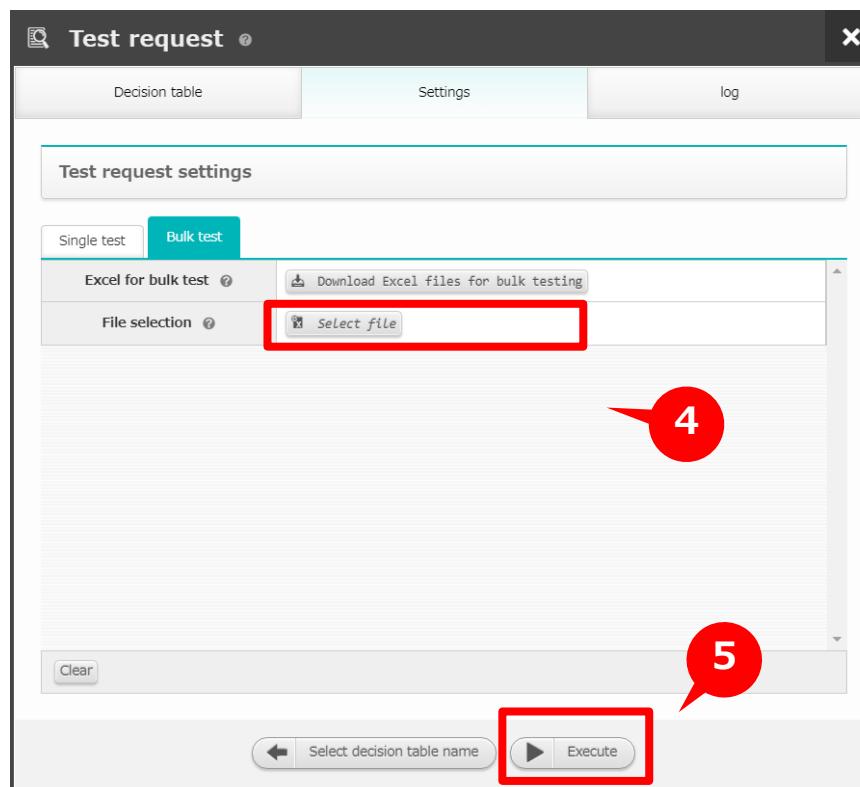
POINT

We will test if the values matches the condition part in the decision table file.

4.2 Register rules (Upload and test request) (4/6)

Input test values and run the test request

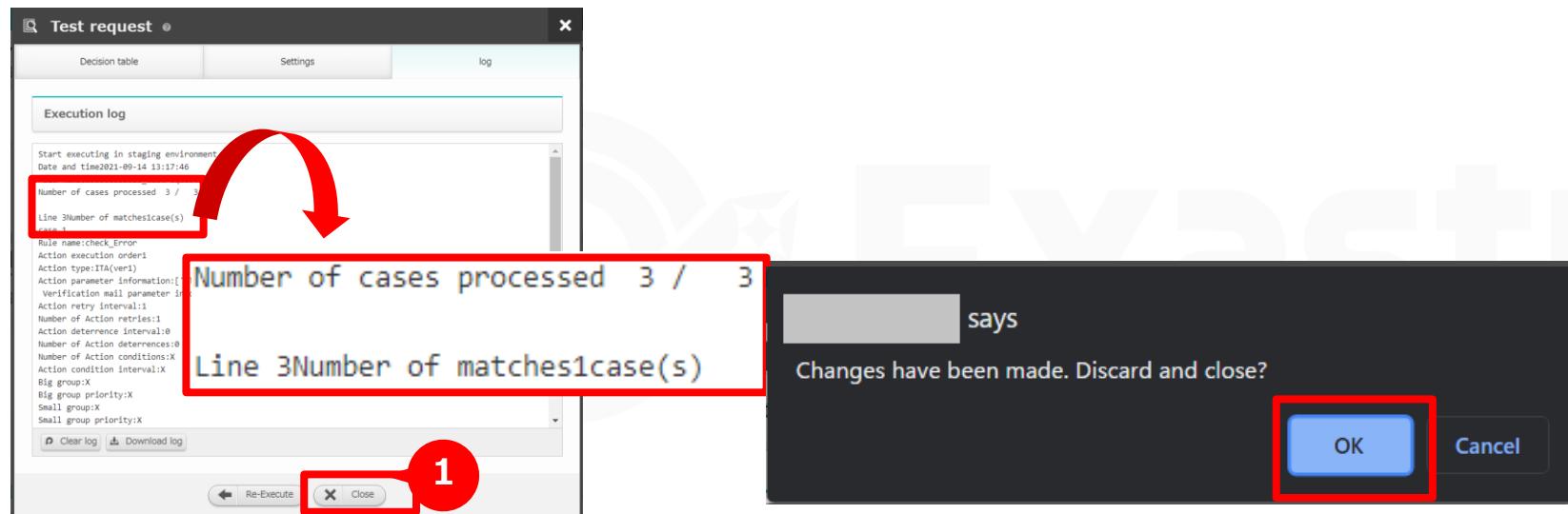
- ④ Press the "Select file" button and select the updated Excel file xx
- ⑤ Press the "Execute" button
- ⑥ Press "OK" on the browser dialogue box



