



# IT Automation Quickstart

※"IT Automation" will be written as "ITA" in this document

# Table of contents

## 1. Introduction

- 1.1 [Introduction](#)
- 1.2 [Scenario overview](#)
- 1.3 [Terminology](#)

## 2. Screen Description

- 2.1 [Web console screen description \(Login\)](#)
- 2.2 [Screen Description \(Main Menu\)](#)
- 2.3 [Screen Description \(Menu\)](#)

## 3. Preparation

- 3.1 [Uploading Playbook and Linking it to a job \(Movement\)](#)
- 3.2 [Implementing job \(Movement\) into Jobflow \(Conductor\)](#)
- 3.3 [Configure CMDB Data sheet and Parameter sheet](#)
- 3.4 [Register Data to Datasheet](#)
- 3.5 [Link Parameter sheet item to Playbook variable](#)

## 4. Execution(First time)

- 4.1 [Register target \(Linux machine\) to Device list](#)
- 4.2 [Register Operation name \(Operation\)](#)
- 4.3 [Register data to Parameter sheet](#)
- 4.4 [Execute Jobflow \(Conductor\)](#)
- 4.5 [Check Execution results](#)

# Table of Contents

## 5. Execution (Second time)

- 5.1 [Register Operation name \(Operation\)](#)
- 5.2 [Register data to Parameter sheet](#)
- 5.3 [Execute Jobflow \(Conductor\)](#)
- 5.4 [Check Execution results](#)

## 6. Checking the CMDB Parameter history

- 6.1 [Executing operations and History Management](#)
- 6.2 [Checking the CMDB Parameter history](#)

## A Appendix

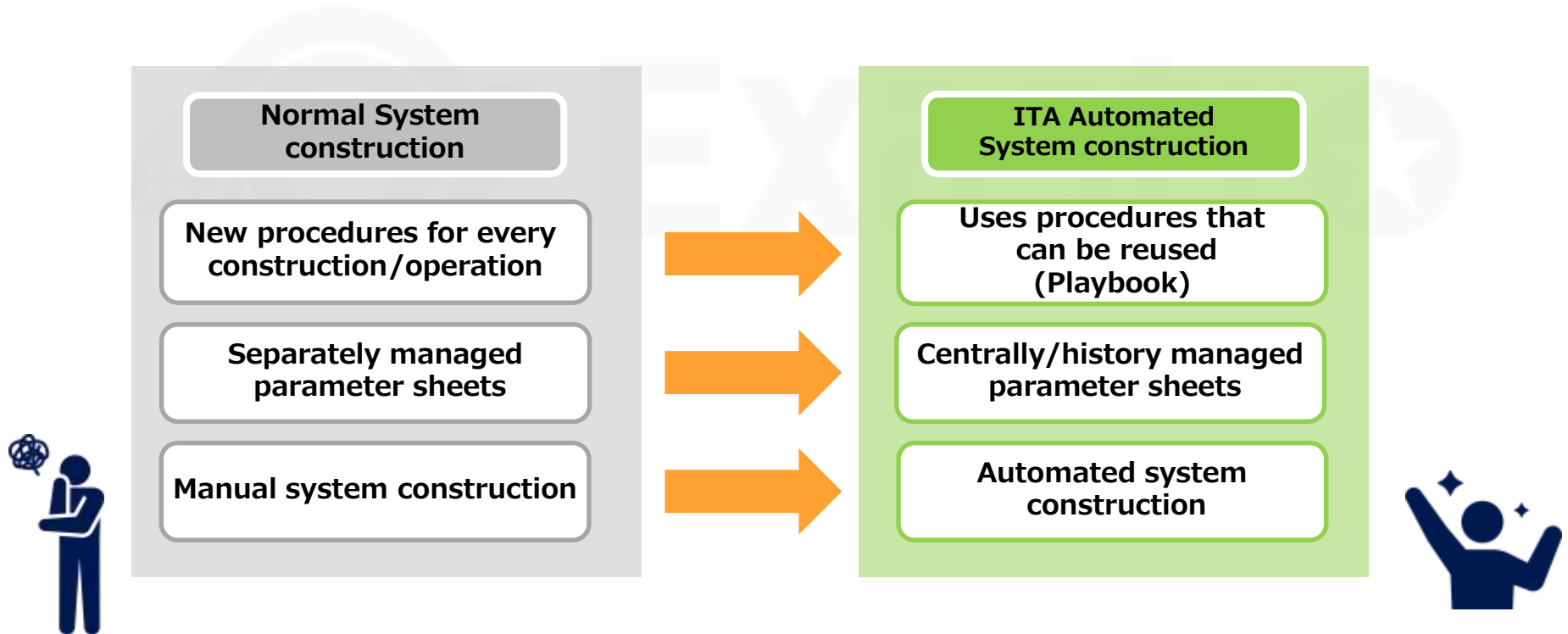
- Reference① [【Ansible-Legacy】 Single Execution](#)
- Reference② [【Ansible-Legacy】 Execution check](#)

# 1. Introduction

# 1.1 Introduction (1/2)

This document serves as a quick start guide for users who are using IT Automation (written as ITA) for the first time.

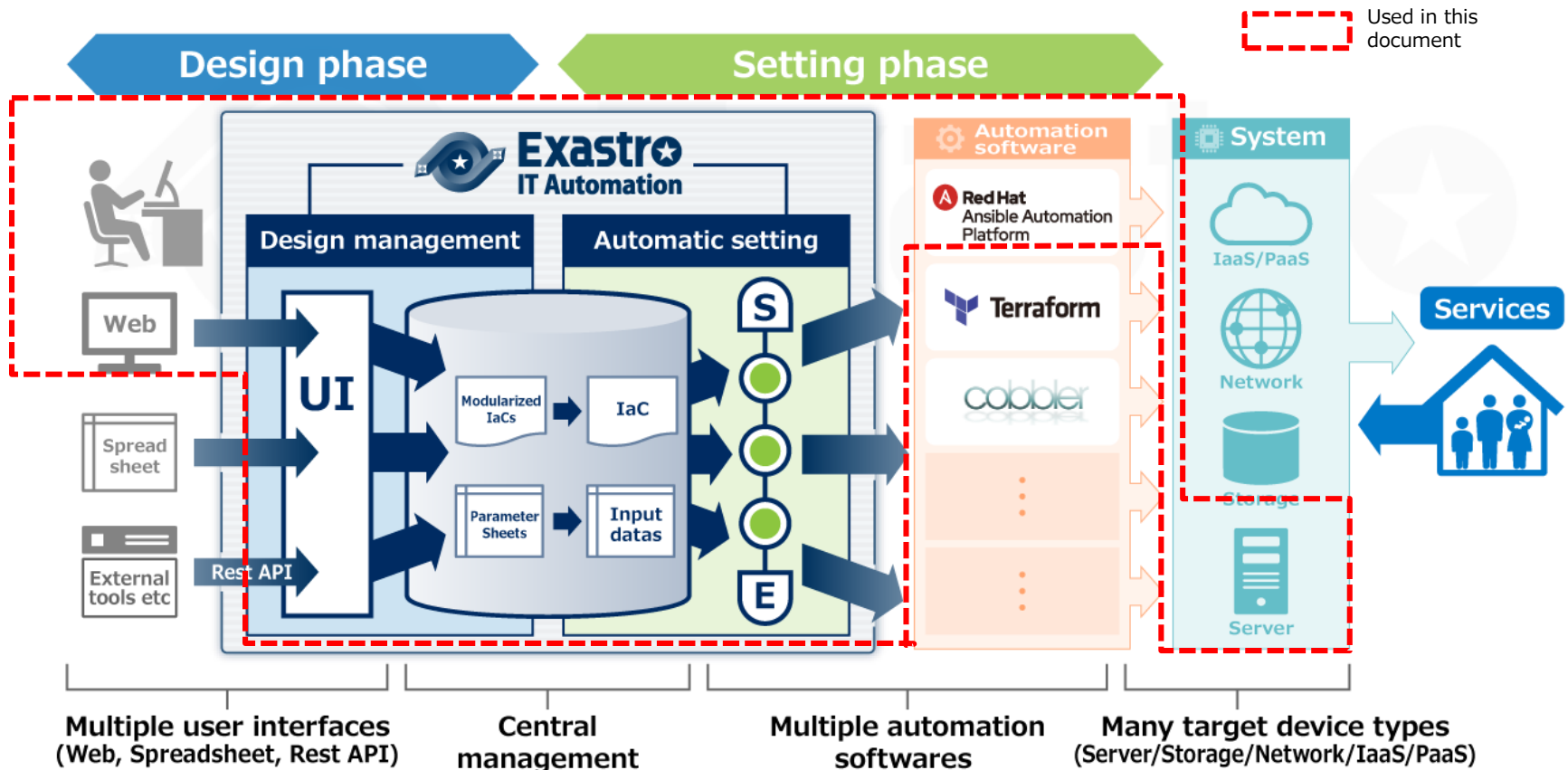
By installing Linux server packages, we can automate and centralize and automate package management for each server. That way, we can use ITA and have a more efficient system than we could achieve from a conventional system.



# 1.1 Introduction (2/2)

## Main ITA functions used in this document.

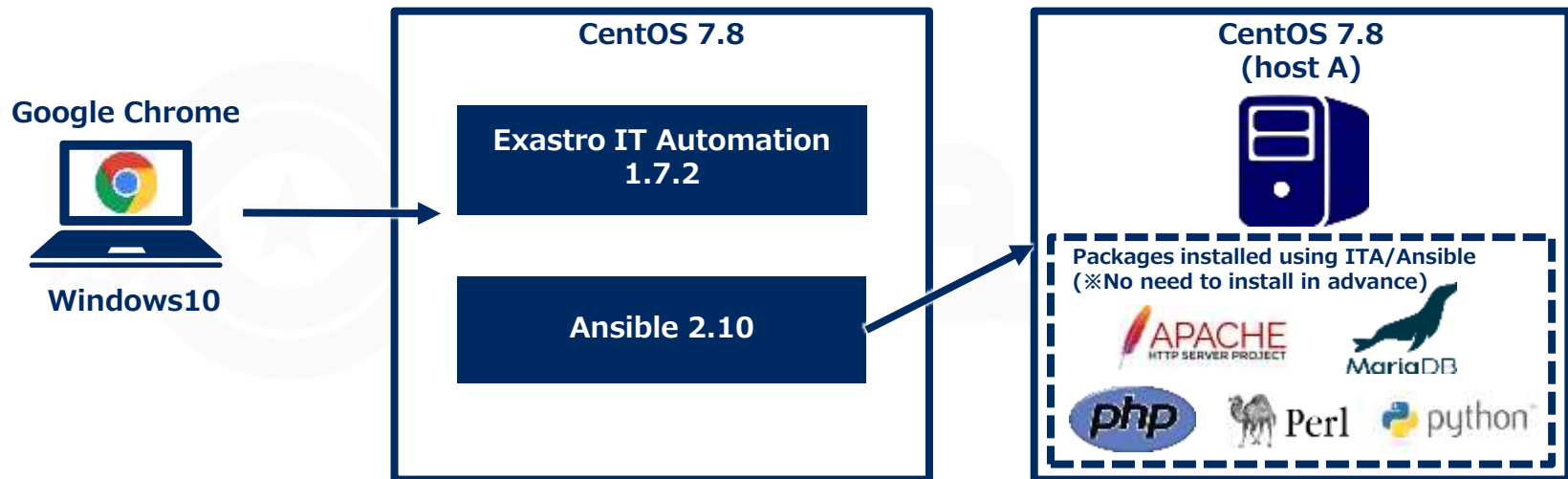
- Linking with Automation software (Ansible).
- Parameter management (Creation, Registration and history management of Menus)
- Linking Variables (Automatic registration of substitute values)



# 1.2 Scenario overview(1/3)

In this scenario, we will use Ansible Driver to manage the parameters for each server and automate the Yum Package installations, which is often used when constructing Linux servers.

