



ITA_User instruction manual

Ansible-driver

— Version 1.10 —

Disclaimer

All the contents of this document are protected by copyright owned by NEC Corporation.
Unauthorized reproduction or copying of all or part of the contents of this document is prohibited.
The contents of this document are subject to change without prior notice in the future.
NEC Corporation is not responsible for any technical or editorial errors or omissions in this document.
NEC Corporation do not guarantee accuracy, usability, certainty of the content in this document.

Trademark

- Linux is registered trademark or trademark of Linus Torvalds, registered in the U.S. and other countries.
- Red Hat is registered trademark or trademark of Red Hat, Inc., registered in the U.S. and other countries.
- Apache, Apache Tomcat, Tomcat are registered trademarks or trademarks of Apache Software Foundation.
- Ansible is a registered trademark or trademark of Red Hat, Inc.
- Ansible Automation Controller is a registered trademark or trademark of Red Hat, Inc.
- Ansible Automation Platform is a registered trademark or trademark of Red Hat, Inc.

The names of other systems, company name and products mentioned in this document are registered trademarks or trademarks of their respective companies.

The ® mark and TM mark is not specified in this document.

※「Exastro IT Automation」is written as「ITA」in this document.

※"Ansible Tower" changed name to "Ansible Automation controller" in Ansible Automation Platform 2.0.

This document contains both Ansible Tower and Ansible Automation Controller.

Table of contents

Table of contents	2
Introduction	4
1 Ansible Driver Overview	5
1.1 Ansible Core	5
1.2 Ansible Automation Controller	5
1.3 About Ansible driver	6
2 Variable handling in Ansible driver	7
2.1 Variable type	7
2.2 Extract variables and register specific values	10
2.3 Variable handling according to substitution value registration	12
3 Ansible driver console menu configuration	13
3.1 Menu/screen list	13
4 Ansible driver operation procedure	16
4.1 Workflow	16
4.1.1 Workflow of Ansible-Legacy	16
4.1.2 Workflow of Ansible-Legacy Role	19
4.1.3 Workflow of Ansible-Pioneer	22
5 Ansible driver function-operation method explanation	26
5.1 Basic console	26
5.1.1 Device list	26
5.1.2 Input operation list	32
5.2 Ansible common console	33
5.2.1 Interface information	33
5.2.2 Ansible Automation Controller host list	39
5.2.3 Global variable list	41
5.2.4 Template list	43
5.2.5 File list	47
5.2.6 Collection interface information	49
5.2.7 Collection item value list	49
5.3 Ansible-Legacy/Legacy Role/Pioneer console	50
5.3.1 OS type master	50
5.3.2 Movement list	52
5.3.3 Playbook file list (Ansible-Legacy only)	55
5.3.4 Role package list (Ansible-Legacy Role only)	57
5.3.5 Dialog type list (Ansible-Pioneer only)	59
5.3.6 Dialog files (Ansible-Pioneer only)	61
5.3.7 Movement details	63
5.3.8 Nested variable list (Ansible-Legacy Role only)	65
5.3.9 Substitution value auto-registration setting	68
5.3.10 Target host	75
5.3.11 Substitution value list	76
5.3.12 Check operation status	82
5.3.13 Execution list	85
5.3.14 Execution	86
6 How to write construction code	88

6.1	Write Playbook (Ansible-Legacy)	88
6.2	Write Dialog file (Ansible-Pioneer).....	89
6.3	Write role package (Ansible-Legacy Role)	108
6.4	Write ITA readme (Ansible-Legacy Role only).....	112
6.5	Write translation table (Ansible-Legacy Role only).....	114
6.6	ita_readme file and translation table (Ansible-Legacy Role only).....	117
6.7	BackYard contents	127
6.8	Ansible usage guideline ITA additional rules	130
7	Application operation.....	131
7.1	Maintenance	131
7.2	About the maintenance method.....	132
8	Appendix	133
8.1	The linkage between the input data used during Ansible execution and ITA menu.....	133
8.1.1	Ansible-Legacy input data.....	134
8.1.2	Ansible-Pioneer input data	136
8.1.3	Ansible-LegacyRole input data	138
8.1.4	Directly executing the input data	140
8.2	Result data created during Ansible execution	141
8.2.1	Legacy/LegacyRole List of files saved in result data	141
8.2.2	List of files saved in Pioneer result data	142
8.3	Option Parameter list	144
8.4	Using ITA proprietary variables in Ansible Automation Controller.	147
8.5	Data resources deleted when executing.	149

Introduction

This document explains the function and the operation method of ITA.

1 Ansible Driver Overview

This chapter explains Ansible, Ansible Automation Controller and Ansible driver.

1.1 Ansible Core

Ansible Core is a platform construction automation tool that makes deploying applications/systems to many construction management targets easy.

Ansible Core can implement various operations by describing YAML textfiles called Playbook which record routine operations and executing them.

Tasks are linked to processing programs called modules, and can control various devices.

For more information regarding Ansible Core, please refer to the Ansible Core manual.

1.2 Ansible Automation Controller

Ansible Automation Controller is a management platform that extends the function of Ansible, a platform construction automation tool such as "access control", "job scheduling", "task visualization", etc.

Ansible can combine "project", "inventory", "credentials" to create "job template" and execute with Ansible.

By combining multiple "Job Templates" to create a "Workflow Job Template", a more diverse workflow can be expressed.

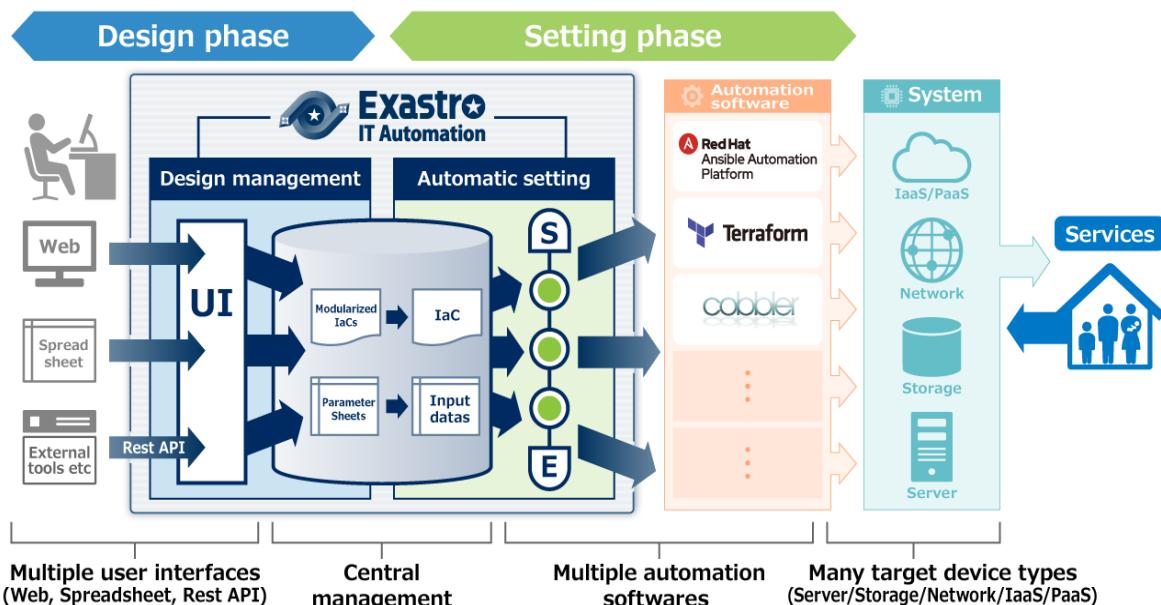
For more information regarding Ansible Automation Controller, please refer to the Ansible Automation Controller product manual.

For the information regarding ITA compatible Ansible Automation Controller versions, please refer to "System Configuration/Environment Construction Guide-Ansible-driver".

Please note that the notation according to the newest version may not be used.

1.3 About Ansible driver

Ansible driver operates as options of ITA system. Ansible driver selects using Ansible Core or Ansible Automation Controller and perform actual operation configuration automatically according to each construction target server, storage, network device registered on ITA system.



The following 3 modes are available in Ansible driver according to the usage.

① Legacy mode

Configure settings to various hosts using the standard function of Ansible.

Register the construction code in a single YAML file and construct the operation pattern with that combination.

This mode is assumed to be used in works such as environment configuration of operation system and network

② Legacy Role mode

Similar with the Legacy mode, configure settings to various hosts using the standard function of Ansible.

Register construction code as package and construct work pattern with the combination of Role. It is assumed that this mode will be used when using the Role package provided by the product department to install product or construct environment, etc.

③ Pioneer mode

Possible to add individual module to Ansible and input setting in interactive mode.

Supports all devices that can login using Telnet or SSH, regardless of server, storage or network device.

Since interacting with target device directly is required, appropriate IT skills are required.

In addition, Ansible driver can configure the variables in Playbook from screen. For details please refer to "[Variable handling in Ansible driver](#)" in this manual.

2 Variable handling in Ansible driver

2.1 Variable type

Ansible driver can set the specific value of Playbook variable in the setting screen in ITA.

※For the detail of setting method, please refer to "[5.3.11 Substitution value list](#)" in this manual

8 of the Playbook variables can be used as variables in ITA. See the table below for more information..