4.2 Register rules (Upload and test request) (5/6)

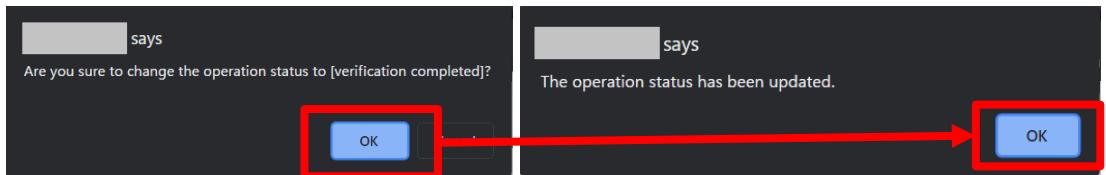
Check that the rule is active

- Check the execution log in the log tab
 - Press the “Close” button
 - Press “OK” on the browser dialogue box



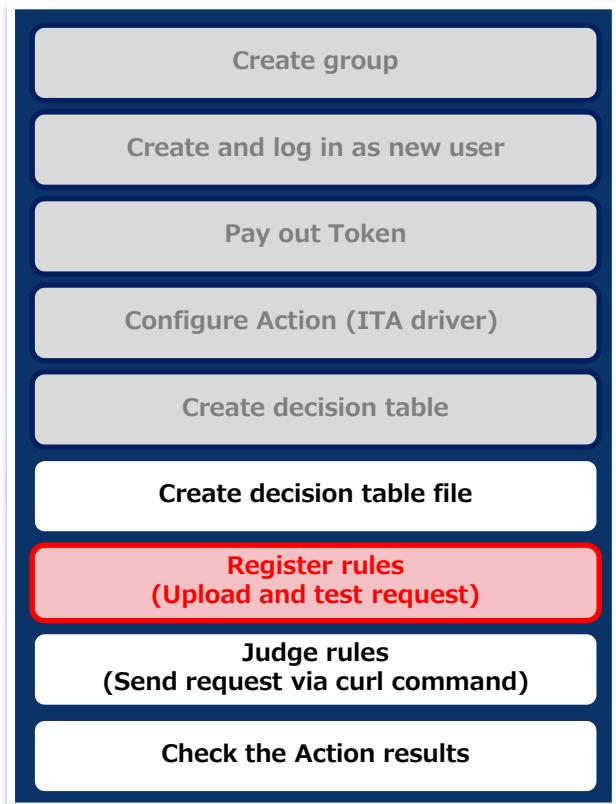
- If the rule matched

- Press “OK” on the browser dialogue box



POINT

If test matches the rule we created in chapter 4.1, the execution log will display “Line X Number of matches X case(s)”
The decision table will change status to Verification completed when it's rule is hit.