## Environment

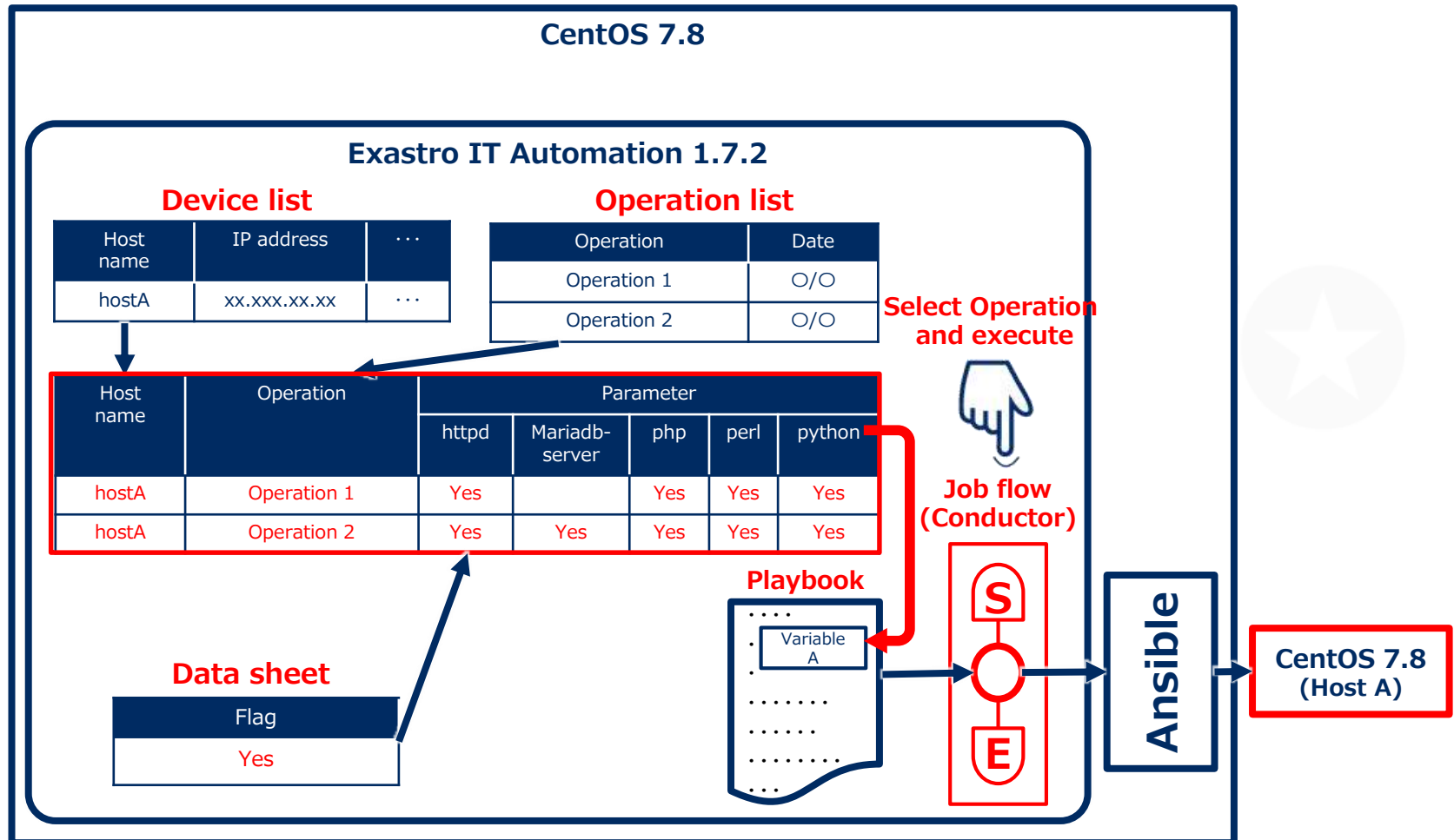


### Systems used

- Exastro IT Automation 1.7.2
- CentOS Linux 7.8(for ITA Server)
- CentOS Linux 7.8(for Target machine)
- Windows 10(Client)
- Google Chrome (Win10 side)

# 1.2 Scenario overview (2/3)

## Scenario execution image

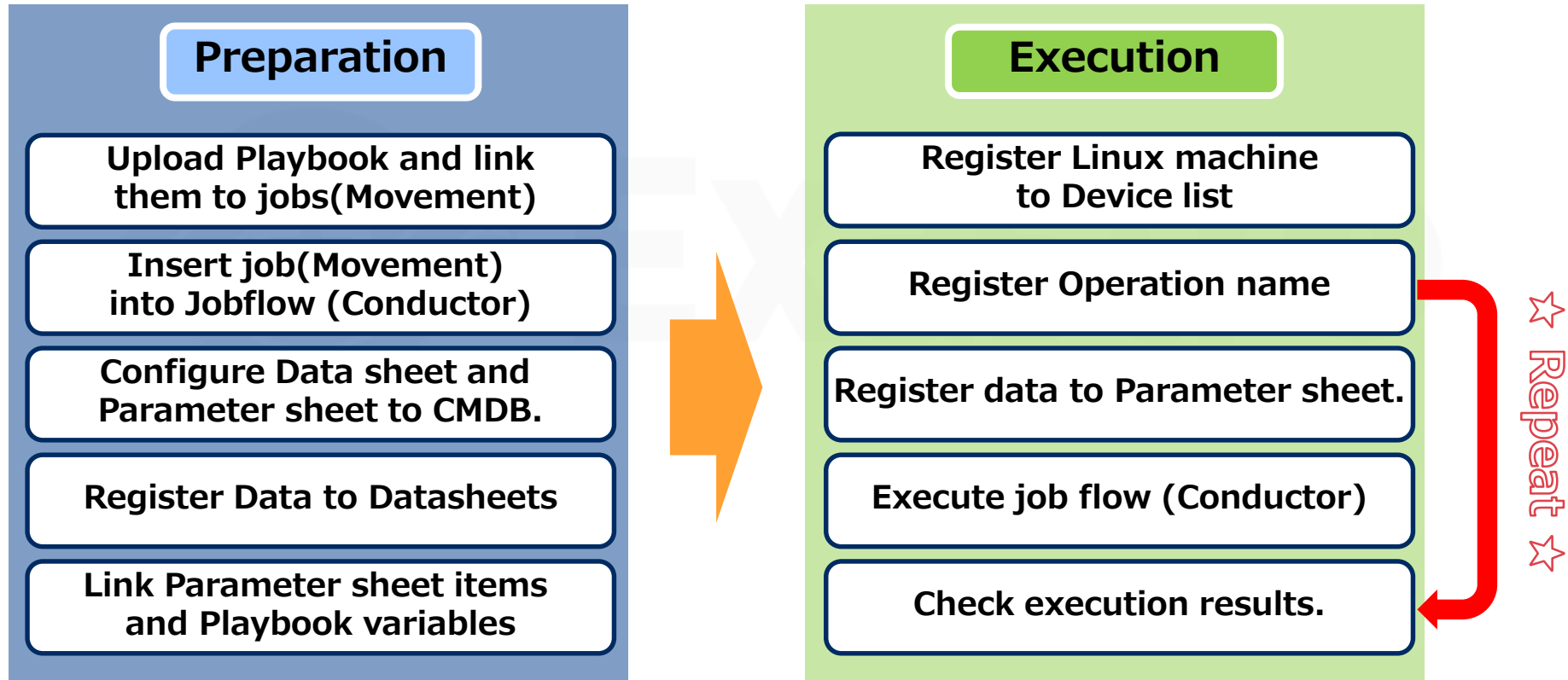




## 1.2 Scenario overview(3/3)

### Post-installation Ansible Legacy execution scenario.

- The figure below illustrates the scenario as well as the Developer(Preparation)/Operator(Execution) operations.



# 1.3 Terminology

The following table explains the different terminology used in this document

| Word                         | Description   |
|------------------------------|---|
| Playbook                     | A file that describes routine tasks that can be executed with Ansible.<br>All Playbook are written in YAML format.  |
| Ansible-Legacy               | A function that allows users to use Ansible from ITA. In the Legacy console, this is used when YAML files are used for the building code.   |
| Operation<br>name(Operation) | Operation unit in ITA. Users can set their execution dates in advance, manage the execution history and more.   |
| Conductor                    | A sequence of work units. It can be executed after an operation name has been linked to it.<br>Combine several parts called Nodes to create a workflow.<br>It can then be used to execute configuration/construction operations on multiple machines. |
| Movement                     | Configuration/Construction units used with each of the devices construction tools.  |

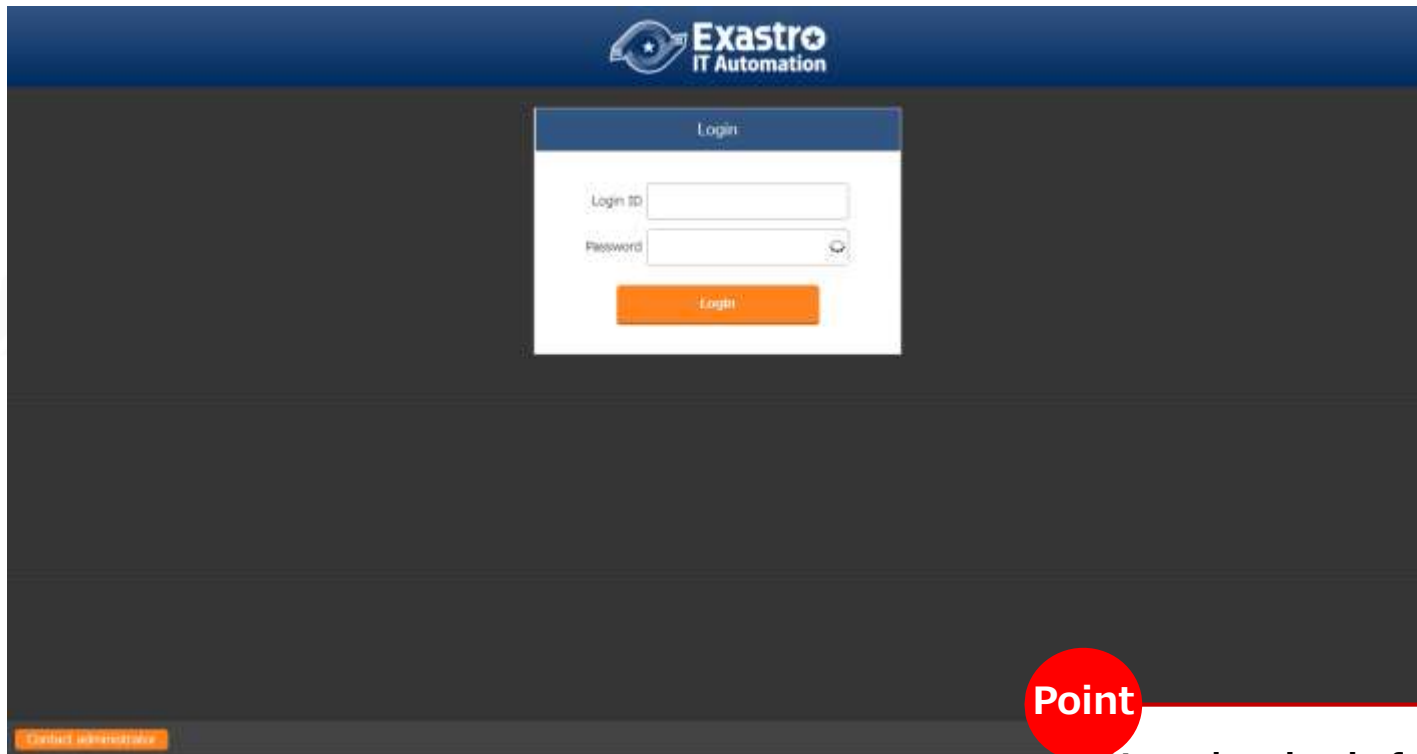
For more information regarding the terminology, please refer to the first step guide.  
If you want more information regarding Exastro ITA, please refer to the Document page on the community website.