Table2.1 variable type

Type	Content	Legacy	Pioneer	Legacy Role
Normal variable	A variable that can define one specific value to the variable name. Please write the variable in Playbook as {{△VAR_xxx△}}. △:half-width space xxx: half-width alphanumeric character and underscore (_) e.g.) VAR_users: root	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Multiple specific value variable	A variable that can define multiple specific value for a variable name. Please write the variable in Playbook as {{△VAR_xxx△}}. △:half-width space xxx: half-width alphanumeric character and underscore(_) e.g.) VAR_users: - root - mysql	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Nested variable	Hierarchical variables. Please write the variable in Playbook as {{△VAR_xxx△}}. △:half-width space xxx: half-width alphanumeric character and underscore(_) e.g.) VAR_users: - user name: alice authorized: password The member variable name can use ascii characters (0x20 ~ 0x7e) except for the following 7 characters. " . [] ' ¥ : For more information, please see Yaml syntax .	<input type="checkbox"/>	<input type="checkbox"/>	<input type="radio"/>
Global variable	Variable registered from the "Global variable" menu.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Template	Variable registered from the "Template list" menu.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

embedded variable				
File embedded variable	Variable registered from the "File list" menu.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Type	Content	Legacy	Pioneer	Legacy Role																						
ITA original variable	<p>Original variable defined by ITA. The following items in the basic console device list can be handled as variables.</p> <table border="1"> <thead> <tr> <th>Item name</th> <th>variable name</th> </tr> </thead> <tbody> <tr> <td>host name</td> <td><code>__loginhostname__</code></td> </tr> <tr> <td>protocol</td> <td><code>__loginprotocol__</code></td> </tr> <tr> <td>login user ID</td> <td><code>__loginuser__</code></td> </tr> <tr> <td>login password</td> <td><code>__loginpassword__</code></td> </tr> </tbody> </table> <p>The 「__」 surrounding the variable name is a pair of 2 half-width underscore.</p> <p>For "device list", please refer to "User instruct manual_basic console"</p> <p>Operations when executing can be handled as the following variable.</p> <table border="1"> <thead> <tr> <th>Item name</th> <th>Variable name</th> </tr> </thead> <tbody> <tr> <td>Operation</td> <td><code>__operation__</code></td> </tr> </tbody> </table> <p>Setting value : Scheduled date/time for execution <code>YYYY/MM/DD HH:MM</code> <code>_operation</code> ID : operation name</p> <p>Directory path when executing can be handled as the following variable.</p> <table border="1"> <thead> <tr> <th>Item name</th> <th>Variable name</th> </tr> </thead> <tbody> <tr> <td>Operation directory path</td> <td><code>__workflowdir__</code></td> </tr> </tbody> </table> <p>By creating a file under the operation directory path in Playbook, users can download the result data file of "<u>operation execution</u>" menu.</p> <p>The directory path shared by each Movement during Symphony execution can be handled as following variable.</p> <table border="1"> <thead> <tr> <th>Item name</th> <th>Variable name</th> </tr> </thead> <tbody> <tr> <td>Symphony Operation directory path</td> <td><code>__symphony_workflowdir__</code></td> </tr> </tbody> </table> <p>By creating files under the Symphony operation directory path in Playbook, files can be shared between each Movement. Also, when operation is executed from ansible driver, <code>_workflowdir_</code> will be set to same path.</p> <p>The directory path shared by each Movement during</p>	Item name	variable name	host name	<code>__loginhostname__</code>	protocol	<code>__loginprotocol__</code>	login user ID	<code>__loginuser__</code>	login password	<code>__loginpassword__</code>	Item name	Variable name	Operation	<code>__operation__</code>	Item name	Variable name	Operation directory path	<code>__workflowdir__</code>	Item name	Variable name	Symphony Operation directory path	<code>__symphony_workflowdir__</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Item name	variable name																									
host name	<code>__loginhostname__</code>																									
protocol	<code>__loginprotocol__</code>																									
login user ID	<code>__loginuser__</code>																									
login password	<code>__loginpassword__</code>																									
Item name	Variable name																									
Operation	<code>__operation__</code>																									
Item name	Variable name																									
Operation directory path	<code>__workflowdir__</code>																									
Item name	Variable name																									
Symphony Operation directory path	<code>__symphony_workflowdir__</code>																									

	<p>Conducror execution can be handled as following variable.</p> <table border="1"> <thead> <tr> <th>Item name</th><th>Variable name</th></tr> </thead> <tbody> <tr> <td>Conductor</td><td><code>__conductor_workflowdir__</code></td></tr> <tr> <td>Operation directory path</td><td></td></tr> </tbody> </table> <p>By creating files under the Conductor operation directory path in Playbook, files can be shared between each Movements. Also, when operation is executed from ansible driver, <code>_workflowdir_</code> will be set to same path.</p> <p>Each file path of the collect function can be handled as the following variables.</p> <table border="1"> <thead> <tr> <th>Item name</th><th>Variable name</th></tr> </thead> <tbody> <tr> <td>Operation directory (in) [<code>_parameters_</code>] path</td><td><code>__parameters_dir_for_epc__</code></td></tr> <tr> <td>Operation directory (in) [<code>_parameters_file_</code>] path</td><td><code>__parameters_file_dir_for_epc__</code></td></tr> <tr> <td>Operation result directory (out) [<code>_parameters_</code>] path</td><td><code>__parameter_dir__</code></td></tr> <tr> <td>Operation result directory (out) [<code>_parameters_file_</code>] path</td><td><code>__parameters_file_dir__</code></td></tr> </tbody> </table> <p>[<code>_parameters_</code>] : Source file (parameter) for storage destination. [<code>_parameters_</code>] : Collected file for storage destination. ※File placement when the target of the parameter is a file upload column. For more information about the collect function, please refer to the "ITA_User_Instruction_Manual_Collect function".</p>	Item name	Variable name	Conductor	<code>__conductor_workflowdir__</code>	Operation directory path		Item name	Variable name	Operation directory (in) [<code>_parameters_</code>] path	<code>__parameters_dir_for_epc__</code>	Operation directory (in) [<code>_parameters_file_</code>] path	<code>__parameters_file_dir_for_epc__</code>	Operation result directory (out) [<code>_parameters_</code>] path	<code>__parameter_dir__</code>	Operation result directory (out) [<code>_parameters_file_</code>] path	<code>__parameters_file_dir__</code>		
Item name	Variable name																		
Conductor	<code>__conductor_workflowdir__</code>																		
Operation directory path																			
Item name	Variable name																		
Operation directory (in) [<code>_parameters_</code>] path	<code>__parameters_dir_for_epc__</code>																		
Operation directory (in) [<code>_parameters_file_</code>] path	<code>__parameters_file_dir_for_epc__</code>																		
Operation result directory (out) [<code>_parameters_</code>] path	<code>__parameter_dir__</code>																		
Operation result directory (out) [<code>_parameters_file_</code>] path	<code>__parameters_file_dir__</code>																		
Substitution variable	Variable "LCA_XXX" that used to handle variables in Defaults variable definition file or ITA readme other than "VAR_XXX" type in ITA. For details, please refer to " 6.5 Write translation table (Ansible-Legacy Role only) ".	x	x	○															

2.2 Extract variables and register specific values

Users can extract variables from files and playbooks uploaded to ITA and register specific values from the different Ansible menus. The Specific values registered from the Ansible menus are output to the host variable file when executed.

See the section below for extracting variables.

(1) Ansible-Legacy

Extract the variable definitions in the following format from the Playbook uploaded in "Playbook file list (5.3.3 Playbook file list (Ansible-Legacy only) in this manual)".

Format	Specific value settings
<code>{{△VAR_xxx△}}</code> <code>{{△VAR_xxx△ △filter△}}</code>	These specific values can be registered from the " 5.3.9 Substitute value auto registration settings " and "5.3.11 substitute value management" menus. The specific value registration process is different if the user wants to register multiple values.
<code>{{△GBL_xxx△}}</code> <code>{{△GBL_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.3 Global variable list" menu.
<code>{{△TPF_xxx△}}</code> <code>{{△TPF_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.4 Template list" menu.
<code>{{△CPF_xxx△}}</code> <code>{{△CPF_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.5 File list" menu.

※ △:half-width space

xxx: half-width alphanumeric character and underscore (_)

(2) Ansible-Pioneer

Extract the same variable definition as Ansible-Legacy from the dialog file uploaded in "Dialog files(5.3.6 Dialog files (Ansible-Pioneer only) in this manual)"

Format	Specific value settings
<code>{{△VAR_xxx△}}</code> <code>{{△VAR_xxx△ △filter△}}</code>	These specific values can be registered from the " 5.3.9 Substitute value auto registration settings " and "5.3.11 substitute value management" menus. The specific value registration process is different if the user wants to register multiple values.
<code>{{△GBL_xxx△}}</code> <code>{{△GBL_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.3 Global variable list" menu.
<code>{{△TPF_xxx△}}</code> <code>{{△TPF_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.4 Template list" menu.
<code>{{△CPF_xxx△}}</code> <code>{{△CPF_xxx△ △filter△}}</code>	These specific values can be registered from the "5.2.5 File list" menu.

(3) Ansible-Legacy Role

Extract the variable from the Playbook in role package uploaded in "Role package list(5.3.4 Role package list (Ansible-Legacy Role only) in this manual)"

Please refer to "Role package list (5.3.4 Role package list (Ansible-Legacy Role only)) in this

manual)" for details.

By creating translation table, ITA can handle the variables other than "VAR_xxx" defined in defaults variable definition file and ITA readme. Please refer to "["6.5 Write translation table \(Ansible-Legacy Role only\)"](#)" for details.

Defined variables with the following format from Playbooks from uploaded role packages will be extracted.

Format	Role package directory				Specific value settings
	meta	handlers	templates	tasks	
<code>{{△GBL_xxx△}}</code> <code>{{△GBL_xxx△ △filter△}}</code>	<input type="radio"/>	<input checked="" type="checkbox"/>			These specific values can be registered from the "5.2.3 Global variable list" menu.
<code>{{△TPF_xxx△}}</code> <code>{{△TPF_xxx△ △filter△}}</code>					These specific values can be registered from the "5.2.4 Template list" menu.
<code>{{△CPF_xxx△}}</code> <code>{{△CPF_xxx△ △filter△}}</code>					These specific values can be registered from the "5.2.5 File list" menu.

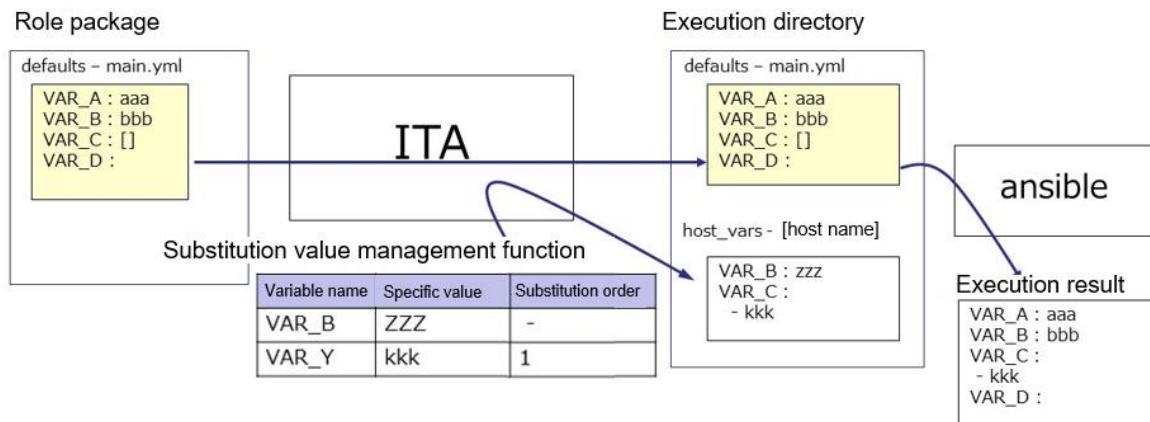
= Playbook with variable definition extracted

= Playbook without variable definition extracted

2.3 Variable handling according to substitution value registration

By using substitution value registration function, it is possible to overwrite the value of variable defined in Playbook.