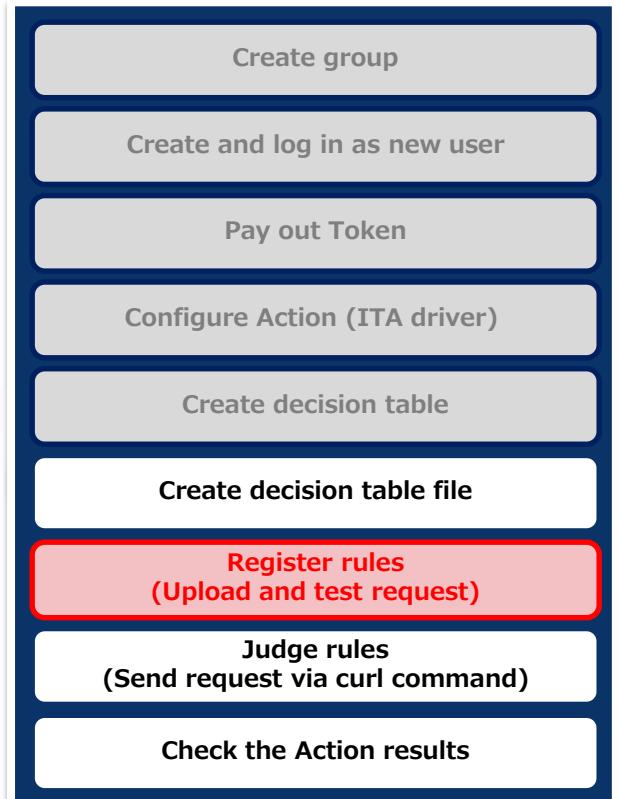
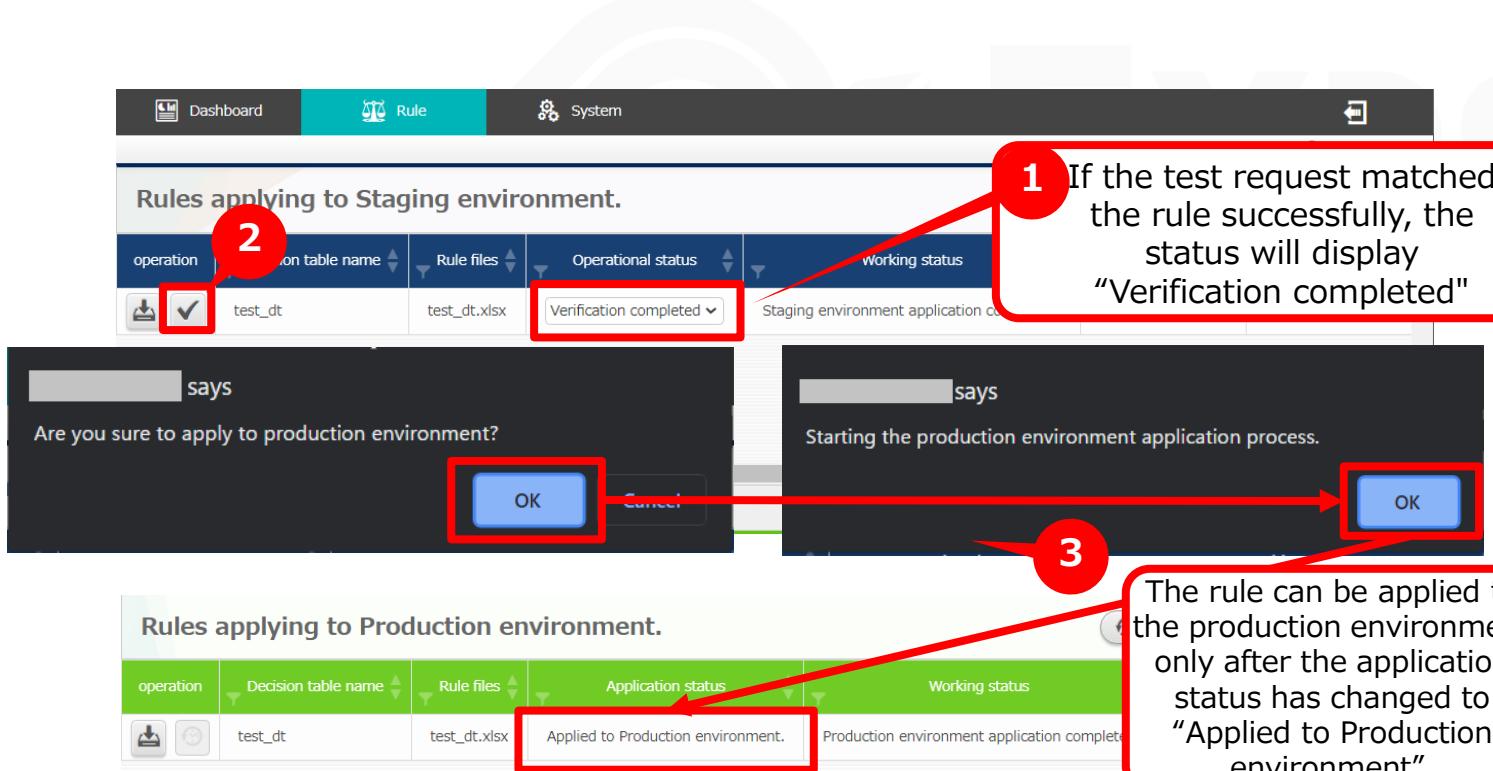