## 2. Screen Description

## 2.1 Web console screen description (Login)

### Web Console login screen

- Accessing ITA via URL after it has been installed will direct the user to the login screen
- ※For information regarding how to install ITA, please refer to the Online Install manual



#### Point

Users logging in for the first time will be asked to change their password

## 2.2 Screen Description (Main Menu)

### Screen description (Main Menu)

- The main functions of the Main Menu screen is as following

Exastro Management Console

User name [System Administrator]  
Login ID [administrator]  
Change password Logout

Menu

DASHBOARD

Main menu

System settings

Menu group list

Menu list

Role list

User list

Role - Menu link list

Role - User link list

Sequence list

Single sign-on Basic Preference

Single sign-on Attribute Preference

version

Contact administrator

Menu group

Management Basic Console Export/Import Symphony Conductor Create Menu

File control m File control ch Input Substitution w Reference Contrast

HostGroup ma Ansible-Legacy Ansible-Pioneer Ansible-Legac Ansible Comm Cobbler

Terraform

Movement

Work status

Work result

Work history

Point

Menu bar

Menu groups

For more detailed information regarding the different functions, please refer to the manual.

## 2.3 Screen Description (Menu) (1/2)

### Screen Description (Menus)

- The name of the basic functions are as following.

The screenshot displays the Exastro Ansible-Legacy web interface. On the left is a sidebar menu with options like 'Main menu', 'Movement list', 'Playbook files', etc. The main content area is titled 'Description' and includes a 'Display filter' section with buttons for 'Filter' and 'Clear filter'. Below this is a 'List/Update' section containing a table of movement records.

**Submenu outline**

- Explanation** : Contains a brief description regarding the menu.
- Display Filter** : Lets the user search for registered information
- List/Update** : Displays registered information

**Point**

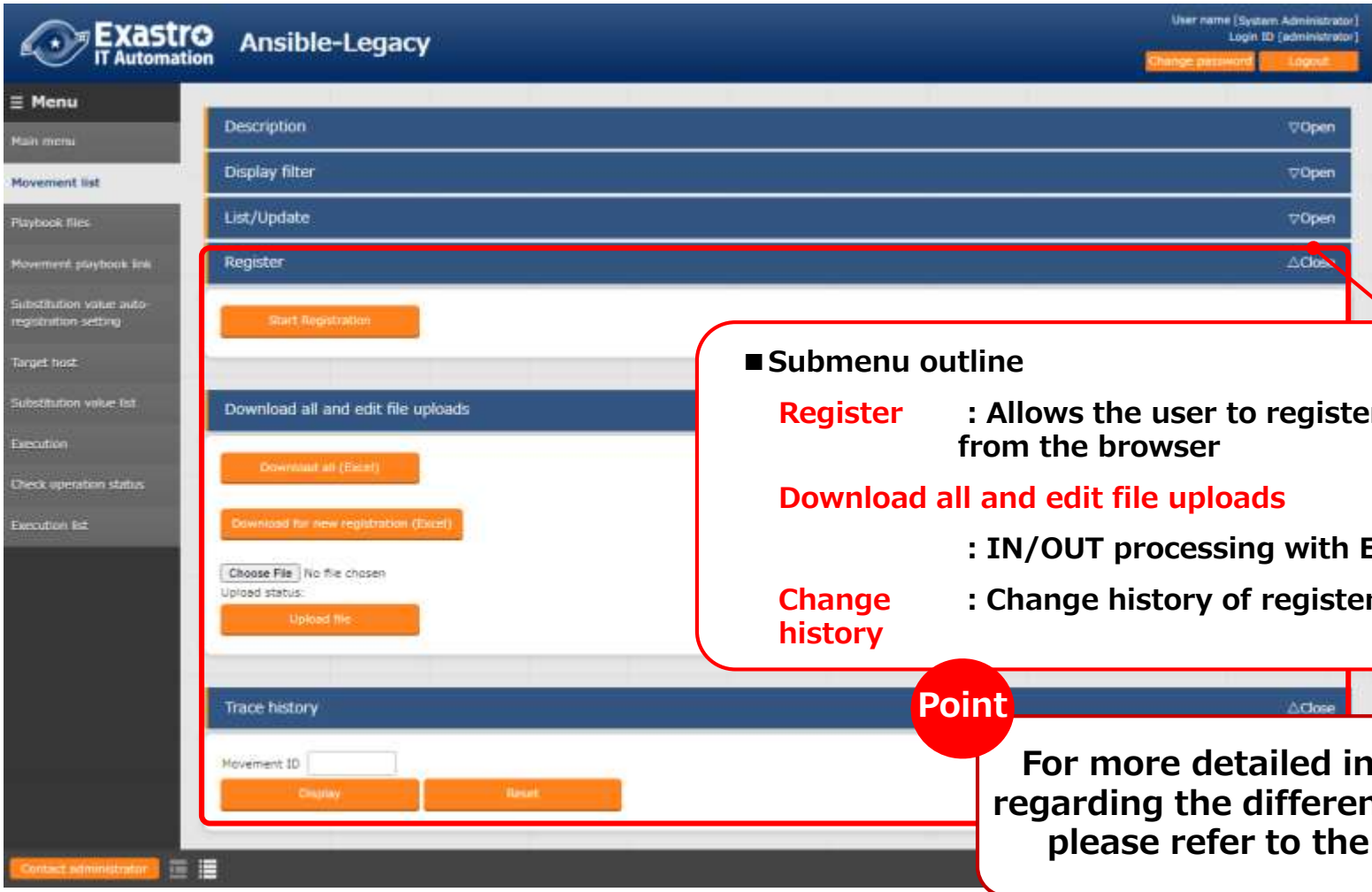
For more detailed information regarding the different functions, please refer to the manual.

| History | Update | Discard | Movement ID | Movement Name      | Orchestrator   | Delay timer | Host specific format | MINIMUM con | Last update date/time | Last updated by      |
|---------|--------|---------|-------------|--------------------|----------------|-------------|----------------------|-------------|-----------------------|----------------------|
| History | Update | Discard | 5           | copy_file          | Ansible Legacy |             | IP                   |             | 2021/06/17 13:07:32   | System Administrator |
| History | Update | Discard | 6           | create_directory   | Ansible Legacy |             | IP                   |             | 2021/06/17 13:07:43   | System Administrator |
| History | Update | Discard | 7           | create_file        | Ansible Legacy |             | IP                   |             | 2021/06/17 13:07:52   | System Administrator |
| History | Update | Discard | 8           | melt_file          | Ansible Legacy |             | IP                   |             | 2021/06/17 13:08:13   | System Administrator |
| History | Update | Discard | 9           | forced_termination | Ansible Legacy |             | IP                   |             |                       |                      |
| History | Update | Discard | 10          | remove_directory   | Ansible Legacy |             | IP                   |             |                       |                      |
| History | Update | Discard | 11          | remove_file        | Ansible Legacy |             | IP                   |             |                       |                      |
| History | Update | Discard | 12          | Set Timezone       | Ansible Legacy |             | IP                   |             |                       |                      |

## 2.3 Screen Description (Menu) (2/2)

### Screen description (Menu)

- The name of the basic functions are as following



The screenshot displays the Exastro Ansible-Legacy web interface. On the left is a sidebar menu with categories like 'Main menu', 'Movement list', 'Playbook files', and 'Execution'. The 'Register' submenu is highlighted in the main content area. This submenu contains several options: 'Start Registration', 'Download all and edit file uploads', 'Download for new registration (Excel)', 'Choose File' (with a file selection interface), 'Upload file', and 'Trace history'. A red box highlights the 'Register' submenu and its contents. A red arrow points from the 'Submenu' label to the 'Register' submenu. A red circle with the word 'Point' inside points to the 'Trace history' option. A red box on the right contains a 'Submenu outline' with descriptions for 'Register', 'Download all and edit file uploads', and 'Change history'.

**Submenu**

**Submenu outline**

- Register** : Allows the user to register records from the browser
- Download all and edit file uploads** : IN/OUT processing with Excel
- Change history** : Change history of registered records

**Point**

For more detailed information regarding the different functions, please refer to the manual.

### 3. Preparation



# 3.1 Uploading Playbook and linking it to a job(Movement) (1/5)

## Playbook preparation

- First, we need to create the Playbook files that we are going to use.

Use your desired editor program to create the following YAML file and save it to your local hard drive.

yum\_package\_install.yml

```
- name: install the latest version of packages
  yum:
    name: "{{ item }}"
    state: latest
  with_items:
    - "{{ VAR_packages }}"
```

### Point

Make sure that the character code is "UTF-8" and the newline code is "LF".  
The file should be saved as an yml file.  
Please be check that the indents are correct.

```
yum_package_install.yml X

1  | name: install the latest version of packages
2  | yum:
3  |   name: "{{ item }}"
4  |   state: latest
5  | with_items:
6  |   - "{{ VAR_packages }}"
7
```

Uploading Playbook  
and linking it to a job(Movement)

Implementing job (Movement)  
into Jobflow (Conductor)

Configure CMDB Data sheet  
and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item  
to Playbook variable.

Register target (Linux machine)  
to Device list.

## 3.1 Uploading Playbook and linking it to a job(Movement) (2/5)

Register Movement to the Movement list.

- In the next step, we will register a Movement.

From the main menu , go to the Ansible-LegacyRole menu and then to the Movement list menu.

The screenshot displays the Exastro IT Automation Ansible-LegacyRole dashboard. The main menu on the left includes options like 'Movement list', 'Basic package list', 'Movement role link', 'Nested variable list', 'Substitution value auto-regulation setting', 'Target host', 'Substitution value list', 'Execution', 'Check operation status', and 'Execution list'. The central dashboard features a grid of icons for various tools and services, including 'Management', 'Basic Console', 'Export/Import', 'Symphony', 'Conductor', 'Create Menu', 'File control', 'Input', 'Substitution', 'Reference', 'Contract', 'HostGroup', 'Ansible-Leg', 'Ansible-Push', 'Ansible-Cli', 'Ansible-Co', and 'Cobbler'. A red box labeled '2' highlights the 'Movement list' option in the main menu. A red box labeled '1' highlights the 'Ansible-Leg' icon in the central dashboard grid. The right side of the dashboard shows three summary cards: 'Movement' (13 total), 'Work status' (0 total), and 'Work result' (1 total). Below these cards is a 'Work history' table.

Uploading Playbook  
and linking it to a job(Movement)

Implementing job (Movement)  
into Jobflow (Conductor)

Configure CMDB Data sheet  
and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item  
to Playbook variable.

Register target (Linux machine)  
to Device list.

## 3.1 Uploading Playbook and linking it to a job(Movement) (3/5)

Register Movement to the Movement list.

- Click the “Start Registration” button.

Follow the table listed below and fill out the different items before pressing the “Register” button.

The screenshot shows the Exastro Ansible-LegacyRole interface. The 'Register' button is highlighted with a red box and a red circle with the number 3. The interface includes a menu on the left, a description field, a display filter, a table with columns for Movement ID, Movement Name, Delay Time, Host Name, Format, and Host Connection, and a 'Start Registration' button.

Uploading Playbook  
and linking it to a job(Movement)

Implementing job (Movement)  
into Jobflow (Conductor)

Configure CMDB Data sheet  
and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item  
to Playbook variable.

Register target (Linux machine)  
to Device list.

The screenshot shows the 'Register' form in the Exastro interface. A red box highlights the 'Movement name' and 'Host format' fields, with a red circle and the number 4 pointing to it. The 'Register' button is highlighted with a red box and the number 5. The form includes a table with columns for Movement ID, Movement Name, Delay Time, Host Name, Format, and Host Connection.

Movement name

Host format

PackageInstall

IP

## 3.1 Uploading Playbook and linking it to a job(Movement) (4/5)

### Register Playbook to the Playbook file menu.

- Next, we will register the Playbook we created earlier to the Playbook files menu.

From the main menu , go to the Ansible-Legacy menu and then to the Playbook files menu. Fill out the items marked with red using the information from table listed below and press the “Register” button.