The relationship between Playbook variable and the variable value registered in substitution value management function is shown as the following figure.



The value of the variable registered in the substitution value management function is output to the variable definition file (`host_vars`) for each host, and executed on each host by using the original Playbook and variable definition file as input in Ansible.

The priority of variable values in the result is as below.

- ① Value registered in substitution value management function
- ② Value specified in Playbook variable

Please refer to "5.3.11 [Substitution value list](#)" for details.

3 Ansible driver console menu configuration

This chapter explains the configuration of ITA console menu.

For the method to log in Web console and the components / basic operations of the menu screen, please refer to the "[First Step Guide](#)".

3.1 Menu/screen list

① ITA basic console menu

The menu list of ITA basic console used in Ansible driver is as below.

Table 3.1-1 Basic console menu/screen list

No	Menu group	Menu·Screen	Description
1	ITA basic console	Device list	Maintain (View/Register/Update/Discard) operation target system list.
2		Linked menu	Manage the configuration management database linked with Substitution value auto-registration setting menu.
3		Input operation list	Maintain(View/Register/Update/Discard) input operation list

② Ansible common console menu

Ansible common console menu list is as below.

Table 3.1-2 Common console menu/screen list

No	Menu group	Menu·Screen	Description
1	Ansible common console	Interface information	Select whether to use Ansible or Ansible Automation Controller server as the execution engine for the construction operation. Manage the path of directory shared by ITA system, Ansible driver server, and execution engine server and the connection interface information to the execution engine server.
2		Ansible Automation Controller host list	Manage the information needed to execute Ansible Automation Controller's RestAPI and the information needed to transfer construction files to Ansible Automation Controller.
3		Global variable list	Manage the variable commonly used in Playbook or dialog file, etc. (referred to as global variable hereafter) and their specific values.
4		Template list	Manage template files and embedded variables used in the template modules, etc. in Playbook.
5		Contents list	Manage the files and embedded variable used in each module in Playbook.

③ Ansible console menu

The menu list according to each Ansible console is as below.

Table 3.1-3 Ansible driver console menu/screen list

No	menu group			Menu·Screen	Hidden menu	Description
	Ansible console	Legacy	Pioneer			
1			○	OS type		Manage the type of OS of the devices.
2	○	○	○	Movement list		Manage the list of Movements registered in Symphony.
3	○			Playbook files		Manage Playbook file.
4		○		Role package list		Manage role package.
5			○	Dialog type list		Manages the type of grouping dialog files of the same purpose as dialog type.
6			○	Dialog files		Manage the OS type linked with the dialog type and the ITA system original format work procedure file (referred to as dialog file in the following).
7	○	○	○	Movement-Playbook link (Movement - Dialog file type link, Movement-Role link)		Manage links between Movements and Playbook files.
8		○		Nested variable list		Manage the maximum iteration array count if nested variable is configured as iterative array.
9	○	○	○	Substitution value auto-registration setting		Manage Movement and variable linked to every item value of operation and host registered in the configuration management database menu.
10	○	○	○	Target host		Manage the host used in Movement.
11	○	○	○	Substitution value list		Manage the substitution value of variable.
12	○	○	○	Execution		Select the Movement and Operation for work execution and indicate the execution.
13	○	○	○	Check operation status		Displays the operation execution status.
14	○	○	○	Execution list		Manage the operation execution history.
15	○	○	○	Variable name list	○	Legacy: Manages variable names used in Playbooks uploaded to the Playbook file collection Pioneer: Manages variable names used by dialogue files uploaded to the Dialogue file collection Legacy Role: Manages variable names defined in the ITAreadme file and other default variable definition files found in the Package "zip" file uploaded to the Role package list menu.

No	menu group			Menu·Screen	Hidden menu	Description			
	Ansible console								
	Legacy	Legacy	Pioneer						
16	<input checked="" type="radio"/>	<input checked="" type="radio"/>	<input checked="" type="radio"/>	Movement variable association list	<input type="radio"/>	Manages variables used in Movements			
17		<input checked="" type="radio"/>		Role name list	<input type="radio"/>	Manages roles registered in the Package "zip" file uploaded to the Role package list menu.			
18		<input checked="" type="radio"/>		Role variable name list	<input type="radio"/>	Manages variables defined in the ITAreadme file and other default variable definition files found in the Package "zip" file uploaded to the Role package list menu. The variables are managed per role.			
19		<input checked="" type="radio"/>		Variable specific value list	<input type="radio"/>	Manages the specific values of variables defined in the ITAreadme file and other default variable definition files found in the Role Package "zip" file uploaded to the Role package list menu.			
20		<input checked="" type="radio"/>		Member variable list	<input type="radio"/>	Manages member variables of nested variables defined in the ITAreadme file and other default variable definition files found in the Role Package "zip" file uploaded to the Role package list menu.			
21		<input checked="" type="radio"/>		Nested variable member list	<input type="radio"/>	Manages the structure of nested variables defined in the ITAreadme file and other default variable definition files found in the Role Package "zip" file uploaded to the Role package list menu.			
22		<input checked="" type="radio"/>		Nested variable array combination list	<input type="radio"/>	Manages repeat cycles of the nested values defined in the ITAreadme file and other default variable definition files found in the Role Package "zip" file uploaded to the Role package list menu.			
23		<input checked="" type="radio"/>		Reading variable list	<input type="radio"/>	Manages variables defined in the conversion table file found in the Role Package "zip" file uploaded to the Role Package list menu.			

※1 Hidden menus are used to update/register data with the Backyard function.

These menus are set to be hidden when you install the Ansible Driver function.

If you want to display the hidden menus, access the "Management console -> Role/Menu link list" menu and restore the menus you wish to display. For more information, please see the

"User_manual_Management_Console".

Please note that updating some of the data in these menus might cause the Backyard function to not function properly.

4 Ansible driver operation procedure

This chapter explains the operation procedure for using each Ansible console.

4.1 Workflow

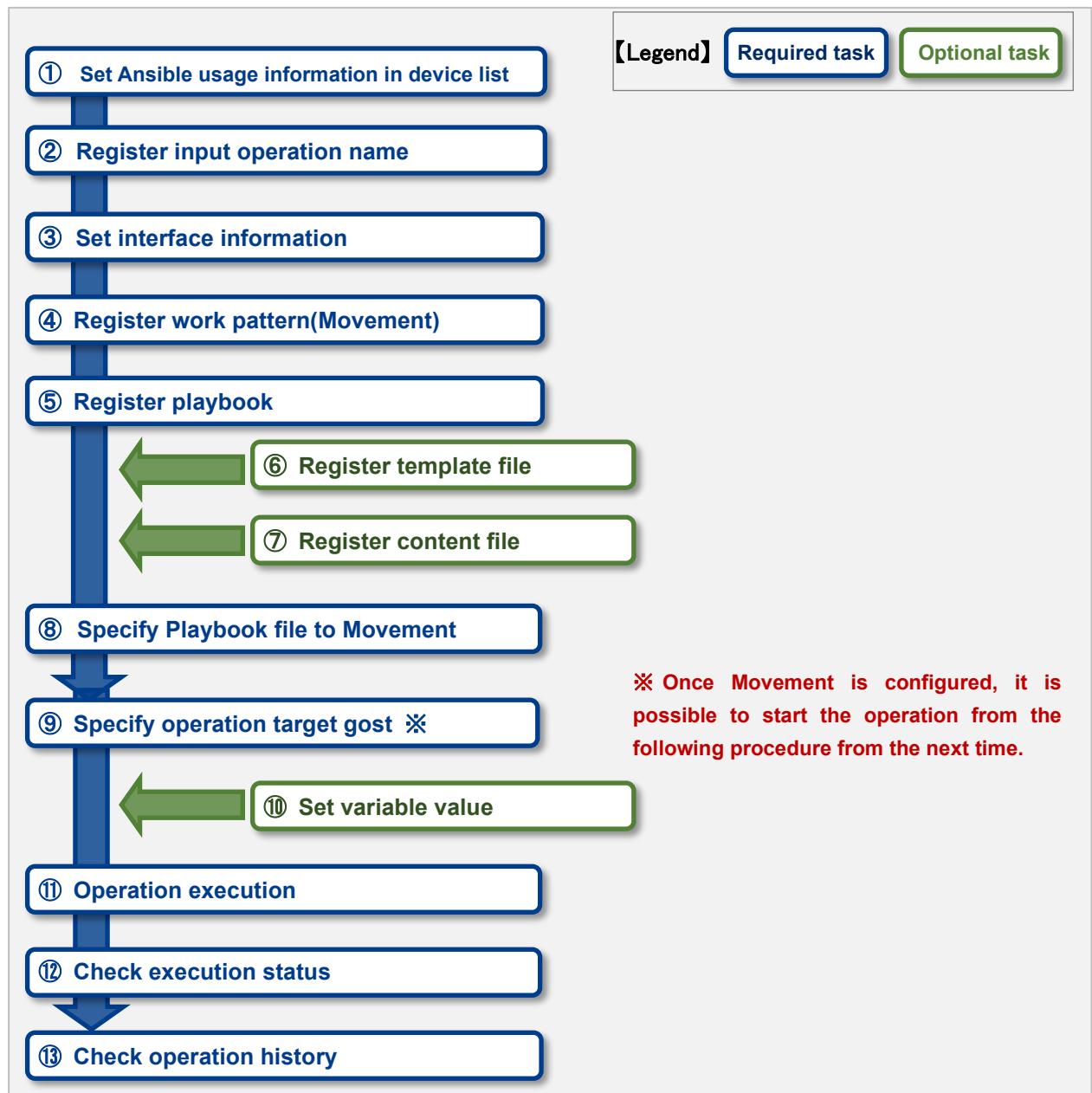
The standard workflow of each Ansible console is as follow.