4.2 Register rules (Upload and test request) (6/6)

Make the tested rule valid for production environment

- In order to use the rule in the production environment, we will need to change the status from "Not applied" to "Verification completed"

- Check that the status has changed status to "Verification completed"
- Press the "Apply" button
- Press "OK" on the browser dialogue box



POINT

The operation status is automatically updated every 5 seconds. for more information regarding the different statuses, please see <[User manual - Rule screen- \(2\) Rule screen \(Production\)](#)>

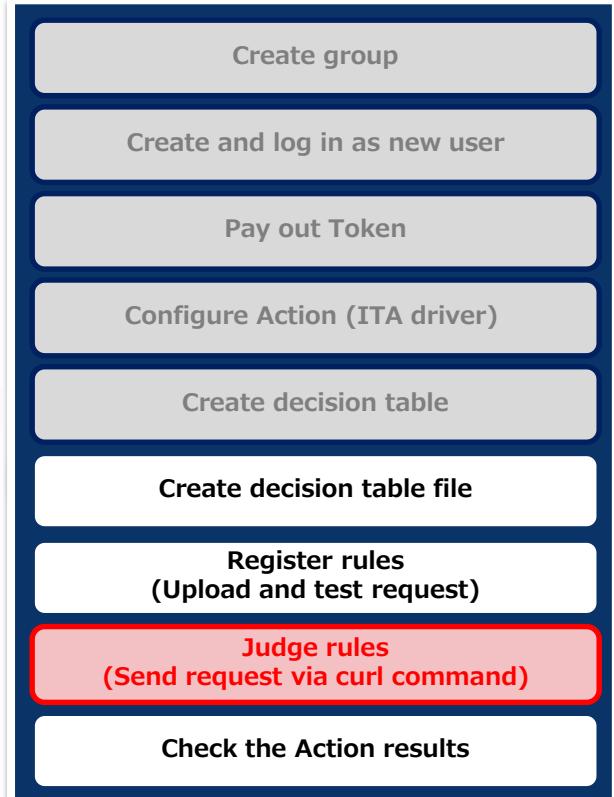
4.3 Judge rules (Send request via curl command) (1/2)

Input a message and match the production applied rule xx

- Open the terminal and change the following command to fit your rule and run it.
- See "A. Appendix Sample 1" for a more detailed example of the curl command.

```
curl -X POST -k "https://<①Host name>/oase_web/event/event/eventsrequest" ¥  
      -H "accept: application/json" ¥  
      -H "Authorization: Bearer <⑥Token>" ¥  
      -d '{"decisiontable":"<②Decision table name>","request type":"<③Request  
type>","eventdatetime":"<④Event start date>","eventinfo":["<⑤Event information>"]}'
```

① Host name	Input a valid host name or IP address
② Decision table name	Input a decision table applied to the production environment
③ Request type	Input the request destination type, "1" E.g.) "requesttype":"1"
④ Event start date	Input the event start date in the following format: yyyy/mm/dd hh:mm:ss E.g.) "eventdatetime":"2020/01/01 01:01:01"
⑤ Event information	Specify the event information in a list format. E.g.) "2","AAA"
⑥ Token	Input the token created at the "Pay out Token" screen



POINT

For more information regarding HTTPS requests, please see the
[<REST API Function User manual>](#)