The screenshot shows the Exastro Ansible-Legacy interface. The left sidebar has a 'Menu' section with 'Playbook files' highlighted. The main area shows a table of Playbook files with columns: Playbook id, Playbook name, Playbook files, Access permission, Last update date/time, and Last update. A red box labeled '1' points to the 'Playbook files' menu item. Another red box labeled '2' points to a table with the following content:

| Playbook file name  | Playbook file           |
|---------------------|-------------------------|
| yum_package_install | yum_package_install.yml |

Below this table is a 'Register' form. A red box labeled '3' points to the 'Register' button. A red box labeled 'Point' points to the 'Upload in advance' button in the 'Playbook files' column of the table.

**Uploading Playbook and linking it to a job(Movement)**

**Implementing job (Movement) into Jobflow (Conductor)**

**Configure Cmdb Data sheet and Parameter sheet**

**Register Data to Datasheet**

**Link Parameter sheet item to Playbook variable.**

**Register target (Linux machine) to Device list.**

**Point**

If you are uploading a Playbook, make sure to hit the "Upload in advance" button before pressing the "Register" button.

# 3.1 Uploading Playbook and linking it to a job(Movement) (5/5)

## Register “Movement-Playbook link”

- Next, we will link the playbook to the earlier registered Movement

From the main menu, go to the Ansible-Legacy menu and then to the “Movement-Playbook link” menu. Fill out the items marked with red using the information from table listed below and press the “Register” button.

**Uploading Playbook and linking it to a job(Movement)**

**Implementing job (Movement) into Jobflow (Conductor)**

**Configure CMDB Data sheet and Parameter sheet**

**Register Data to Datasheet**

**Link Parameter sheet item to Playbook variable.**

**Register target (Linux machine) to Device list.**

| Movement        | Playbook file       | Include order |
|-----------------|---------------------|---------------|
| Install Package | yum_package_install | 1             |

**Point**

If you want to registered a single movement to multiple Playbooks. For 1:1, please input 1.

## 3.2 Implementing job (Movement)into Jobflow (Conductor)

### Create “Conductor”

- In the next step, we will implement the Movement into a conductor.  
From the Conductor menu group, access the Conductor Class edit screen.  
Follow the numbered steps below and press the “Register” button.

**Name**  
InstallPackage

Input field for Remarks and such.

Drag and Drop

Drag a line between “OUT” and “IN”

Register

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

## 3.3 Configure CMDB Data sheet and Parameter sheet (1/4)

### Create Data sheet

- Next, we will create a Data sheet

From the "Menu create" menu group, access the "Create • Define menu" menu.

Fill out the items marked with red using the information from table listed below and press the ""Create"" button.

1. Create - Define menu

2. Package Flag

3. Item name, Input method, Maximum byte size, Required, Unique constraint

| Item name    | Input method | Maximum byte size | Required | Unique constraint |
|--------------|--------------|-------------------|----------|-------------------|
| Package flag | String       | 32                | ✓        | ✓                 |

4. Create

| Menu name       | Creation Target | Display order |
|-----------------|-----------------|---------------|
| Flag Definition | Data sheet      | 2             |

- Uploading Playbook and linking it to a job(Movement)
- Implementing job (Movement) into Jobflow (Conductor)
- Configure CMDB Data sheet and Parameter sheet
- Register Data to Datasheet
- Link Parameter sheet item to Playbook variable.
- Register target (Linux machine) to Device list.



## 3.3 Configure CMDB Data sheet and Parameter sheet (2/4)

### Create Parameter sheet

- In the next step, we will create a parameter sheet.

In the "Create menu" menu group, go to the "Define/Create Menu" menu.

Follow the steps below and fill out the items with the values written in the tables.

Click "Group"

Group name

Install Package

Menu name

Creation target

Display Order

| Menu name            | Creation target                   | Display Order |
|----------------------|-----------------------------------|---------------|
| Install Package list | Parameter sheet (Host/ Operation) | 1             |

Uploading Playbook and linking it to a job (Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

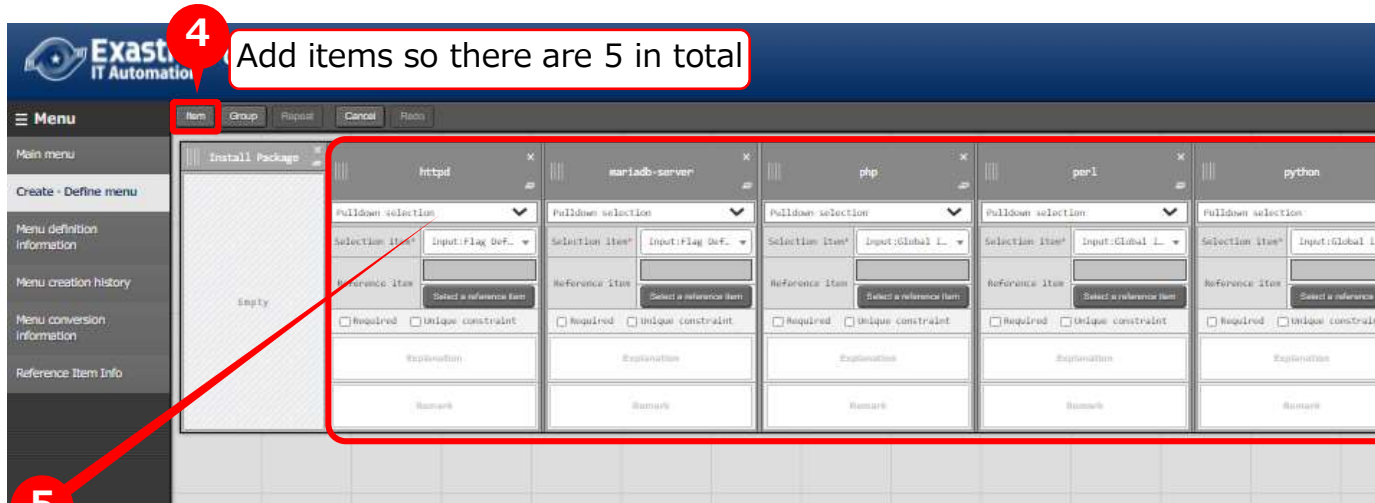


## 3.3 Configure CMDB Data sheet and Parameter sheet (3/4)

### Create Parameter sheet

- Add items and fill the items with the values written in the table below.

**4** Add items so there are 5 in total



**5**

| Item name      | Input method       | Selection item                                |
|----------------|--------------------|---|
| httpd          | Pulldown Selection | Input:Flag Definition: Parameter/Package flag |
| mariadb-server | Pulldown Selection | Input:Flag Definition: Parameter/Package flag |
| php            | Pulldown Selection | Input:Flag Definition: Parameter/Package flag |
| perl           | Pulldown Selection | Input:Flag Definition: Parameter/Package flag |
| python         | Pulldown Selection | Input:Flag Definition: Parameter/Package flag |

**Uploading Playbook and linking it to a job(Movement)**

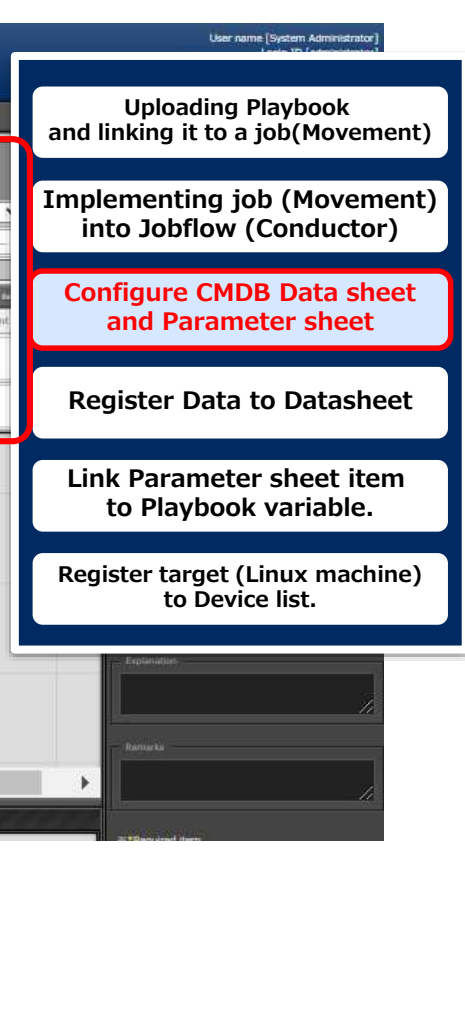
**Implementing job (Movement) into Jobflow (Conductor)**

**Configure CMDB Data sheet and Parameter sheet**

**Register Data to Datasheet**

**Link Parameter sheet item to Playbook variable.**

**Register target (Linux machine) to Device list.**



### 3.3 Configure CMDB Data sheet and Parameter sheet (4/4)

# Create Parameter sheet

- After following the steps below, click the “Create” button.

**6** Drag and drop the items into the column group

When finished, the items should look like this

**7**

**8** List(Preview)

- Uploading Playbook and linking it to a job(Movement)
- Implementing job (Movement) into Jobflow (Conductor)
- Configure CMDB Data sheet and Parameter sheet
- Register Data to Datasheet
- Link Parameter sheet item to Playbook variable.
- Register target (Linux machine) to Device list.

## 3.4 Register Data to Datasheet

### Register data to "Flag definition"

We are now going to register data in the Flag Definition (Datasheet) that we created earlier. Go to the "Input" menu group and then to the "Flag Definition" Menu. Then in the Register submenu, fill the items with the values below and press "Register".

Exastro IT Automation Input

User name [System Administrator]  
Login ID [administrator]  
Change password Logout

Menu

Main menu

Install Package list

Flag Definition

1

2

Package Flag

Yes

3

Register

Uploading Playbook  
and linking it to a job(Movement)

Implementing job (Movement)  
into Jobflow (Conductor)

Configure CMDB Data sheet  
and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item  
to Playbook variable.

Register target (Linux machine)  
to Device list.

## 3.5 Link Parameter sheet item to Playbook variable (1/3)

Create "Substitution value auto-registration settings".

- Lastly, we will automatically register substitute values.

In the "Ansible-Legacy" Menu group, go to the "Substitution value auto-registration setting" menu.

Follow the steps below and fill the items with the values written in the table below.

Exastro IT Automation Ansible-Legacy

User name [System Administrator] Login ID [administrator] Change password Logout

Menu

Main menu

Movement list

Task list

Movement playbook list

Substitution value auto-registration setting

Target host

Substitution value list

Parameter sheet (row)

| Item No.   | Menu group:Menu                                       | Item                            | Last update date/time | Last updated by |
|------------|---|---------------------------------|-----------------------|-----------------|
| Auto-Input | 2100011611:Substitution value:SB:Install Package list | Parameter/Install Package/httpd | Auto-Input            | Auto-Input      |

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

| Menu group:Menu                                      | Item                                     | Registration method | Movement         | Key Variable Variable name | Substitution order |
|--|--|---------------------|------------------|----------------------------|--------------------|
| 2100011611:Substitution value:3:Install Package list | Parameter/Install Package/httpd          | Key type            | 1:PackageInstall | 1:VAR_packages             | 1                  |
| 2100011611:Substitution value:3:Install Package list | Parameter/Install Package/mariadb-server | Key type            | 1:PackageInstall | 1:VAR_packages             | 2                  |
| 2100011611:Substitution value:3:Install Package list | Parameter/Install Package/php            | Key type            | 1:PackageInstall | 1:VAR_packages             | 3                  |
| 2100011611:Substitution value:3:Install Package list | Parameter/Install Package/perl           | Key type            | 1:PackageInstall | 1:VAR_packages             | 4                  |
| 2100011611:Substitution value:3:Install Package list | Parameter/Install Package/python         | Key type            | 1:PackageInstall | 1:VAR_packages             | 5                  |

## 3.5 Link Parameter sheet item to Playbook variable (2/3)

Create "Substitution value auto-registration settings".

- Follow the table below and press the "Register" button.