The details of each operation is writed in the next section.

Please refer to "User instruction manual_Basic console" for how to use the ITA basic console.

4.1.1 Workflow of Ansible-Legacy

The workflow of executing operation using Ansible-Legacy is as follows.



- **Workflow details and references**

- ① **Set Ansible usage information in device list**

Set the Ansible usage information to each devices in the device list screen of ITA basic console.
Please refer to “[5.1.1 Device list](#)” for details.

- ② **Register input operation name**

Register the input operation name for work from the input operation list screen of ITA basic console.
Please refer to “[5.1.2 Input operation list](#)” for details.

- ③ **Set interface information**

Select using whether Ansible Core or Ansible Automation Controller server as the execution engine and register the connection information of the execution engine server from the interface information screen of Ansible common console.
Please refer to “[5.2.1 Interface information](#)” for details.

- ④ **Register work pattern (Movement)**

Register the Movement for operation from the Movement list screen of Ansible-Legacy console.
Please refer to “[5.3.2 Movement list](#)” for details.

- ⑤ **Register playbook**

Register the Playbook used in operation from the Playbook files screen of Ansible-Legacy console.
Please refer to “[5.3.3 Playbook file list \(Ansible-Legacy only\)](#)” for details.

- ⑥ **Register template file (execute if necessary)**

Register/Update/Discard the template file (src) and the template embedded variable used in the template module, etc. of Playbook from the template list screen of Ansible common console.
Please refer to “[5.2.4 Template list](#)” for details.

- ⑦ **Register content file (execute if necessary)**

Register the file used to configure the operation target server from the contents list screen of Ansible common console.

Please refer to “[5.2.5 Contents list](#)” for details.

- ⑧ **Specify Playbook file to Movement**

In the Ansible-Legacy Console -> Movement-Playbook link (Movement-Dialogue file type link, Movement-Role link) screen, specify a Playbook to the registered Movement.

Please refer to “[5.3.7 Movement-Playbook link \(Movement - Dialog file type link, Movement-Role link\)](#)” for details.

- ⑨ **Specify operation target host**

Specify the operation target host from the target host screen of Ansible-Legacy console.
Please refer to “[5.3.10 Target host](#)” for details.

⑩ Set variable value (execute if necessary)

Set the value of the variable in the Playbook which has been registered to Movement from the substitution value list screen in Ansible-Legacy console. If variable is not used, then configuration is not required.

Please refer to "[5.3.11 Substitution value list](#)" for details.

⑪ Operation execution

Select and set execution date, input operation and indicate operation execution from the execution screen of Ansible-Legacy console.

Please refer to "[5.3.14 Execution](#)" for details.

⑫ Check operation status

The status of executed operation is displayed in real time in the "Check operation status" screen of Ansible-Legacy console. In addition, users can perform emergency stop on operation and monitor the execution log and error log.

Please refer to "[5.3.12 Check operation status](#)" for details.

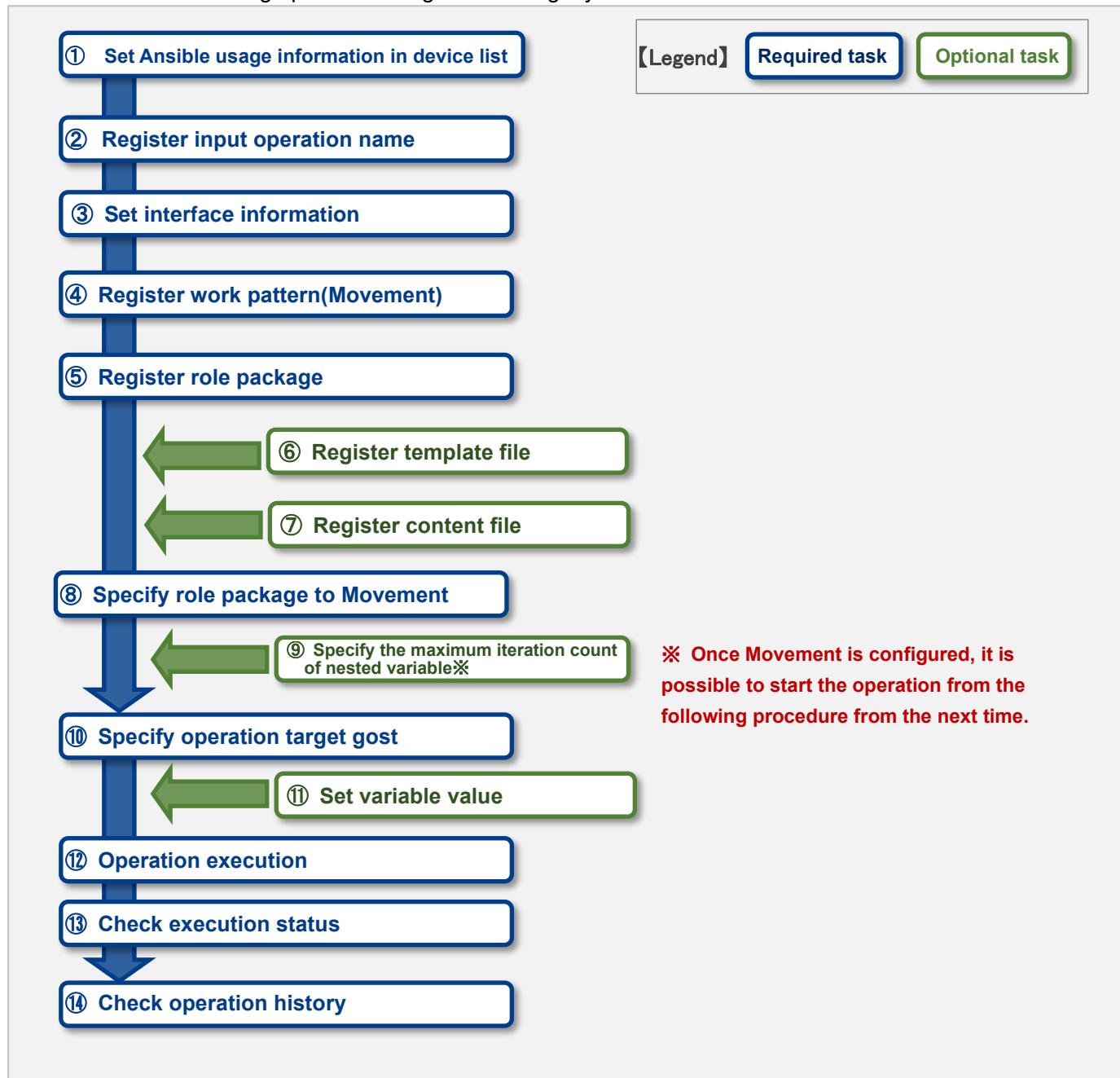
⑬ Check operation history

The list of executed operation is displayed in the execution list screen of Ansible-Legacy console and users can check the execution history.

Please refer to "[5.3.13 Execution list](#)" for details.

4.1.2 Workflow of Ansible-Legacy Role

The workflow of executing operation using Ansible-Legacy Role is as follows.



- **Workflow details and references**

- ① **Set Ansible usage information in device list**

Set the Ansible usage information according to each devices in the device list screen of ITA basic console.

Please refer to “[5.1.1 Device list](#)” for details.

- ② **Register input operation name**

Register the input operation name from the input operation list screen of ITA basic console.

Please refer to “[5.1.2 Input operation list](#)” for details.

- ③ **Set interface information**

Select using whether Ansible Core or Ansible Automation Controller server as the execution engine and register the connection information of the execution engine server from the interface information screen of Ansible common console.

Please refer to “[5.2.1 Interface information](#)” for details.

- ④ **Register work pattern (Movement)**

Register the Movement for operation from the Movement list screen of Ansible-Legacy Role console.

Please refer to “[5.3.2 Movement list](#)” for details.

- ⑤ **Register role package**

Register the role package used in operation from the role package list screen of Ansible-Legacy Role console.

Please refer to “[5.3.4 Role package list \(Ansible-Legacy Role only\)](#)” for details.

- ⑥ **Register template file (execute if needed)**

Register/Update/Discard the template file (src) and the template embedded variable used in the template module, etc. of Playbook from the template list screen of Ansible common console.

Please refer to “[5.2.4 Template list](#)” for details.

- ⑦ **Register content file (execute if needed)**

Register the file used to configure the operation target server from the contents list screen of Ansible common console.

Please refer to “[5.2.5 Contents list](#)” for details.

- ⑧ **Specify role package to Movement**

Specify the Playbook file to the registered Movement from Movement details screen of Ansible-Legacy Role console.

Please refer to “[5.3.7 Movement details](#)” for details.

- ⑨ **Specify the maximum iteration count of nested variable**

Specify the maximum iteration count of the array of member variables defined in nested variables from Nested variable list screen of the Ansible-Legacy Role console.

Please refer to “[5.3.8 Nested variable list \(Ansible-Legacy Role only\)](#)” for details.

- ⑩ **Specify operation target host**

Specify the operation target host from the target host screen of Ansible-Legacy Role console.

Please refer to “[5.3.10 Target host](#)” for details.

(11) Set variable value (execute if needed)

Set the value of the variable in the Playbook which has been registered to Movement from the substitution value list screen in Ansible-Legacy Role console. If variable is not used, then configuration is not required.

Please refer to “[5.3.11 Substitution value list](#)” for details.

(12) Operation execution

Select and set execution date, input operation and instruct operation execution from the execution screen of Ansible-Legacy Role console.

Please refer to “[5.3.14 Execution](#)” for details.

(13) Check operation status

The status of executed operation is displayed in real-time in the "Check operation status" screen of Ansible-Legacy console. In addition, users can perform emergency stop on operation and monitor the execution log and error log.

Please refer to “[5.3.12 Check operation status](#)” for details.