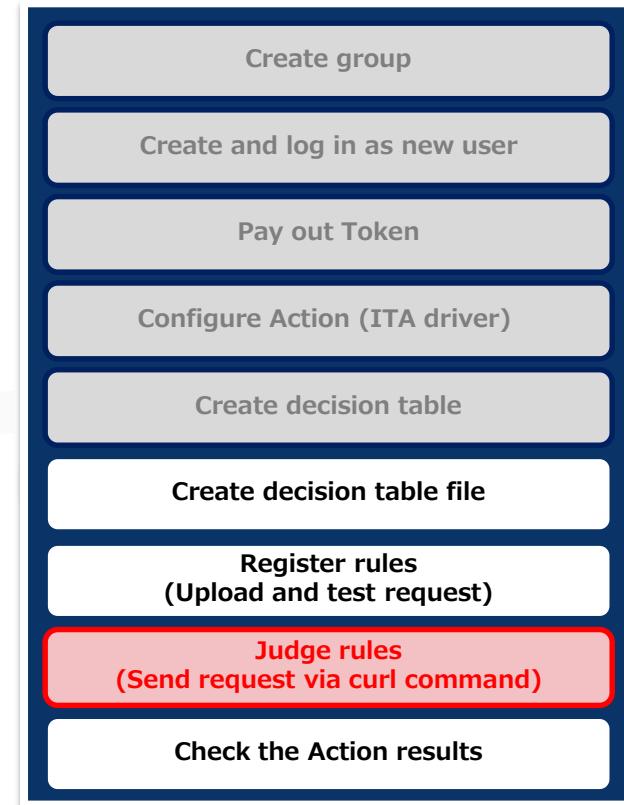
4.3 Judge rules (Send request via curl command) (2/2)

Check the Request history screen

- The request sent using the curl command should be displayed
- You can choose what items to display by pressing the ■ button.

The screenshot shows the 'Request history' screen with a table of five rows. The columns are: Rule matching status, Request type, Decision table name, Request received date/time, and Event information. A red box highlights the '■' button at the bottom of the table's column headers, which opens a dropdown menu. The dropdown menu lists eight items, each with a checked checkbox: Rule matching status, Request type, Decision table name, Request received date/time, Event information, Event occurrence date/time, and Event serial number. The 'Event information' row in the table also contains JSON data related to HDD usage.

Rule matching status	Request type	Decision table name	Request received date/time	Event information
✓	Production environment	test_dt	Sept. 14, 2021, 3:40 p.m.	{"EVENT_INFO": ["Error:", "HDD usage 80% over"]}
✓	Production environment	test_dt	Sept. 14, 2021, 3:39 p.m.	{"EVENT_INFO": ["Error:", "HDD usage 80% over"]}
✓	Production environment	test_dt	Sept. 14, 2021, 3:38 p.m.	{"EVENT_INFO": ["Error:", "HDD usage 80% over"]}
?	Staging environment	test_dt	Sept. 14, 2021, 1:30 p.m.	{"EVENT_INFO": ["Error:", "HDD usage 20% over"]}
✓	Staging environment	test_dt	Sept. 14, 2021, 1:30 p.m.	{"EVENT_INFO": ["Error:", "Memory usage 80% over"]}



4.4 Check the Action results (1/3)

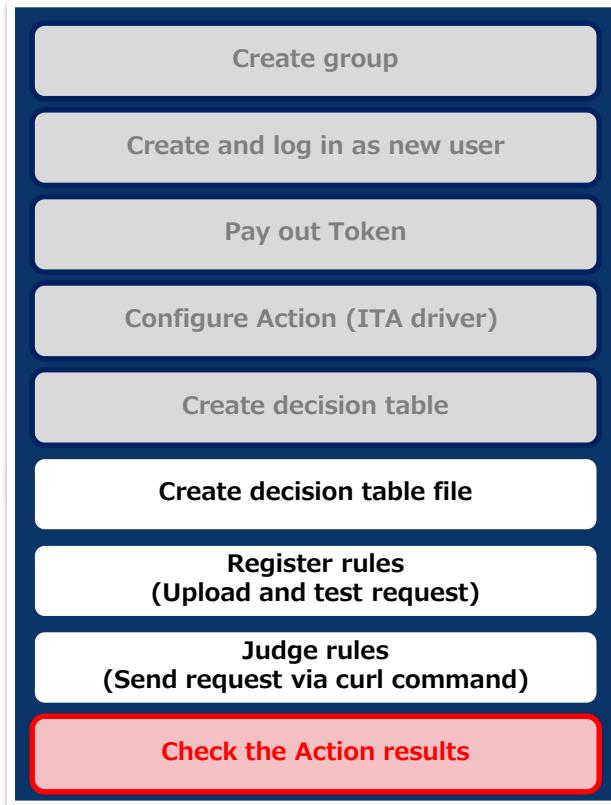
Run action

- If the rule matches, the action will be executed just as we set it to do in the preparation section. You can check the action in the Action history settings screen.
- The action is the results of the contents we configured in the action part in chapter 4.1 Create decision table file

Action History

- Check that the action executed by the rule matching is being correctly displayed in the action history screen.