The screenshot shows the 'Substitution value auto-registration setting' form in the Exastro Ansible-Legacy interface. The form has a sidebar menu on the left with options like 'Main menu', 'Movement list', 'Playbook files', 'Movement playbook list', 'Substitution value auto-registration setting', 'Target host', 'Substitution value list', 'Execution', 'Check operation status', and 'Execution list'. The main form area has a 'Description' field, a 'Display filter' field, a 'List/Update' button, and a 'Register' button. Below these is a table titled 'ITEM variable(s)' with columns: 'Item No.', 'registration method', 'Movement', 'variable name', 'key', 'value', 'last update date/time', and 'last updated by'. The table contains one row with 'auto-input' in the 'Item No.' column, 'key type' in the 'registration method' column, 'IT.PackageInstall' in the 'Movement' column, 'E-VAR\_packages' in the 'variable name' column, and 'auto-input' in the 'key' and 'value' columns. The 'Register' button is highlighted with a red box and a red circle with the number 2.

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

Point

The following table describes the 3 different variable link registration methods.

| Registration method | Use | Description  |
|---------------------|-----|--|
| Value type          |     | Basic registration type. Links the value written in the table to the variable.                                   |
| Key type            | ●   | Links the table item (column name) to the variable. If the item's setting value is blank, it will not be linked. |
| Key-Value type      |     | Links both the item name (Key) and the setting value (Value) to the variable.                                    |

In this scenario, we want to assign the table items (column name) to the Playbook as a specific value, so we will choose the "Key" registration method.

For more information, please see the [Exastro System Operation and Construction Efficiency Guide](#)

## 3.5 Link Parameter sheet item to Playbook variable (3/3)

Create "Substitution value auto-registration settings".

Use the Display filter to check that you have registered 5 items.

This will end the preparation operations.

Exastro IT Automation Ansible-Legacy

User name [System Administrator]  
Login ID [administrator]  
Change password Login

Menu

- Main menu
- Movement list
- Playbook file
- Movement playbook link
- Substitution value auto-registration setting
- Target host
- Substitution value list
- Execution
- Check operation status
- Execution list

3 Display filter

4 Filter Clear filter

5 Check that 5 items are registered

Lib/Update

| History | Update | Discard | Item name                    | Menu group              | Parameter sheet   | Last update date/time | Last updated by      |
|---------|--------|---------|------------------------------|-------------------------|-------------------|-----------------------|----------------------|
| History | Update | Discard | 100001001 Substitution value | 10 Install Package list | Parameter/Install | 2021/07/01 12:10:13   | System Administrator |
| History | Update | Discard | 200001001 Substitution value | 20 Install Package list | Parameter/Install | 2021/07/01 12:10:26   | System Administrator |
| History | Update | Discard | 300001001 Substitution value | 30 Install Package list | Parameter/Install | 2021/07/01 12:10:55   | System Administrator |
| History | Update | Discard | 400001001 Substitution value | 40 Install Package list | Parameter/Install | 2021/07/01 12:11:22   | System Administrator |
| History | Update | Discard | 500001001 Substitution value | 50 Install Package list | Parameter/Install | 2021/07/01 12:11:52   | System Administrator |

Filter result count: 5

Output Page

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

## 4. Execution (First time)



## 4.1 Register target (Linux machine) to Device list (1/3)

Register target host to "Device list".

- First, we will have to register the target host to which we will install packages to. From the "Basic Console" Menu group, go to the "Device list" menu. Fill in the information written in the table below.

The screenshot shows the Exastro IT Automation Basic Console interface. The 'Menu' on the left has 'Device list' highlighted. The main area shows a table for registering a target host. The table has columns for 'HW Device type', 'Host name', 'IP Address', 'MAC Address', and 'Network device name'. A red box highlights the table, and a red circle with the number '2' points to the table. A red circle with the number '1' points to the 'Device list' menu item.

| HW Device type | Host name | IP Address   | MAC Address | Network device name |
|----------------|-----------|--------------|-------------|---------------------|
| 自動入力           | host1     | 192.168.10.1 |             |                     |

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

HW Device type

SV

Host name

(Free host name)

IP Address

(Free IP Address)



## 4.1 Register target (Linux machine) to Device list (2/3)

Register target host to "Device list".

- Use the scrollbar to scroll to the left and fill in the items listed below.

**Exastro IT Automation Basic Console**

User name: System Administrator  
Login ID: administrator  
Change password Logout

**Menu**

- Main menu
- Device list
- Operation list
- Movement list
- FR Diagram
- ER Diagram Menu List
- ER Diagram Item List

**Register**

| Managed system item number | OS device type | host name | IP address | MAC address | Netw | Last update date/time | Last updated by |
|----------------------------|----------------|-----------|------------|-------------|------|-----------------------|-----------------|
| Auto-Input                 | SV             |           |            |             |      | Auto-Input            | Auto-Input      |

**2**

| Login user ID   | Login password management | Login Password   |
|-----------------|---------------------------|------------------|
| (Login user ID) | ●                         | (Login password) |

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

## 4.1 Register target (Linux machine) to Device list (3/3)

Register target host to "Device list".

- Use the table below to fill in the last item and press the "Register" button.

Exastro IT Automation 基本コンソール

ようこそシステム管理者さん  
ログイン中 (Administrator)

パスワード変更 ログイン

Menu

メインメニュー

機能一覧

オペレーション機能

Movement一覧

日時表示

説明

表示フィルタ

一覧/更新

登録

3

Dedicated information for Legacy/Role Authentication method

Password authentication

Ansible利用情報

Legacy/Role利用情報

Windows接続情報

接続更新日時 接続更新者

接続方式

ポート番号

サーバ証明書

自動入力

パスワード認証

ファイルを選択 | 選択されていません

手動アップロード

アップロード状況

自動入力

自動入力

※は必須項目です。

戻る 登録

4

Point

全件ダウンロードとファイルアップロード編集

変更履歴

Uploading Playbook and linking it to a job(Movement)

Implementing job (Movement) into Jobflow (Conductor)

Configure CMDB Data sheet and Parameter sheet

Register Data to Datasheet

Link Parameter sheet item to Playbook variable.

Register target (Linux machine) to Device list.

In order to run Ansible-Legacy, the following 6 items must be filled.  
**"Host name", "IP Address", "Login User ID", "Login Password Management", "Login password", "Authentication method※"**

※In this document, "Authentication method" is written as "Password authentication".

## 4.2 Register Operation name (Operation)

### Register "Operation name" to "Operation list"

In this step, we will register an Operation name. From the "Basic Console" menu group, go to the "Operation list" menu.

Input the following information and press the "Register" button.

The screenshot shows the Exastro Basic Console interface. The left sidebar contains a 'Menu' section with 'Operation list' highlighted. The main area displays the 'Operation list' page with a 'Register' button. A red box labeled '1' points to the 'Operation list' menu item. Another red box labeled '2' points to the 'Register' form fields. A third red box labeled '3' points to the 'Register' button.

**Register Operation name (Operation)**

Register data to Parameter sheet

Execute Jobflow (Conductor)

Check Execution results

| No.        | Operation ID | Operation name | Scheduled date for execution | Access permission | Last update date/time | Last updated by |
|------------|--------------|----------------|------------------------------|-------------------|-----------------------|-----------------|
| Auto-Input | Auto-Input   | Operation 1    | 2021/07/02 14:09             | Auto-Input        | Auto-Input            | Auto-Input      |

| Operation name | Scheduled date and time |
|----------------|-------------------------|
| Operation 1    | (Free date/time)        |

## 4.3 Register data to Parameter sheet (1/2)

### Register data to Install Package list.

In the next step, we are going to input data to the Install package list (Parameter sheet) that we prepared earlier.

Go to the "Input" menu and then the "Install package list" menu.

Input the following information and press the "Register" button.

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

Check Execution results

Exastro IT Automation Input

User name [System Administrator] Login ID [adminmaster] [Change password](#) [Logout](#)

Menu

Main menu

Install Package list

Register

| No | Host name | Operation                       | httpd | mariadb-server | php | perl | python | Last update date/time | Last updated by |
|----|-----------|---------------------------------|-------|----------------|-----|------|--------|-----------------------|-----------------|
| 1  | Host-001  | 2021/07/02 14:05:21:Operation 1 | Yes   |                | Yes | Yes  | Yes    | Auto input            | Auto input      |

Back Register

| Host name                    | Operation                                  | httpd | mariadb-server | php | perl | python |
|------------------------------|--|-------|----------------|-----|------|--------|
| (Previously registered host) | (previously specified date )_1:Operation 1 | Yes   |                | Yes | Yes  | Yes    |

## 4.3 Register data to Parameter sheet (2/2)

### Register data to Install Package list.

- Similarly to when we configured substitution value settings, open the display filter and press the "filter" button to check if the registration was done correctly.

The screenshot shows the 'Exastro IT Automation Input' interface. A red box labeled '1' highlights the 'Display filter' button in the top right of the main content area. A red box labeled '2' highlights the 'Filter' button in the filter dialog. A red box labeled '3' highlights the 'Operation' table in the main content area. The table has columns: history, update, discard, edit, test name, ID, operation name, reference date, scheduled date for execution, last update date/time, and last updated by. The first row shows a green 'Ready' status, a red 'Discard' button, and the text '11TA-test-targeted 22 operation 1 2021/07/02 14:00 2021/07/02 14:00 2021/07/02 14:14:48 system administrator'.

| history | update | discard | edit | test name          | ID | operation name | reference date   | scheduled date for execution | last update date/time | last updated by      |
|---------|--------|---------|------|--------------------|----|----------------|------------------|------------------------------|-----------------------|----------------------|
| Ready   |        | Discard |      | 11TA-test-targeted | 22 | operation 1    | 2021/07/02 14:00 | 2021/07/02 14:00             | 2021/07/02 14:14:48   | system administrator |

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

Check Execution results

## 4.4 Execute Jobflow (Conductor) (1/3)

### Run Conductor

- We will now start the Conductor.

From the "Conductor" Menu group, go to the "Conductor Execution" Menu.

Next, select "Conductor" and "Operation" and press "Execute".

The screenshot displays the Exastro Conductor web interface. The left sidebar contains a 'Menu' section with 'Conductor execution' highlighted (1). The main content area shows a table of conductor classes (2) and a table of operations (3). The 'Execute Jobflow (Conductor)' button is highlighted in a red box (4). A callout box on the right lists the steps: 'Register Operation name (Operation)', 'Register data to Parameter sheet', 'Execute Jobflow (Conductor)', and 'Check Execution results'.

**Conductor Class List (2):**

| Select                   | Conductor class ID | Conductor name   | Explanation | Access permission    | Remarks | Last update date/time | Last updated by      |
|--------------------------|--------------------|------------------|-------------|----------------------|---------|-----------------------|----------------------|
| <input type="checkbox"/> | 1                  | 2/InstallPackage |             | Role to allow access |         | 2021/07/01 11:34:52   | System administrator |

Filter result count: 1

**Operation List (3):**

| Select                   | No. | Operation ID | Operation name | Scheduled date for execution | Last execution date |
|--------------------------|-----|--------------|----------------|------------------------------|---------------------|
| <input type="checkbox"/> | 21  | 21           | 21 Operation 3 | 2021/07/01 14:00             |                     |