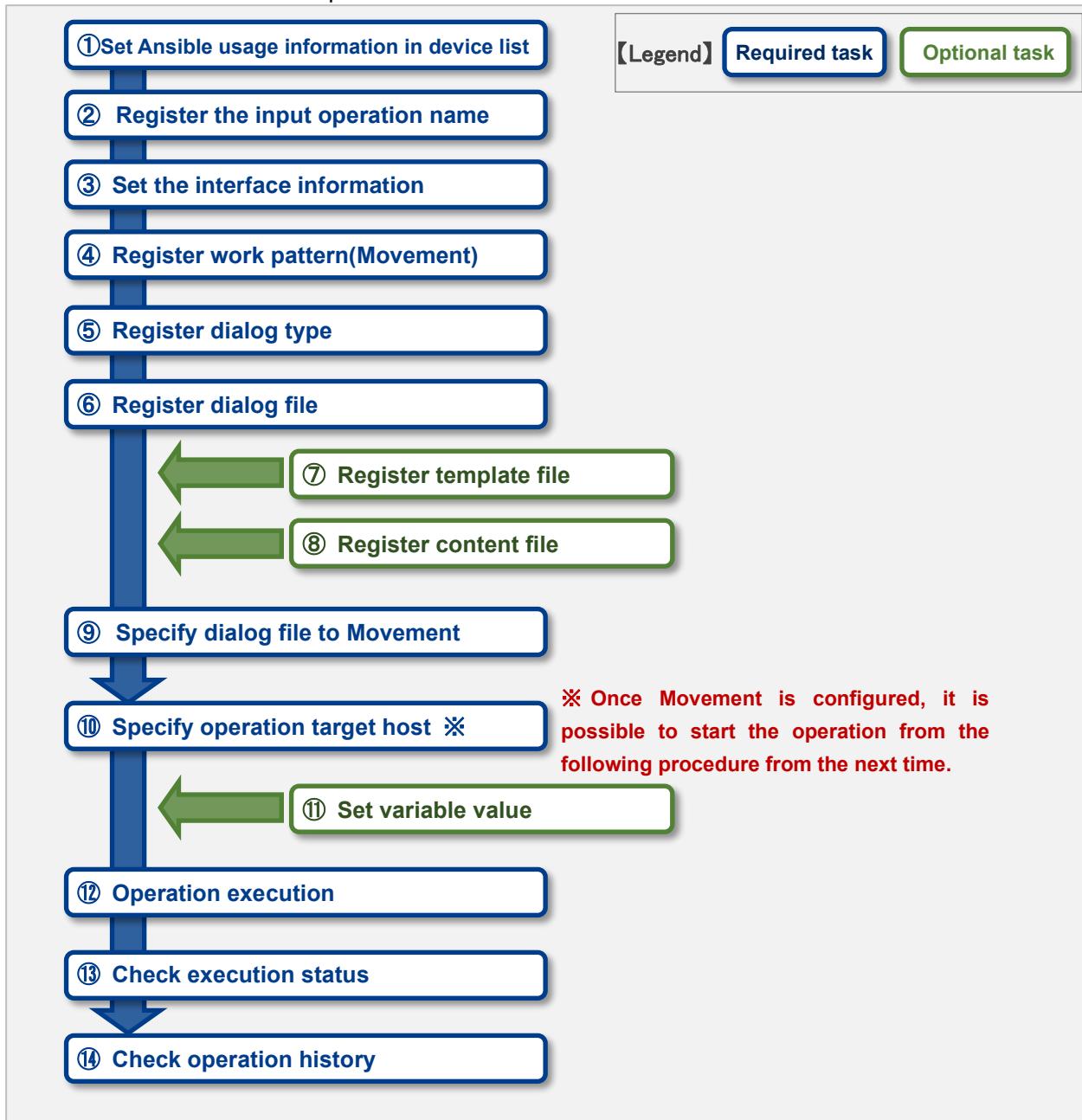
(14) Check operation history

The list of executed operation is displayed in the execution list screen of Ansible-Legacy Role console and users can check the execution history.

Please refer to “[5.3.13 Execution list](#)” for details.

4.1.3 Workflow of Ansible-Pioneer

The workflow to execute the operation in Ansible-Pioneer is as follows.



- **Workflow details and references**

- ① **OS type registration**

Set the OS type of the device to be operated from Pioneer.

- ② **Set Ansible usage information in device list**

Set the Ansible usage information for each device from the device list screen of the ITA basic console.

For details, please refer to "[5.1.1 Device list](#)".

③ Register the input operation name

Register the input operation name from the input operation list screen of ITA basic console.

Please refer to “[5.1.2 Input operation list](#)” for details.

④ Register the interface information

Select using whether Ansible Core or Ansible Automation Controller server as the execution engine and register the connection information of the execution engine server from the interface information screen of Ansible common console.

Please refer to “[5.2.1 Interface information](#)” for details.

⑤ Register work pattern (Movement)

Register the Movement for operation from the Movement list screen of Ansible-Pioneer console.

Please refer to “[5.3.2 Movement list](#)” for details

⑥ Register dialog type

Register dialog type from the dialog type list screen of Ansible-Pioneer console.

Ansible-Pioneer defines the differences for each OS type in each dialog file, and combines the same purpose dialog file as dialog type to absorb (abstract) the device difference.

Please refer to “[5.3.5 Dialog type list \(Ansible-Pioneer only\)](#)” for details.

⑦ Register dialog file

Register dialog file according to the combination of dialog type and OS type from the dialog files screen of Ansible-Pioneer console.

Please refer to “[5.3.6 Dialog files \(Ansible-Pioneer only\)](#)” for details.

⑧ Register template file (execute if needed)

Register/Update/Discard the template file (src) and the template embedded variable used in the template module, etc. of Playbook from the template list screen of Ansible common console.

Please refer to “[5.2.4 Template list](#)” for details.

⑨ Register content file (execute if needed)

Register the file used to configure the operation target server from the contents list screen of Ansible common console.

Please refer to “[5.2.5 Contents list](#)” for details.

⑩ Specify dialog file to Movement

Specify dialog file to the registered Movement from movement details screen of Ansible-Legacy Role console.

Please refer to “[5.3.7 Movement details](#)” for details.

⑪ Specify operation target host

Specify the operation target host from the target host screen of Ansible-Pioneer console.

Please refer to “[5.3.10 Target host](#)” for details.

⑫ Set variable value (execute if needed)

Set the value of the variable in the Playbook which has been registered to Movement from the substitution value list screen in Ansible-Pioneer console. If variable is not used, then configuration is not required.

Please refer to “[5.3.11 Substitution value list](#)” for details.

⑬ Operation execution

Select and set execution date, input operation and indicate operation execution from the execution screen of Ansible-Pioneer console.

Please refer to “[5.3.14 Execution](#)” for details.

⑭ Check operation status

The status of executed operation is displayed in real time in the Check operation status screen of Ansible-Pioneer console. In addition, users can perform emergency stop on operation or monitor the execution log and error log.

Please refer to “[5.3.12 Check operation status](#)” for details.

⑮ Check operation history

The list of executed operation is displayed in the execution list screen of Ansible-Pioneer console and users can check the execution history.

Please refer to “[5.3.14 Execution list](#)” for details.

■Legend of Registration screen item list

The content of the Registration screen item list are writed in the next section.

① Item	② Description	③ Input required	④ Input type	⑤ Restrictions

① Item

- The item name in the submenu.

② Description

- The description for the item.

③ Input required

- ○ : Items that entering contents are required for them.
- - - : Items that entering contents are optional for them.

④ Input type

- Manual: Items that require manual input.
- Auto: Items whose content are entered automatically.
- Checkbox: Check box format item.
- Button: Radio button format item.
- List: List box format item.

⑤ Restrictions

- The restrictions for the item(Limitation on number of characters, etc.)

5 Ansible driver function · operation method explanation

This chapter explains each console function used in Ansible driver.

5.1 Basic console

This section writes the operation of ITA basic console.

Please refer to the ITA basic console manual for this operation and perform the operation in the ITA basic console screen.

5.1.1 Device list

- (1) Registration/Update/Discarding information of operation target host is performed in the "Device list" menu.

This document explains the items (red frame) in the device list required for Ansible driver operations.

Please see the "Exastro-ITA_User_Instruction_Manual_Basic_Console.pdf" together with this document.

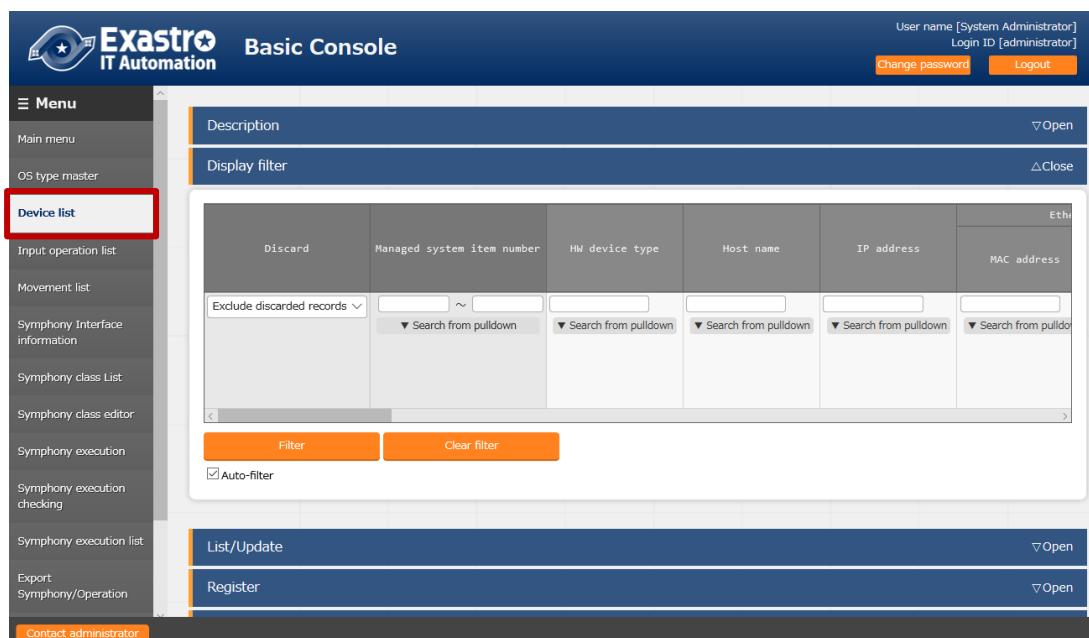


Figure 5.1-1 Submenu screen (Device list)

- (2) Click the "Register" - "Start Registration" button to register the device information.