Status	operation	Decision table name	Rule name	Action type	Last execution date and time	Last executed by
✓		test_dt	check_Error	ITA(ver1)	2021, 09, 14, 17:29	ActionDriverProcedure



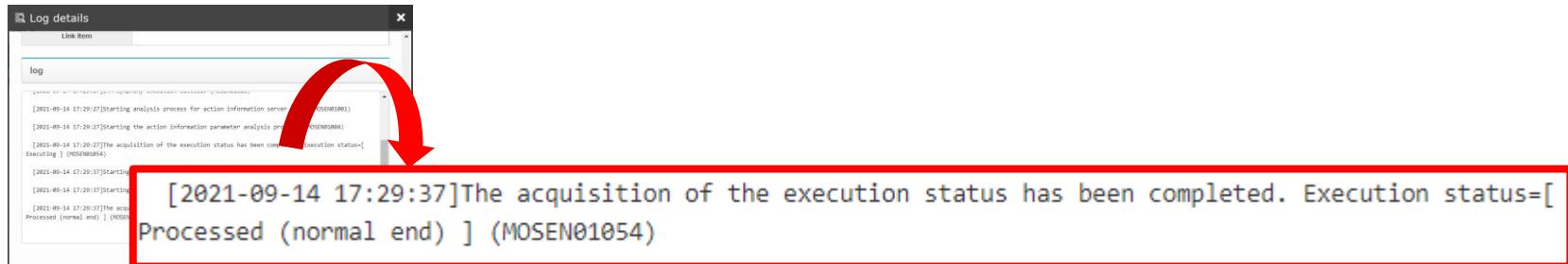
POINT

For more information regarding the different icons and buttons, please see [<User manual -Action history->](#)

4.4 Check the Action results (2/3)

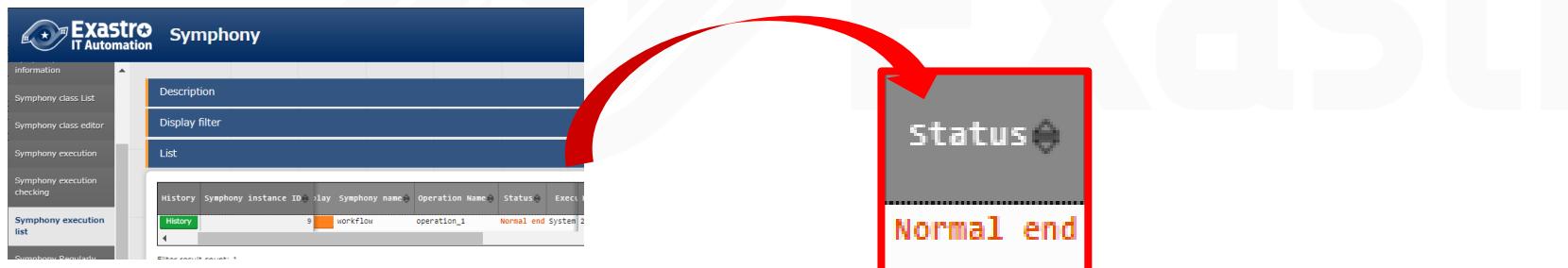
Check ITA has run the process

- Check the OASE log that the process has ended correctly



The screenshot shows a log entry from the OASE log details window. A red arrow points to the log entry, and a red box highlights the text: [2021-09-14 17:29:37]The acquisition of the execution status has been completed. Execution status=[Processed (normal end)] (MOSEN01054).

- Go to the ITA Symphony list and check that the status is displaying “Normal end”