Filter result count: 1

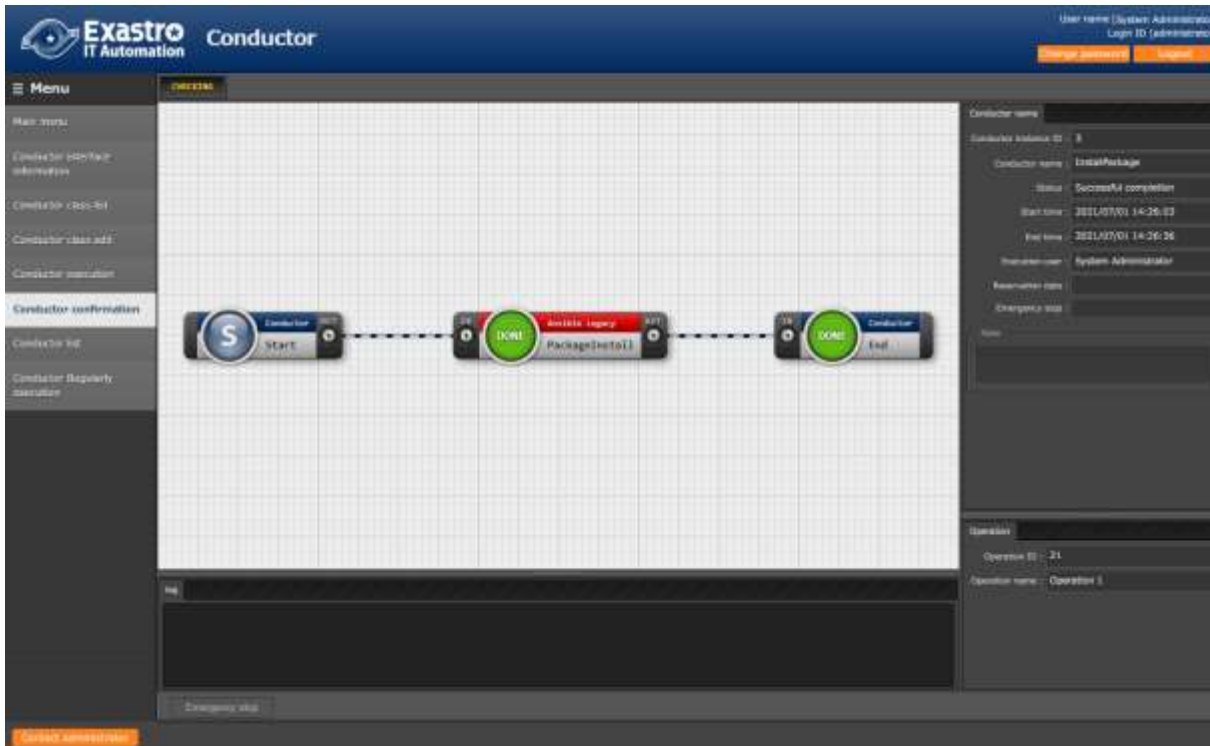
**Execute Jobflow (Conductor) (4):**

The 'Execute' button is highlighted in a red box at the bottom right of the interface.

## 4.4 Execute Jobflow (Conductor) (2/3)

### Execution results

- Executing the Conductor will move the user to the "Conductor confirmation" screen where execution status and execution logs are displayed.



Register Operation name  
(Operation)

Register data to  
Parameter sheet

**Execute Jobflow (Conductor)**

Check Execution results

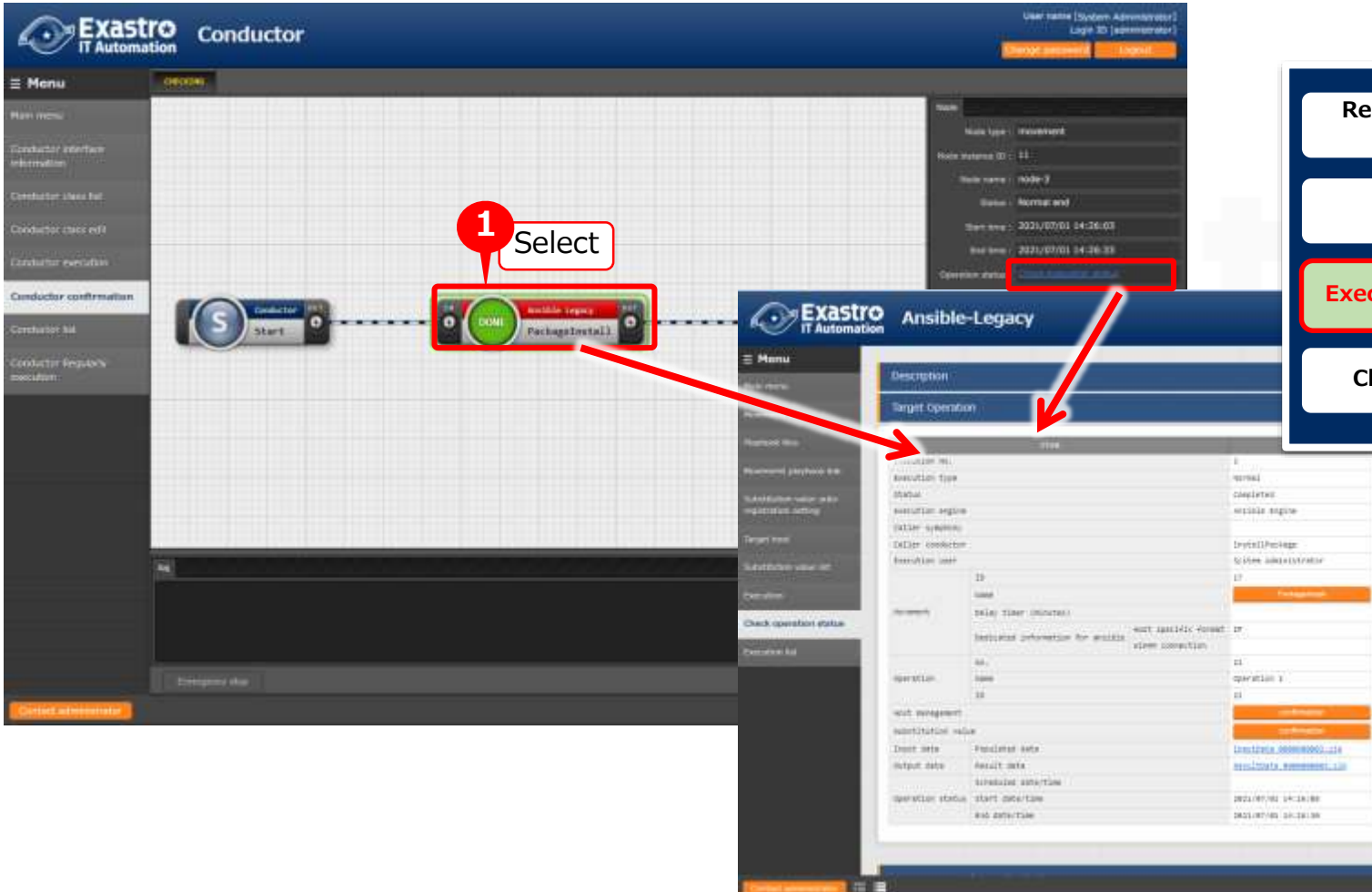
**Point**

The execution status  
and execution log  
can be checked  
in real-time.

## 4.4 Execute Jobflow (Conductor) (3/3)

### Execution results

- Select a job (Movement) and press either the "Done" icon or the Operation status on the right to see more details.



The screenshot displays the Exastro Conductor interface. On the left, a sidebar menu includes options like 'Main menu', 'Conductor interface information', 'Conductor class list', 'Conductor class edit', 'Conductor execution', 'Conductor confirmation', 'Conductor list', and 'Conductor frequently executed'. The main area shows a jobflow diagram with a 'Start' node and a 'PackageInstall' node. A red box highlights the 'PackageInstall' node, with a red circle containing the number '1' and the word 'Select' next to it. A red arrow points from this node to a detailed view of the job execution. The detailed view shows the job's status as 'Completed' and provides a table of execution results.

**Register Operation name (Operation)**

**Register data to Parameter sheet**

**Execute Jobflow (Conductor)**

**Check Execution results**

| operation   | id | name        | status    | start time          | end time            |
|-------------|----|-------------|-----------|---------------------|---------------------|
| operation 1 | 1  | operation 1 | Completed | 2021/07/01 04:18:00 | 2021/07/01 04:18:00 |



## 4.5 Check Execution results (1/3)

### Execution results

- In the detailed results screen, we can use the progress status (Execution log) to check the Ansible execution log.

The screenshot displays the Ansible execution results interface. On the left is a dark sidebar with a 'Menu' icon and several navigation items: 'Main menu', 'Movement list', 'Playbook files', 'Movement playbook link', 'Substitution value auto-registration setting', 'Target host', 'Substitution value list', 'Execution', 'Check operation status', and 'Execution list'. The 'Check operation status' item is highlighted.

The main content area shows a table at the top with execution details:

|                     |                 |  |
|---------------------|-----------------|--|
| Output data         | Result data     | <a href="#">ansible-playbook doc00000001.yml</a> |
| Scheduled date/time |                 |  |
| Operation status    | Start date/time | 2021/07/01 14:26:00                              |
|                     | End date/time   | 2021/07/01 14:26:30                              |

Below the table is a section titled 'Progress status(Execution log)'. It contains a filter input and a checkbox labeled 'Display only corresponding lines'. The log content shows the following steps:

```
verifying : httpd-2.4.6-97.el7.centos.x86_64 1/3
verifying : mailcap-2.1.41-2.el7.noarch 2/3
verifying : httpd-tools-2.4.6-97.el7.centos.x86_64 3/3

Installed:
httpd.x86_64 @1:2.4.6-97.el7.centos

Dependency Installed:
httpd-tools.x86_64 @1:2.4.6-97.el7.centos mailcap.noarch @1:2.1.41-2.el7

complete!
1
META: ran handlers
META: ran handlers

PLAY RECAP *****
its-test-target02 : ok=1 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
```

Below the execution log is a section titled 'Progress status(Error log)' with a 'Close' button in the top right corner. It also has a filter input and a checkbox labeled 'Display only corresponding lines'.

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

Check Execution results

## 4.5 Check Execution results (2/3)

### Execution results

- Use the Execution log to see if httpd, php, perl and python are installed.

Progress log(Execution log)

Installed:

httpd.x86\_64 0:2.4.6-97.el7.centos      php.x86\_64 0:5.4.16-48.el7

Dependency Installed:

httpd-tools.x86\_64 0:2.4.6-97.el7.centos    libzip.x86\_64 0:0.10.1-8.el7  
mailcap.noarch 0:2.1.41-2.el7      php-cli.x86\_64 0:5.4.16-48.el7  
php-common.x86\_64 0:5.4.16-48.el7

Updated:

perl.x86\_64 4:5.16.3-299.el7\_9      python.x86\_64 0:2.7.5-90.el7

Dependency Updated:

perl-libs.x86\_64 4:5.16.3-299.el7\_9    python-libs.x86\_64 0:2.7.5-90.el7

Complete!

```
"  
  ]  
}
```

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

**Check Execution results**

## 4.5 Check Execution results (3/3)

### Check the Target machine.

- Check that the packages are installed on the Target machine.

hostA

```
$ yum list installed httpd
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: ftp-srv2.kddilabs.jp
* extras: ftp-srv2.kddilabs.jp
* updates: ftp-srv2.kddilabs.jp
Installed Packages
httpd.x86_64                2.4.6-97.el7.centos        @updates
```

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

**Check Execution results**

## 5. Execution (Second time)

# 5.1 Register Operation name (Operation)

Register Operation name to the "Operation list".

- This step will be the same as the first time we registered an operation name.

From the "Basic Console" menu group, go to the "Operation list" menu.

Input the information below and press the "Register" button.

The screenshot shows the Exastro Basic Console interface. On the left, the 'Menu' sidebar has 'Operation list' highlighted with a red box and a red circle with the number 1. The main content area shows the 'Operation list' table with columns: No., operation ID, operation name, sch, last update date/time, and last updated by. Below the table is a 'Register' form with a red box and a red circle with the number 2. The form contains a table with columns: No., operation ID, operation name, scheduled date for execution, and Action. The first row has 'Auto-Input', 'Auto-Input', 'Operation 2', and '2021/07/03 14:07'. Below the table is a 'Register' button with a red box and a red circle with the number 3.

**Register Operation name  
(Operation)**

**Register data to  
Parameter sheet**

**Execute Jobflow (Conductor)**

**Check Execution results**

**Operation name**

Operation 2

**Reservation  
date/time**

(Free date/time)

# 5.2 Register data to Parameter sheet

## Register data to “Install Package list”

- From the "Input" menu group, go to the "Install package list" menu
- Input the information below and press the "Register" button. Please note that the packages we are installing are different from the first time.



Register Operation name (Operation)

Register data to Parameter sheet

Execute Jobflow (Conductor)

Check Execution results

| Host name                    | Operation                                  | httpd | mariadb-server | php | perl | python |
|------------------------------|--|-------|----------------|-----|------|--------|
| (Previously registered host) | (Previously specified date )_1:Operation 1 | Yes   | Yes            | Yes | Yes  | Yes    |

## 5.3 Execute Jobflow (Conductor) (1/3)

### Run Conductor

- We will now run the Conductor a second time.