Figure 5.1-2 Registration screen(Device list-common item)

Figure 5.1-3 Registration screen(Device list-Ansible usage information)

- (3) The list of common item in registration screen is as follows.

Input of the columns with a red asterisk (*) after their column name in the web screen is required.

In the case of using Ansible driver, please enter the usage information of Ansible.

If operation is executed while required column is not entered, unexpected errors may occur.

Table 5.1-1 Registration screen item list (Device list)

Item	Description		Input required	Input type	Restrictions
Managed system item number	A unique ID that identifies the registration information is entered automatically.		-	Auto	-
host name	Enter host name. If you set the hostname to localhost and use pioneer as the working host, you may get an error when executing the operation. In that case, please add the path to the python3 file installed on the ansible server in the following parameter to the add inventory file option. Exp) ansible_python_interpreter: /usr/bin/python3		<input type="radio"/>	Manual	Maximum length 128 bytes
IP address	Enter IP address in xxx.xxx.xxx.xxx format.		<input type="radio"/>	Manual	Maximum length 15 bytes
EtherWake OnLan	MAC address	Enter MAC address.		-	Manual
	Network	Enter network device name.		-	Manual

Item		Description	Input required	Input type	Restrictions
device name					256 bytes
Login user ID		Enter network device name.	<input checked="" type="radio"/>	Manual	Maximum length 30 bytes
Login password	Management	Select "●" when using ITA to manage password.	<input checked="" type="radio"/>	List	-
	Login password	Specify password.	<input checked="" type="radio"/>	Manual	Maximum length 30 bytes
ssh authentication key file		Specify the ssh authentication key file and enter the file when using key authentication. Required when specifying the ssh authentication key file if authentication method is the key authentication.	-	File	Maximum size 10K bytes
Ansible dedicated information	Authentication method	Select the authentication method used when connecting from Ansible/Ansible Automation Controller to the target device. ● Password Authentication If you also choose ● for Login password Management, you will be required to input a login password. ● Key Authentication (No passphrase) You must upload an SSH secret key file (id_ras). ● Key Authentication (With passphrase) You must upload an SSH seccret key file (id_ras) and input a passphrase. ● Key Authentication (Key Exchanged)※1 You will not be required to upload an SSH secret key file (id_ras) ● Password Authentication (winrm) If necessary, input the WinRM connection information. If you choose anything else than Password authentication (winrm), you must configure the following settings for the target device. -Set the Login-user's sudo permissions to /etc/sudoers with NOPASSWD Example) Demo_user ALL=(ALL) NOPASSWD:ALL	<input checked="" type="radio"/>	List	As writed in the description column.
		Port no.	-	Manual	As writed in the description column
	WinRM connection	Server certificate	-	File	Maximum size 10K bytes

Item				Description	Input required	Input type	Restrictions
				port To omit server certificate authentication, add the following to the inventory file additional option. ansible_winrm_server_cert_validation=ignore			
Pioneer dedicated information	Protocol		<p>Select the protocol (ssh/telnet) for when logging in to the target device.</p> <ul style="list-style-type: none"> ● If you selected ssh Select something other than "Password Authentication (winrm)" for Authentication method. ● If you selected telnet you will connect to telnet without using the set value for Authentication method. 		<input type="radio"/>	List	-
	OS type		<p>Select the OS of target device.</p> <p>The OS types registered in the OS type master are displayed in list.</p>		<input type="radio"/>	List	-
Connection options		<p>(In the case of connecting via ssh)</p> <p>If users want to set options other than the ssh options set in /etc/ansible.cfg/ssh_args , please enter the desired option.</p> <p>(In the case of connecting via telnet)</p> <p>If users want to set options when connecting via telnet, please enter the desired option.</p>		-	Manual	Maximum length 512 bytes	
Inventory file addition option		<p>Enter the option parameter of inventory file that is not set in ITA.</p> <p>e.g.)</p> <pre>ansible_connection: network_cli ansible_network_os: nxos</pre>		-	Manual	Maximum length 512 bytes	
Tower dedicated information	Instance group name※2	<p>If the Ansible Automation Controller is a Cluster configuration, select which Ansible Automation Controller instance group it should be executed in.</p> <p>The instance group set here will be set to the Tower's Evently Objects.</p> <p>If nothing is selected, the default Ansible Automation Controller value will be used.</p> <p>If the Ansible Automation Controller in use is not a cluster configuration, you can leave this blank.</p>		<input type="radio"/>	List	-	

Item		Description	Input required	Input type	Restrictions
	Connection type	<p>Set the connection type for Ansible Automation Controller authentication credentials. Normally Machine is selected. In the case where Ansible_Connection needs to be to local Network OS, Choose Network.</p> <p>If a Network is selected the user must set Platform Options other than (ansible_connection) for the additional inventory file options</p> <p>Exp)</p> <p>Example of Inventory file addition option settings.</p> <p>Set value when Network OS is ios.</p> <pre>ansible_network_os: ios ansible_become: yes ansible_become_method: enable</pre> <p>For more information regarding the Ansible Automation Controller authentication connection type, please refer to the "Authentication type" document.</p> <p>For information regarding Network OS, ansible_connection and Platform options, please refer to the "Platform_Options" document.</p>	<input type="radio"/>	List	
Remarks		Free description field.	-	Manual	Maximum length 4000 bytes

※1 Distribution of public key file required when the authentication method is key authentication (key exchanged).

• For Ansible Engine

Make an ssh connection to the target host from the "Ansible common console=>User set in the interface information" of the server where ansible is installed.

Copy the user's public key to the user that will log in to the device's "Authorized keys".

• For Ansible Automation Controller

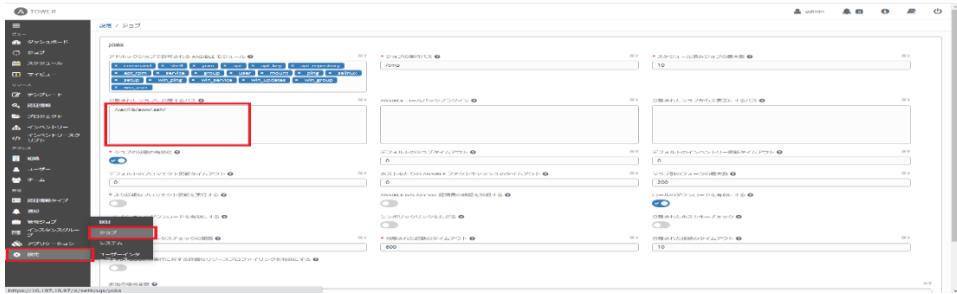
Connect from the Ansible Automation Controller's awx user to the Operation host with SSH.

Copy the awx user's public key to the user that will log in to the devices' "Authorized keys"

You will also need to configure settings in the Tower web's "Setting"->"Job"->"paths to expose to isolated jobs".

For more information, please refer to the "Exastro-

ITA_System_Configuration_Environment_Construction_Guide_Ansible-driver" chapter 5, Ansible Automation Controller Initial settings.



Note that in Ansible Automation Controller4.x and later versions, it is not possible to use the awx user's ssh directory, meaning that it is not possible to connect to the operation host with a key authentication (key changed).

※2 You can select from the data acquired from the Ansible Driver Backyard function, "Ansible Automation Controller Data Synchronization".

5.1.2 Input operation list

- (1) In the "Input operation list" screen, the operations for the target host to be executed by the orchestrator are managed. Operations are selected from the menu in ITA basic console

The screenshot shows the Exastro IT Automation Basic Console interface. At the top, there is a header with the Exastro logo, the text 'Basic Console', and user information ('User name [System Administrator]', 'Login ID [administrator]', 'Change password', 'Logout'). On the left, a vertical sidebar titled 'Menu' lists various options: Main menu, OS type master, Device list, Input operation list (which is highlighted with a red box), Movement list, Symphony Interface information, Symphony class List, Symphony class editor, Symphony execution, Symphony execution checking, Symphony execution list, Export Symphony/Operation, and Contact administrator. The main content area has a title 'Description' with a 'Display filter' section. The filter section contains fields for 'Discard' (set to 'Exclude discarded records'), 'No.' (with search dropdowns), 'Operation ID' (with search dropdowns), 'Operation name' (with search dropdowns), and 'Scheduled date for execution' (with search dropdowns). Below the filter are 'Filter' and 'Clear filter' buttons, and a checked 'Auto-filter' checkbox. Further down, there are sections for 'List/Update' (with 'Register', 'Download all and edit file uploads', and 'Trace history' options), each with a '▽ Open' link.

Figure 5.1-4 Submenu screen (Input operation list)

Please refer to the related manual "User instruction manual_Basic console" for the details of registration method.

5.2 Ansible common console

This section writes the operation of Ansible common console.

5.2.1 Interface information

- (1) In the "interface information" menu, select using whether Ansible Core or Ansible Automation Controller Ansible Automation Controller for the execution engine and register/update/discard the shared directory path between ITA system, Ansible driver server, and execution engine server , and the connection interface information of the execution engine server.

The screenshot shows the Ansible Common interface. On the left, there's a sidebar with a 'Main menu' and a red box highlighting the 'Interface information' option. The main area has tabs for 'Description' (with a ▽Open button) and 'Display filter' (with a △Close button). Below these are two tables: one for 'Discard' (with a 'Filter' and 'Clear filter' button) and one for 'List' (with a 'List' tab and a 'Filter' button). The 'List' table has columns for Update, Discard, Item number, Host, Protocol, Port, execution eng, Last update date/time, and Last updated by. A single row is shown with values: 1, exastro-it-automation-v1-3-0-en, https, 443, Ansible, 2020/01/28 10:59:30, and System Administrator. At the bottom of the interface, there's a 'Contact administrator' link.

Figure 5.2-1 Submenu screen (interface information)

- (2) Click the "List" - "Update" button to register the interface information.

The registration screen has six input fields: Host*, Protocol*, Port*, execution engine*, Data relay storage path (ITA)*, and Data relay storage path. The values entered are: exastro-it-automation, https, 443, Ansible, /exastro/data_re, and /exastro/data_re. There are navigation arrows at the bottom of the form.

Figure 5.2-2 Registration screen (Interface information)

- (3) The item list of interface information screen is as follows.

If operation is executed while interface information not registered or multiple information is registered, **unexpected errors may occur**.