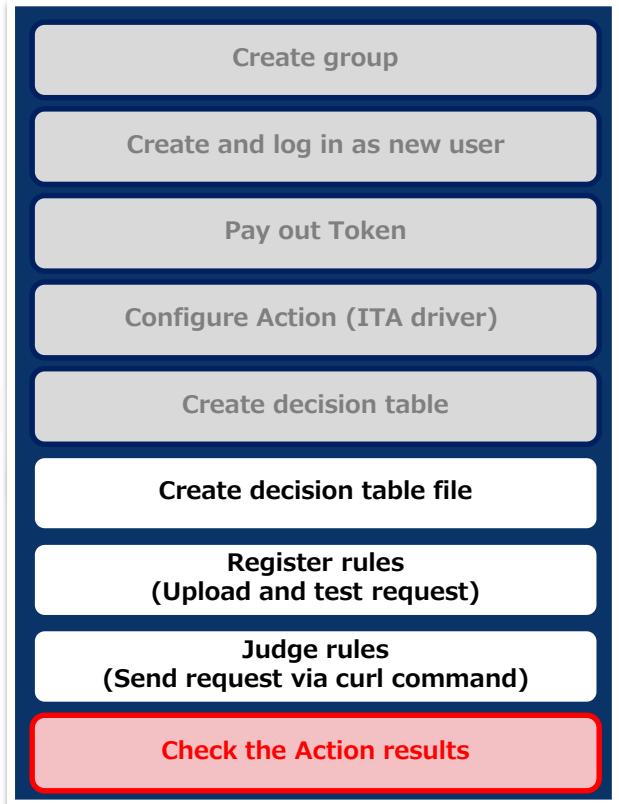


The screenshot shows the Exastro IT Automation Symphony list. A red arrow points to the status column, which displays "Normal end".

- Check that a new directory has been created according to the previously set contents



The screenshot shows two terminal sessions. The left session shows directory creation: tmp]# pwd, /tmp, tmp]# find ./ -name "*directory" -type d, ./testdirectory, tmp]#. The right session shows directory listing: tmp]# find ./ -name "*directory" -type d, ./testdirectory, ./oase_testdirectory, tmp]#.

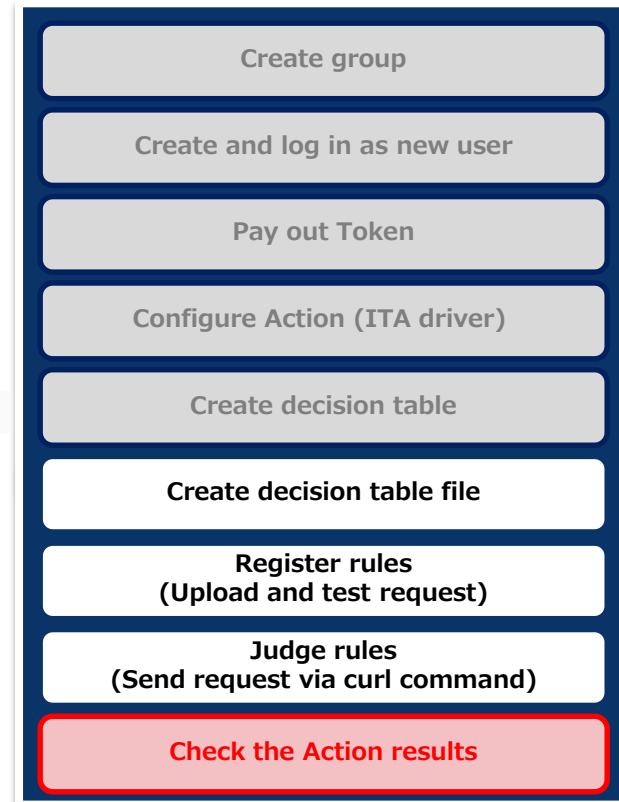


4.4 Check the Action results (3/3)

For Unknown events

- If a request with contents not defined as a known event is sent, a mail with the subject "OASE Unknown event notification" will be sent.

The screenshot shows the OASE interface with the 'Rule' tab selected. On the left, the 'Action history' table has a red box around the first row's status column, which contains a question mark icon. A red arrow points from this icon to the text 'Rule matching status : No rule detected' at the bottom of the table. The main panel displays an email from 'noreply@example.com' with the subject '[OASE]Unknown event notification'. The email body states: 'This mail is automatically sent from ""Operation Autonomy Support Engine"".' followed by a separator line, 'Event information that does not match the configured rules has been submitted to OASE. Please confirm, as this will be registered as an unknown event.' Below this, several event details are listed: [Decision table name] test_dt, [Event start time] 2019-12-31 16:01:01+00:00, [Request Reception time] 2021-09-14 08:38:15.953665+00:00, [Event Information] {"EVENT_INFO":["Error:","TestTest"]}, and [Trace ID] TOS_20210914083815917011_0000000031.



A Appendix



Input sample values and run OASE

- The following settings will make it so if a message containing either "Error:" or "HDD", ITA will start and create a directory called "oase_testdirectory"
- If a message that is not a known event (contains message: and "SKIP", A mail with the subject "[OASE]Unknown event notification" will be sent.

[Configuration]

① "Pay out Token" screen

- Create token that will be used when sending request through curl command.