From the “Conductor” Menu group, go to the “Conductor execution” menu.

Select the Conductor and Operation you want to run and press “Execute”.

The screenshot displays the Exastro Conductor web interface. The left sidebar contains a 'Menu' section with 'Conductor execution' highlighted by a red box and a red circle with the number 1. The main content area shows the 'Conductor execution' page. A table of conductors is displayed, with one conductor selected and highlighted by a red box and a red circle with the number 2. Below this, a table of operations is shown, with one operation selected and highlighted by a red box and a red circle with the number 3. At the bottom of the interface, a red box with a red circle containing the number 4 highlights the 'Execute' button. To the right of the interface, a vertical stack of four buttons is shown: 'Register Operation name (Operation)', 'Register data to Parameter sheet', 'Execute Jobflow (Conductor)' (highlighted with a red border), and 'Check Execution results'.

Exastro IT Automation Conductor

User name [System Administrator] Login ID [admin@exastro] Change password Logout

Menu

- Run menu
- Conductor interface information
- Conductor class list
- Conductor class edit
- Conductor execution**
- Conductor confirmation
- Conductor list
- Conductor Regularly execution

Description [Open]

Scheduling [Close]

Specify the scheduled date/time in (YYYY/MM/DD HH:MM) Immediately execute when blank.  
Scheduled date/time: [ ]

Conductor [Filter]

Conductor [List]

| SELECT                              | Conductor class ID | Conductor name | Explanation | Access permission     | Monitor | Last check | date/time           |
|-------------------------------------|--------------------|----------------|-------------|-----------------------|---------|------------|---------------------|
| <input checked="" type="checkbox"/> | 1                  | installimage   |             | Role: 00 allow access |         |            | 2012/07/01 13:14:00 |

Filter result count: 1

Operation [Filter]

Operation [List]

| SELECT                              | Op | Operation ID | Operation name | Scheduled date for execution | Last execution date | Next | Last op    |
|-------------------------------------|----|--------------|----------------|------------------------------|---------------------|------|------------|
| <input checked="" type="checkbox"/> | 01 | 01           | Operation 1    | 2012/07/01 14:00             | 2012/07/01 14:16    |      | 2012/07/01 |
| <input checked="" type="checkbox"/> | 02 | 02           | Operation 2    | 2012/07/01 14:00             |                     |      | 2012/07/01 |

Filter result count: 2

Conductor execution

Execute

Register Operation name (Operation)

Register data to Parameter sheet

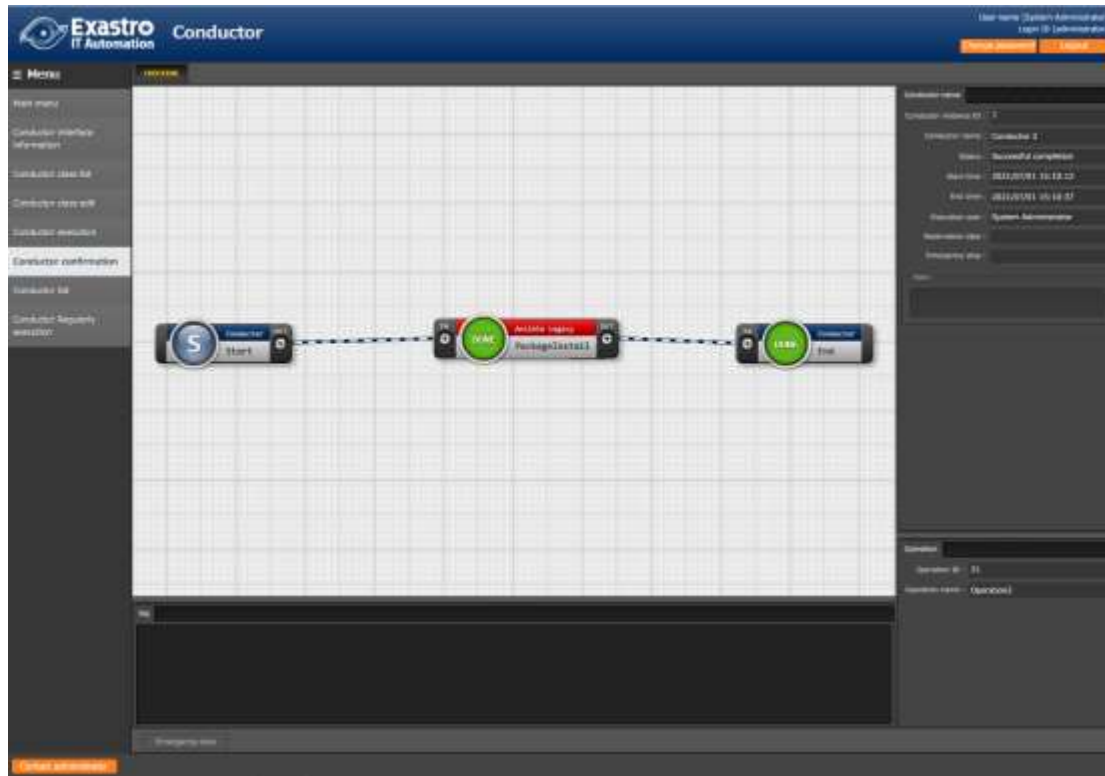
**Execute Jobflow (Conductor)**

Check Execution results

## 5.3 Execute Jobflow (Conductor) (2/3)

### Execution results

- Executing the Conductor will move the user to the "Conductor confirmation" screen where execution status and execution logs are displayed.



Register Operation name  
(Operation)

Register data to  
Parameter sheet

**Execute Jobflow (Conductor)**

Check Execution results

**Point**

The Execution status and the Execution log can be checked in real-time.



## 5.3 Execute Jobflow (Conductor) (3/3)

## Execution results

- Select a job (Movement) and press either the "Done" icon or the Operation status on the right to see more details.

The screenshot displays the Exastro Conductor interface. On the left, a sidebar contains navigation options like 'Menu', 'Main menu', 'Conductor interface information', 'Conductor class list', 'Conductor class edit', 'Conductor execution', 'Conductor configuration', 'Conductor job', 'Conductor regularly execution', and 'Conductor command line'. The main area shows a job flow execution with a 'DONE' status. A red box highlights this status, and a red arrow points to the 'Check operation status' tab. Another red box highlights the 'Register data to Parameter sheet' button, and a red arrow points to the 'Register Operation name (Operation)' button. A third red box highlights the 'Execute Jobflow (Conductor)' button. A fourth red box highlights the 'Check Execution results' button. The interface also shows a 'Description' tab with a table of execution details.

| Item                             | Value                |
|----------------------------------|----------------------|
| operation id                     | 9                    |
| operation type                   | Normal               |
| status                           | Completed            |
| execution engine                 | Ansible Engine       |
| caller synonym                   |                      |
| caller conductor                 | InitialPackage       |
| operation user                   | System Administrator |
| ID                               | 97                   |
| Name                             | PackageInstall       |
| delay timer (seconds)            | 0                    |
| deducted information for ansible | ansible connection   |
| Ansible                          | operations           |
| ID                               | 98                   |
| Host management                  | operations           |
| substitution value               | operations           |
| input data                       | operations           |
| output data                      | operations           |
| operation status                 | operations           |
| start date/time                  | 2021/07/08 15:11:57  |
| end date/time                    | 2021/07/08 15:12:00  |

## 5.4 Check Execution results (1/2)

## Execution results

- In the detailed results screen, we can use the progress status (Execution log) to check the Ansible execution log.

Menu

Main menu

Movement list

Playbook files

Movement playbook list

Substitution value auto registration setting

Target host

Substitution value list

Execution

Check operation status

Execution list

|                    |                                  |                                      |   |
|--------------------|----------------------------------|--------------------------------------|---|
| Reconnect          | Delay timer (minutes)            | Host specific format upon connection | JP                                      |
|                    | Dedicated information for enable |                                      |   |
|                    | No.                              |                                      | 21                                      |
| Operation          | Name                             |                                      | Operation2                              |
|                    | ID                               |                                      | 21                                      |
| Host management    |                                  |                                      | confirmation                            |
| Substitution value |                                  |                                      | confirmation                            |
| Input data         | Populated data                   |                                      | <a href="#">copyData @xxxxxxxxx.rtc</a> |
| Output data        | Result data                      |                                      | <a href="#">resuData @xxxxxxxxx.rtc</a> |
|                    | Scheduled date/time              |                                      |   |
| Operation status   | Start date/time                  |                                      | 2021/07/01 15:11:57                     |
|                    | End date/time                    |                                      | 2021/07/01 15:12:09                     |

Progress status(Execution log)

Filter:

☐ Display only corresponding lines

```

"pro",
"perl",
"python"
},
"msg": "",
"res": 0,
"results": [
  "All packages providing httpd are up to date",
  "All packages providing php are up to date",
  "All packages providing perl are up to date",
  "All packages providing python are up to date",
  ""
]
}
META: ran handlers
META: ran handlers

PLAY RECAP *****
lta-test-target01    : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

```

Progress status(Error log)

Contact administrator

[illegible]

### Register data to Parameter sheet

### Execute Jobflow (Conductor)

## Check Execution results

## 5.4 Check Execution results (2/2)

### Execution results

- Check that the new installed Maria DB's dependency with other packages are correct and that the other 4 packages (htpd,php,perl,python) has been updated.

Progress log (Execution log)

```
Package httpd-2.4.6-97.el7.centos.x86_64 already installed and latest version¥  
Package php-5.4.16-48.el7.x86_64 already installed and latest version¥  
Package 4:perl-5.16.3-299.el7_9.x86_64 already installed and latest version¥  
Package python-2.7.5-90.el7.x86_64 already installed and latest version¥
```

```
~~~~~Abbreviation~~~~~  
~~~
```

Installed:

```
mariadb-server.x86_64 1:5.5.68-1.el7
```

Dependency Installed:

```
mariadb.x86_64 1:5.5.68-1.el7  
perl-Compress-Raw-Bzip2.x86_64 0:2.061-3.el7  
perl-Compress-Raw-Zlib.x86_64 1:2.061-4.el7  
perl-DBD-MySQL.x86_64 0:4.023-6.el7  
perl-DBI.x86_64 0:1.627-4.el7  
perl-IO-Compress.noarch 0:2.061-2.el7  
perl-Net-Daemon.noarch 0:0.48-5.el7  
perl-PIRPC.noarch 0:0.2020-14.el7
```

Dependency Updated:

```
mariadb-libs.x86_64 1:5.5.68-1.el7
```

Complete!

```
"  
]  
}
```

Register Operation name  
(Operation)

Register data to  
Parameter sheet

Execute Jobflow (Conductor)