Table 5.2-1 Registration screen item list (Interface information)

Item	Description	Input required	Input type	Restrictions
Execution engine	Select the execution engine between Ansible and Ansible Automation Controller. When Ansible Automation Controller is selected, in order to execute ansible-vault command, Ansible Engine interface is also needed.	<input type="radio"/>	List	
Ansible Core interface	Host	Enter the host name (or IP address) of Ansible server. It is recommended to enter host name when using HTTPS communication.	<input type="radio"/>	Manual Maximum length 128 bytes The user must configure Ansible Core interface settings, even if "Ansible Core" is not set as the execution engine.
	Protocol	Enter either http/https as the protocol with Ansible, Ansible Automation Controller server.	<input type="radio"/>	Manual - The user must configure Ansible Core interface settings, even if "Ansible Core" is not set as the execution engine.
	Port	Enter the connection port (80/443) of Ansible, Ansible Automation Controller. The port is usually HTTPS (443).	<input type="radio"/>	Manual - The user must configure Ansible Core interface settings, even if "Ansible Core" is not set as the execution engine.
	Execution user	Enter the execution user to execute ansible-playbook/ansible-vault command with sudo.	-	Manual Maximum length 64 bytes The user must configure Ansible Core interface settings, even if "Ansible Core" is not set as the execution engine.
	ACCESS_KEY_ID	Enter the access key used for authentication when connecting to the Ansible server.	-	Manual Maximum length 64 bytes The user must configure Ansible Core interface

Item		Description	Input required	Input type	Restrictions
Ansible Automation Controller interface	SECRET_ACCE SS_KEY	Enter the secret access key used for authentication when connecting to the Ansible server.	-	Manual	settings, even if "Ansible Core" is not set as the execution engine. Maximum length 64 bytes The user must configure Ansible Core interface settings, even if "Ansible Core" is not set as the execution engine.
	Host	Select Ansible Automation Controller that will connect to ITA. You can select from the list of hosts that are registered in the Ansible Automation Controller Host list.	<input type="radio"/>	Manual	Maximum length 128 bytes Required
	Protocol	Enter either http/https as the protocol with Ansible, Ansible Automation Controller server.	<input type="radio"/>	Manual	Required when the execution engine is Ansible
	Port	Enter the connection port (80/443) of Ansible, Ansible Automation Controller. The port is usually HTTPS (443).	<input type="radio"/>	Manual	Required if "Ansible Core" is not set as execution engine.
	Organization name	Enter the organization name registered in Ansible Automation Controller.	-	List	Required if "Ansible Core" is not set as execution engine.
	Authentication token	Enter the user authentication token when connecting Ansible Automation Controller server from ITA.	-	Manual	Maximum length 128 bytes. Required if "Ansible Core" is not set as execution engine.
	Delete runtime data	Select whether to delete the data automatically generated by Ansible Automation Controller during operation execution after operation is done. Select "Delete" from the pulldown list to delete.	-	List	Required if "Ansible Core" is not set as execution engine.
SCM List Git link	Host name	A Git repository linked with the Ansible Automation Controller will be created on the host where the Ansible driver backyard feature is installed.		Manual	Maximum length 128 bytes Required if "Ansible Automation Controller" is set as execution engine.
	<input type="checkbox"/> User	Input the user needed in order to connect to the		Manual	Maximum length 128

Item		Description	Input required	Input type	Restrictions	
		Git repository with ssh protocol from the Ansible Automation Controller.			bytes Required if "Ansible Automation Controller" is set as execution engine.	
		Ssh secret key file	Upload the secret key file needed in order to connect to the Git repository with ssh protocol from the Ansible Automation Controller.	File	File selection. Maximum file size: 4GB Required if "Ansible Automation Controller" is set as execution engine.	
		Passphrase	Input the passphrase for the ssh secret key file.	Manual	Maximum length 256 bytes	
Data relay storage path (ITA) ※1		Enter the directory viewed from the ITA system / Ansible driver server.	<input checked="" type="radio"/>	Manual	Maximum length 128 bytes	
Data relay storage path (Ansible/Ansible Automation Controller)		Enter the directory viewed from the Ansible RestAPI and Ansible Automation Controller servers.	<input checked="" type="radio"/>	Manual	Maximum length 128 bytes	
Symphony instance data relay storage path (Ansible/Ansible Automation Controller)		Enter the directory which shares the shared directory between each movement when executing Symphony with Ansible RestAPI, Ansible Automation Controller server. The path viewed from the ITA system is set from the Symphony interface information menu. Please refer to the "User instruction manual_ITA basic console" for the Symphony interface information.	<input checked="" type="radio"/>	Manual	Maximum length 128 bytes	
Conductor instance data relay storage path (Ansible/Ansible Automation Controller)		When executing Conductor, enter the directory shared by each Movement. The path viewed from the ITA system is set from the Conductor interface information menu. For the Conductor interface information, please refer to "ITA User_Instruction_Manual_Conductor".	<input checked="" type="radio"/>	Manual	Maximum length 128 bytes	
Optional parameter		Enter the Movement-common optional parameter of Ansible-Playbook command. Movement-specific optional parameters are entered in the Movement list menu. In the case that the execution engine is Ansible: Enter the optional parameter of Ansible-Playbook command. The -i option is set by ITA.	-	Manual	Maximum length 512 bytes	

Item	Description	Input required	Input type	Restrictions
	<p>In the case that the execution engine is Ansible Automation Controller:</p> <p>The following option parameters can be set:</p> <ul style="list-style-type: none"> -verbosity -f FORKS,--forks=FORKS -l SUBSET,--limit=SUBSET -e EXTRA_VARS,--extra-vars=EXTRA_VARS <p>EXTRA_VARS: Variable name=specific value Variable name=specific value.....</p> <ul style="list-style-type: none"> -t TAGS,--tags=TAGS -b,--become -D,--diff --skip-tags=SKIP_TAGS --start-at-task=START_AT_TASK <p>The original optional parameters of Ansible Automation Controller are as follows.</p> <ul style="list-style-type: none"> -ufc,--use_fact_cache use fact cache -as,--allow_simultaneous enable simultaneous job execution -jsc,--job_slice_count= job slice count <p>For the original optional parameters of Ansible Automation Controller, please refer to the description of job template in the Ansible Automation Controller user guide.</p>			
Number of parallel executions	Enter the maximum numbers of Movement (Legacy/Pioneer/Legacy-Role) that can be executed at the same time.	<input type="radio"/>	Manual	
Status monitoring cycle(milliseconds)	<p>Enter the refresh interval of the log displayed in "5.3.12 Check operation status".</p> <p>Usually the value around 3000 milliseconds is recommended.</p>	<input type="radio"/>	Manual	Minimum value 1000 milliseconds
Number of rows to display progress status	<p>Enter the maximum display line count of the execution log, errorlog in "5.3.12 Check operation status".</p> <p>Usually the value around 1000 lines is recommended.</p>	<input type="radio"/>	Manual	-
NULL link	<p>Set whether to register NULL (blank) value to substitution value list menu if the specific value in parameter sheet is NULL (blank) in the substitution value auto-registration setting menu.</p> <p>This value will be applied when "NULL link" in the substitution value auto-registration setting menu is blank.</p>	<input type="radio"/>	List	-

Item	Description	Input required	Input type	Restrictions
	<ul style="list-style-type: none"> If the "Valid" is set, any value in the parameter sheet will be registered in the substitution value list menu. (NULL value will be registered) If the "Invalid" is set, only specific value in the parameter sheet will be registered in the substitution value list menu (NULL value will not be registered) 			
Remarks	Free description field	-	Manual	Maximum length 4000 bytes

※1 Data relay storage paths also supports configurations where ITA and Ansible operate on separate servers, so the directory paths will be managed separately.

For more information, please refer to the "Exastro-

[ITA_System_Configuration_Enviroment_Construcion_Guide_Ansible-driver](#)" document.

5.2.2 Ansible Automation Controller host list

In [Ansible Automation Controller Host List], register/update/abolish the information required to execute Rest API of Ansible Automation Controller and the information required to transfer the construction materials to Ansible Automation Controller.

If Ansible Automation Controller is built in a cluster configuration, it is necessary to register all host information in the cluster. The user will not have to register Ansible Automation Controller's hop node.

Figure 5.2-3 Submenu screen (Ansible Automation Controller host list)

- (1) Click the "List"- "Update" button to register the Ansible Automation Controller host information.

Figure 5.2-4 Registration screen (Ansible Automation Controller host)

- (2) The list of items on the Ansible Automation Controller host list screen is as follows.

Table 5.2-2 Registration screen item list (Ansible Automation Controller host list)

Item	Description	Input required	Input type	Restrictions
Host	Enter the host name (or IP address) of the Ansible	<input type="radio"/>	Manual	Maximum length

Item	Description		Input required	Input type	Restrictions
	Automation Controller server. For HTTPS communication, the host name is recommended.			input	128 bytes
Authentication method	Select the authentication method used when connecting from Ansible/Ansible Automation Controller to the target device. ● Password Authentication If you also choose ● for Login password Management, you will be required to input a login password. ● Key Authentication (No passphrase) You must upload an SSH secret key file (id_ras). ● Key Authentication (With passphrase) You must upload an SSH secret key file (id_ras) and input a passphrase. ● Key Authentication (Key Exchanged)※1 You will not be required to upload an SSH secret key file (id_ras)		<input type="radio"/>	Manual input	Maximum length 30 bytes
Login user	Enter the login user for connecting to the Ansible Automation Controller server via file transfer (scp). Set and use a password for the login user and the awx user generated when installing Ansible Automation Controller.		<input type="radio"/>	Manual input	Maximum length 30 bytes
Password	This is required when password authentication is selected as the authentication method. Specify the password of the login user.		—	Manual input	Maximum length 30 bytes
Ssh key authentication information	ssh authentication key file	When key authentication is selected in the authentication method, enter the file for key authentication by specifying the ssh authentication key file.		File	Maximum size 4gb
	Passphrase	If passphrase is set to the secret key file, input the passphrase.		Manual input	Maximum 256 bytes
isolated Tower		Select [●] for isolated Tower when it is built in a cluster configuration.		Select	
Remarks		Free description field.		Manual input	Maximum length 4000 bytes

※1 Distribution of the public key file required when the authentication method is Key authentication (key exchanged)

With ssh, connect to the root of the server where ITA is installed to Ansible Automation Controller's awx user.