Token name	practice_token
------------	----------------

(Register other necessary info.)

② "Action settings" screen

- Prepare "ITA Driver ver1"

name	test_ita
------	----------

Version	1.7.1
---------	-------

(Register other necessary info.)

③ "Decision table" screen

- Create the decision table file containing conditions the "Alert level" and "Target" will hit.

Decision table name	test_dt
Permission settings (test_group)	"Can update" for all"

Condition name	Conditional expression
Alert level	Matches the regular expression
target	Matches the regular expression

Unknown event notification	Notify by mail
----------------------------	----------------

(Register other necessary info.)

POINT

Used in the following chapters:

<[3.3 Pay out Token](#)>
<[3.4 Configure Action \(ITA driver\)](#)>
and
<[3.5 Create decision table](#)>

Sample 1 (2/4)

[Operation]

④ "Decision table" file

- Save the decision table file and rename it

File name test_dt.xlsx

- Create a rule that matches with: "Alert level : "Error"" "Target : "HDD""

Rule description	Alert level	Target	Rule name	Action type	Action parameter information
Rule 1	^.*Error.*\$	^.*HDD.*\$	check_Error	ITA (ver1)	ITA_NAME=test_ita,SYMPHONY_CLASS_ID=1,OPERATION_ID=1
Rule 2	^.*Warning.*\$	^.*memory.*\$	check_Warning	blank	X
Rule 3	^.*info.*\$	^.*%.*\$	check_Info	blank	X

(Use the "Example" sheet in the Decision table file as a reference and fill in the rest of the items)

⑤ "Rule (Stage application rule) " Screen

- Upload the edited decision table file

Select file

test_dt.xlsx

POINT

Used in the following chapters:

[<4.1 Create decision table file >](#) and

[<4.2 Register rules \(Upload and test request\) >](#)

Sample 1 (3/4)

⑥Test request screen

- Test what rule that gets matched if there are multiple requests sent to the decision table file.

Select decision table name	test_dt		
Bulk test request			
Request name	Event start date	Alert level	Target
For rule 1	2019-5-17 1:20:30	Error:	HDD usage 80% over
For rule 2	2019-5-17 1:20:30	Warning:	memory usage 80% over
For rule 3	2019-5-17 1:20:30	[info]	HDD usage 20% over
Select file	id000000000xx_testrequest.xlsx		

POINT

Used in this chapter: <[4.2 Register rules \(Upload and test request\)](#)>

⑦Terminal (Linux server)

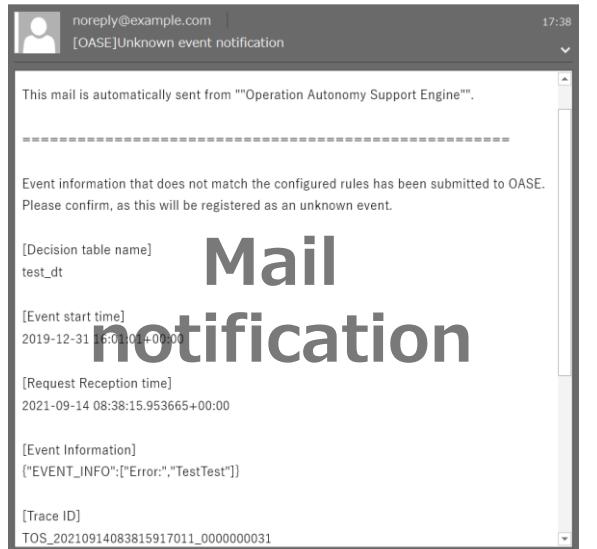
- Use a curl command to send a request to the rule we created. ***Change red text to fit your rule**

```
curl -X POST -k "https://<Hostname>/oase_web/event/event/eventsrequest" -H "accept: application/json" ¥  
-d '{"decisiontable":"test_dt","requesttype":"1","eventdatetime":"2020/01/01  
01:01:01","eventinfo":["Error:","HDD usage 80% over"]}' -H "Authorization: Bearer <Access_Token>"
```

⑧Unknown event notification

Check that you've received a mail with the following contents

Subject	[OASE]Unknown event notification
Main body	[Decision table name] [Event start time] [Request reception time] [Event Information] [Trace ID]



**Mail
notification**

POINT

Used in the following chapters:
[<4.3 Judge rules \(Send request via curl command\)>](#)
and
[<4.4 Check the Action results>](#)



Exastro 