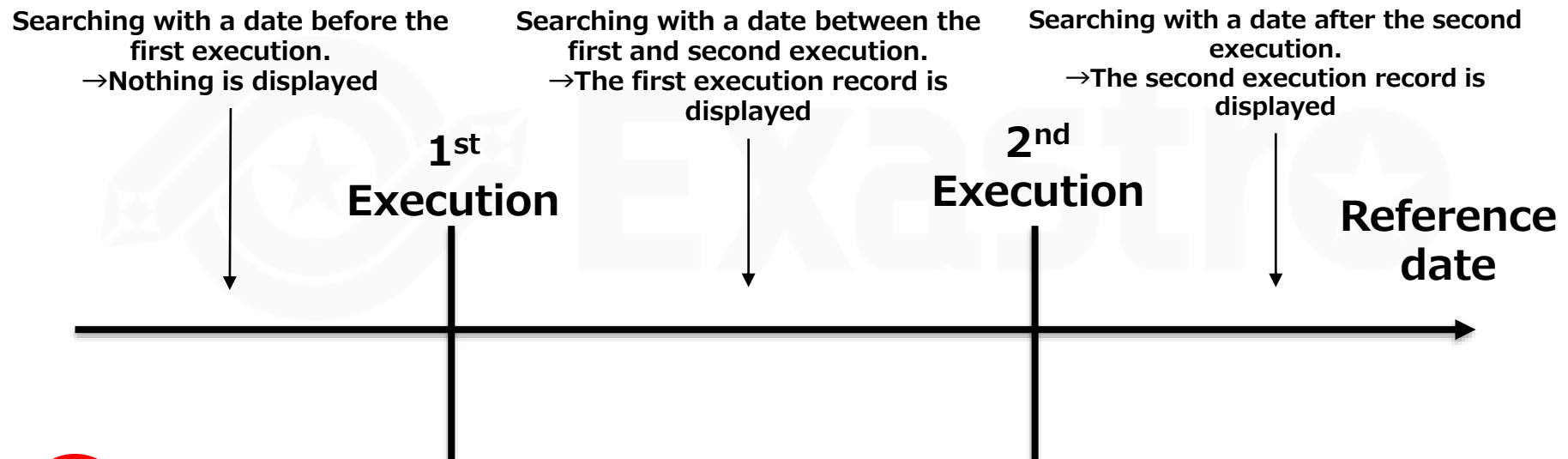
Check Execution results

## 6. Checking the CMDB Parameter history

# 6.1 Executing operations and History Management

## Scenario and History Management

- ITA Manages parameter history and keeps track on who last used it, when it happened and why in the CMDB.
- ITA also comes with functions that are able to extract the parameters of the system at said time. By historically managing parameters, designers and operators both can perform system maintenance without any worries or problems



### Point

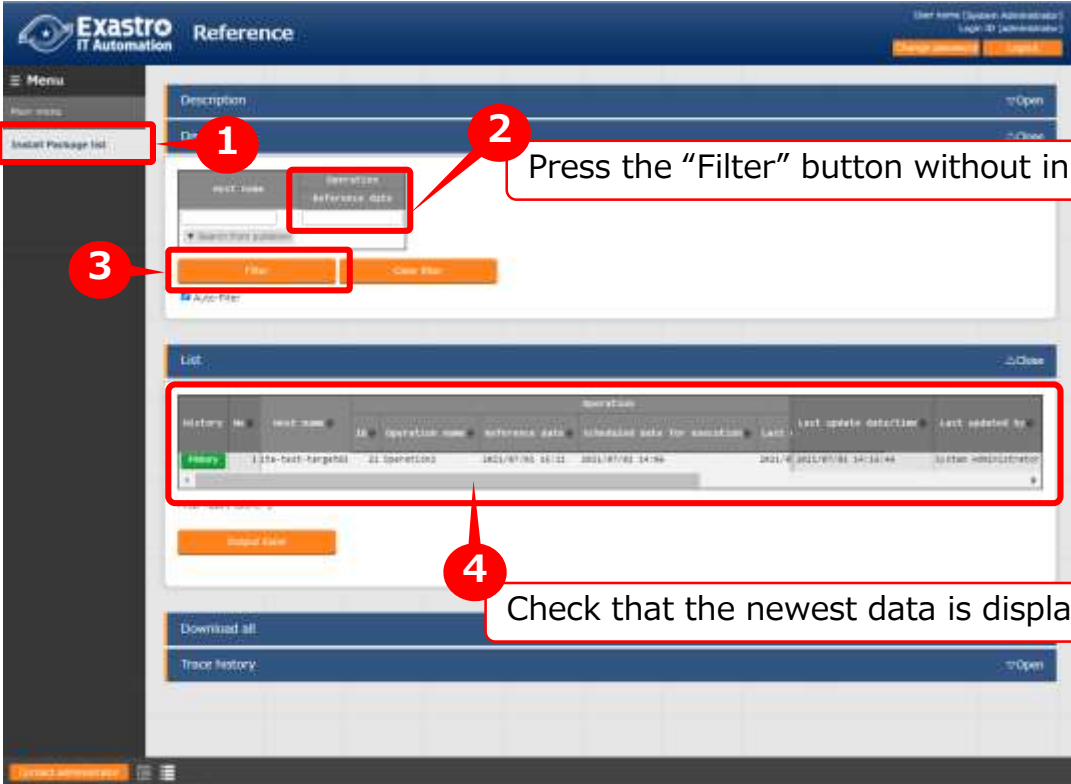
In order for the user to experience history management of Parameters, this scenario contained 2 executions.

## 6.2 Checking the CMDB Parameter history (1/3)

### History Check

- Check if the parameters are actually managed.

From the "Reference" menu group, go to the "Install package list" menu. First, press the "Filter" button without inputting any filters.



The screenshot displays the Exastro IT Automation Reference interface. The left sidebar shows the 'Menu' with 'Install Package List' highlighted. The main area shows the 'Description' tab with a 'Filter' button. A red box highlights the 'Filter' button, and a callout points to it with the text 'Press the "Filter" button without inputting anything'. Below the filter, a table titled 'List' shows the history of package installations. A red box highlights the table, and a callout points to it with the text 'Check that the newest data is displayed correctly.'.

| History | test name          | id | Operation name | reference data   | Scheduled date for execution | Last             | Last update date/time | Last added by        |
|---------|--------------------|----|----------------|------------------|------------------------------|------------------|-----------------------|----------------------|
| History | 13ta-test-target01 | 21 | Operations     | 1821/07/06 16:11 | 2021/07/06 14:56             | 2021/07/06 14:56 | 2021/07/06 14:56      | System Administrator |

## 6.2 Checking the CMDB Parameter history (2/3)

### History Check

- Now, we will input a reference date that took place earlier than the second execution and filter.

The screenshot displays the 'Reference' section of the Exastro IT Automation interface. It includes a 'Display filter' section with a 'Reference date' field and a 'Filter' button. A table below shows the execution history with columns for 'operation', 'reference date', 'scheduled date for execution', 'start', 'end', 'last update date/time', and 'last updated by'. The first row is highlighted in green, indicating the first execution. Red callouts and boxes highlight the 'Reference date' field, the 'Filter' button, and the first row of the table.

1 Input a date earlier than the second execution date.

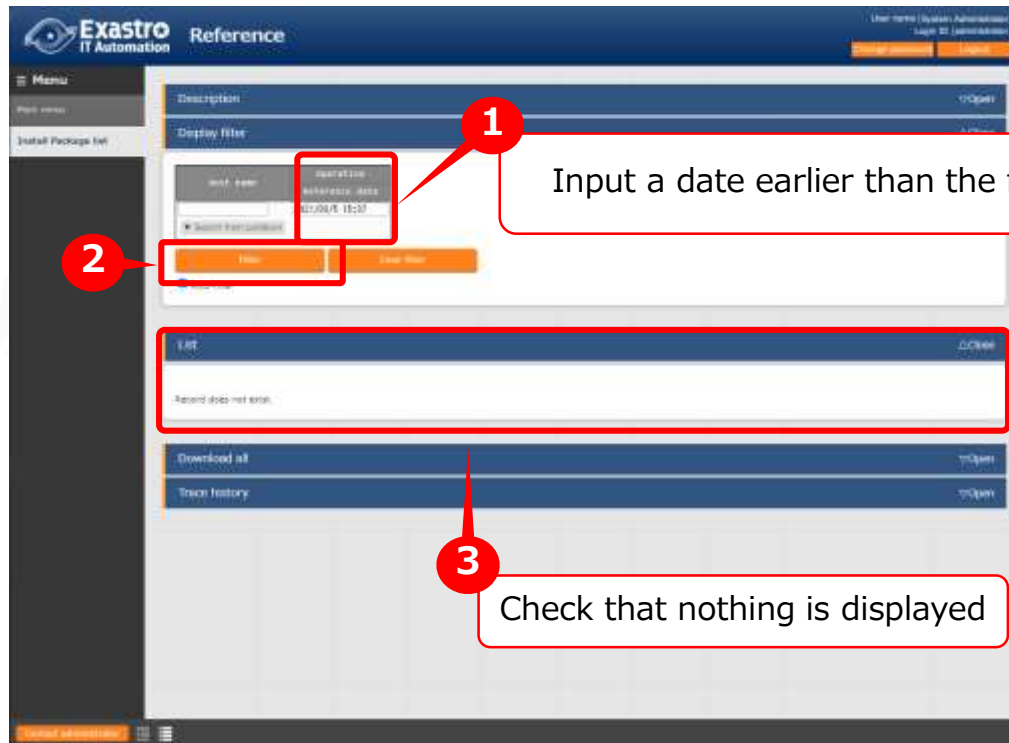
2

3 Check that only the first execution is displayed.

## 6.2 Checking the CMDB Parameter history (3/3)

### History Check

- Lastly, input a date earlier than the first execution.





# A Appendix

# Reference ① 【Ansible-Legacy】 Single Execution

## Execution menu

- Ansible-Legacy has a "Execution" menu where users can execute individual movements and dry run them.

The screenshot displays the Ansible-Legacy web interface. The left sidebar contains a 'Menu' with options: Main menu, Movement list, Playbook file, Movement playbook link, Substitution value auto-registration setting, Target host, Substitution value list, Execution, Check operation status, and Execution list. The main content area shows the 'Execution' menu selected, with sub-panels for 'Description', 'Scheduling', 'Movement [Filter]', and 'Movement [List]'. The 'Movement [List]' panel shows a table with columns: Select, Movement ID, Movement Name, Architecture, Delay time, Host specific format, Dedicated information for ansible, Last update date/time, and Last updated by. A red box highlights the first row, and a red circle with the number '1' points to it with the text 'Select a created Movement'. Below this is the 'Operation [Filter]' and 'Operation [List]' panel. The 'Operation [List]' panel shows a table with columns: Select, No., Operation ID, Operation name, Scheduled date for execution, Last execution date, Access permission, Last update date/time, and Last updated by. A red box highlights the first two rows, and a red circle with the number '2' points to it with the text 'Select an operation linked to the Movement'. At the bottom, the 'Movement ID: 17' and 'Movement Name: #packageinstall' are shown. A red box highlights the 'Dry run' and 'Execute' buttons, and a red circle with the number '3' points to them with the text 'Dry run'. A red circle with the number '4' points to the 'Execute' button with the text 'Execute'.

**1** Select a created Movement

**2** Select an operation linked to the Movement

**3** Dry run

: Checks the playbook's connectivity and syntax

**Execute**

: Executes playbook.

# Reference ② 【Ansible-Legacy】 Execution check

## Execution result

- Pressing either the Execute or the Dry run button will move the user to a screen where execution status and logs are displayed.

The screenshot shows the Exastro Ansible-Legacy interface. The main panel displays the execution status and logs. The 'Progress status(Execution log)' section is highlighted with a red box. Below it, the 'Input data' and 'Result data' sections are also highlighted with a red box. The 'Input data' section shows the 'Host specific forest' as 'dc01m01' and the 'Result data' section shows the 'Host specific forest' as 'dc01m01'.

| Item                              | Value                         |
|-----------------------------------|-------------------------------|
| Execution No.                     | 10                            |
| Execution Type                    | Normal                        |
| Status                            | Completed                     |
| Execution engine                  | Ansible engine                |
| Caller system                     |                               |
| Caller condition                  |                               |
| Execution user                    | System Administrator          |
| ID                                | 17                            |
| Name                              |                               |
| Delay time (seconds)              |                               |
| Dedicated information for ansible | Host specific forest: dc01m01 |
| No.                               | 21                            |
| Name                              | Operation 1                   |
| ID                                | 21                            |
| Host management                   |                               |
| Substituted value                 |                               |
| Input data                        | Populated data                |
| Output data                       | Result data                   |
| Execution status                  |                               |
| Start date/time                   | 2023/07/01 15:41:58           |
| End date/time                     | 2023/07/01 15:44:06           |

Progress status(Execution log)

Filter:  ☐ Display only corresponding lines

```
{"host": "dc01m01", "msg": "All packages providing httpd are up to date.", "result": "ok", "skipped": false, "stdout": "All packages providing httpd are up to date.", "stderr": ""}
```

Point

Here you can see the input data and the execution status.

Point

Here you can see both the execution log and the error log in real time.

Point

Here you can download both the input data and the result data.



**Exastro**