Copy the root's public key to the Ansible Automation Controller's AWX User's authorized keys.

5.2.3 Global variable list

- (1) In the "Global variable list" menu, register/update/discard the global variable name used in Playbook, dialog files, etc.

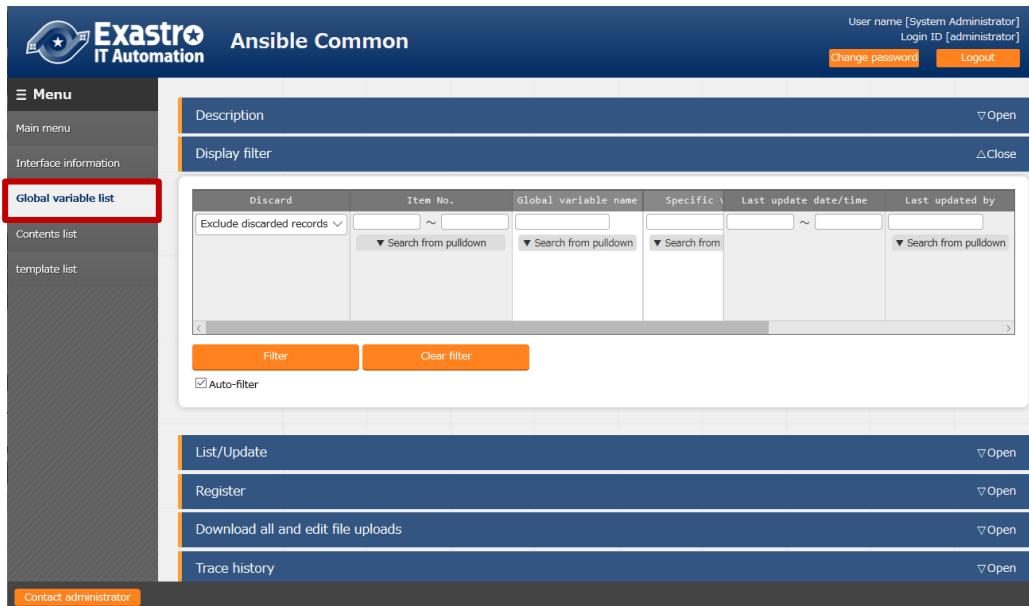


Figure 5.2-5 Submenu screen (Global variable list)

- (2) Click the "Register" - "Start Registration" button to register the operation information.

The screenshot shows a registration form titled 'Register'. The form has a table with columns: 'Item No.', 'Global variable name*', 'Specific value*', 'Variable name description', and 'Remarks'. The 'Item No.' column has a note 'Auto-input'. The 'Global variable name*' and 'Specific value*' columns have red asterisks indicating they are required. Below the table is a note: '※* is a required item.' At the bottom are two buttons: 'Back' and 'Register'.

Figure 5.2-6 Registration screen (Global variable list)

(3) The item list of global variable list screen is as follows.

Table 5.2-3 Registration screen item list (Global variable list)

Item	Description	Input required	Input type	Restrictions
Global variable name	Enter the variable name. Enter the variable name in the "GBL_****" format. Half-width alphanumeric character and underscore (_) can be used. (Minimum length: 1 byte, maximum length: 128 bytes)	<input checked="" type="radio"/>	Manual	As written in the description column.
specific value	Enter the specific value File embedded variable "CPF_" and template embedded variable "TPF_" can be entered in the specific value column. When describing the variables, enclose the variable names with {{}} as describing the variables in the Playbook. e.g.) Entering TPF_sample for specific value '{{△TPF_sample△}}' △: half-width space ': recommended	<input checked="" type="radio"/>	Manual	Maximum length 8192 bytes
Variable name description	Enter the description or comment of the variable.	-	Manual	Maximum length 256 bytes
Remarks	free description field	-	Manual	Maximum length 4000 bytes

5.2.4 Template list

- (1) In the "template list" menu, register/update/discard the Jinja2 template file and the template embedded variable used in the parameter of template module and ios_config module, etc. defined in the Playbook.

If template module is registered in the template list, the template file used in the template module etc. defined in the playbook can be specified by template embedded variable.

Figure 5.2-7 Submenu screen (Template list)

- (2) Click the "Register" - "Start Registration" button to register the file management information.

Template ID	Template embedded variable name*	Template files*	Variable definition
Auto-input	<input type="text"/>	<input type="button" value="Choose File"/> No file chosen <input type="button" value="Upload in advance"/> <small>Upload status:</small>	<input type="text"/>

Figure 5.2-8 Registration screen (Template list)

(3) The items of registration screen are as follows.

Table 5.2-4 Registration screen item list (template list)

Item	Description	Input required	Input type	Restrictions
Template embedded variable name	Enter the variable name embedded in parameters such as template module or ios_config module, etc. Enter the variable name in the "TPF_****" format. Half-width alphanumeric character and underscore (_) can be used.(Minimum length: 1 byte, maximum length: 128 bytes)	<input type="radio"/>	Manual	As writed in the description column.
Template files	Upload the Jinja2 template file used as the parameter of module.	<input type="radio"/>	File	Text format Maximum size 4GB
Variable definition	Define the variable used in the template file. If the template is used only in Ansible-Role and the variable is defined in the default variable definition file, then the variable definition column can be omitted. If the template is used only in Ansible-Role and the variable is defined in the default variable definition file, then the variable definition column can be omitted. If the variable with same name is used in multiple template, the variable definitions have to match. Error will occur during registration if the variable definitions do not match. Although the variable definition is based on the specification of Ansible, there is own specification of ITA.The notes of variable definition is writed in 5.2-5-1.	-	Manual	Maximum length 4000 bytes
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

Please "Upload in advance (①)" the "template files" before "register".

Please click the "Register" button after checking the Playbook file name displayed in the "Upload status(②)".

The screenshot shows a registration interface. At the top, there is a label 'Template files*' followed by an input field containing 'Choose File sample.yml'. Below this is an orange button labeled 'Upload in advance' with a circled '①' to its right. Further down, there is a red-bordered box containing the text 'Upload status: Uploaded. File name sample.yml Size128bytes' with a circled '②' to its right.

Table 5.2-5 Notes of variable definition

Type	Notes
Normal variable	Specific value is optional. e.g.) VAR_sample_1: none VAR_sample_2:
Multiple specific value variable	Specific value is optional. e.g.) VAR_sample_1: △- none VAR_sample_2: [] Please enter 1 or more half-width space(△) before - when defining specific value. The variable definition maybe misinterpreted.
Multistage variable	It is possible to define hierarchical variable structures e.g.) VAR_sample_1: - item1: none item2: VAR_sample_2: - array: - item1: none item2: The template with nested variable defined can only be used in Ansible-Role When used in Ansible-Role, if the variable with same name is defined in default variable definition file, etc., the definition of the variables have to match. If the definition of the variables do not match, an error will occur during registration.
Global variable	The definition of specific value is optional e.g.) GBL_sample_1: none GBL_sample_2:
ITA original variable	The definition of variable is not required.
substitution variable	The 3 kinds of variable that can be defined are as follows. • Normal variable • Multiple specific value variable • Nested variable The note of each variable definition are the same. e.g.) LCA_sample_1: LCA_sample_2: [] LCA_sample_3: - item1: none item2: The template with substitution variable defined can only be used in Ansible-Role

For details, Please refer to the attachment "User Instruction Manual - Ansible-driver attachment- Ansible usage guideline with additional rules"

① Write Playbook

When describing the template registered in template list menu in Playbook, write the appropriate parameter in the template embedded variable name.

If the template embedded variable name is not used, write the variable registered in the substitution list and the path of the file.

e.g.)

Write Playbook

Registration content

- template: src='{{△TPF_hosts△}}' dest=/etc/hosts
△: half-width space

Template embedded variable name	Template file
TPF_hosts	/etc/hosts

Please write the file name in _dest. If the file name is not specified, the work will be executed with the registered template file whose file name is added with the ITA management number in the front of the file name.

For example, in the case of dest=/etc/, the file name will be /etc/10-digit-number_hosts

② Write dialog file

In the case of describing the dialog file, write the template embedded variable name.

e.g.)

Write dialog file

- expect: '{{△_loginuser_△}}@{{△_loginhostname_△}}'
exec: 'scp △ITA user@ITA host name:{{△TPF_hosts△}}△forwarding destination'
- expect: 'password:'
exec: 'ITAuser password'

△: half-width space

Registration content

Template embedded variable name	Template file
TPF_hosts	/etc/hosts

Please write the file name in the forwarding destination.

If the file name is not specified, the work will be executed with the registered template file whose file name is added with the ITA management number in the front of the file name.

For example, in the case of forwarding destination=/etc/, the file name will be /etc/10-digit-number_hosts

△TPF_hosts△ will be replaced by the absolute path during execution.

By reading the variable definition of template with internal process, it is possible to register specific value in menu "5.3.9 Substitution value auto-registration setting" and menu "5.3.11 Substitution value list".

Since the timing of file reading is not in real time, it may take some time^{※1} until the variables can be handled in menu "5.3.9 Substitution value auto-registration setting" and menu "5.3.11 Substitution value list".

※1 The timing of file reading is writed in " 7.2 About the maintenance method", so please refer to it.

5.2.5 File list

- (1) In the "contents list" menu, register/update/discard the file and file embedded variable used in each module defined in the Playbook.

If the files are registered in the contents list, the file used in each module defined in the Playbook can be specified by file embedded variable.

Figure 5.2-9 submenu screen (Contents list)

- (2) Click the "Register" - "Start Registration" button to register the file management information.

File ID	File embedded variable name*	Files*	Remarks	Last update
Auto-input	<input type="text"/>	<input type="button" value="Choose File"/> No file chosen <input type="button" value="Upload in advance"/> <small>Upload status:</small>	<input type="text"/>	Auto-input

Figure 5.2-10 Registration screen (Contents list)

(3) The items of registration screen are as follows.

Table 5.2-6 Registration screen item list (contents list)

Item	Description	Input required	Input type	Restrictions
File embedded variable name	Enter the variable names to be embedded in the parameter of each module. Enter the variable name in the "CPF_****" format. Half-width alphanumeric character and underscore(_) can be used.(Minimum length: 1 byte, maximum length: 128 bytes)	○	Manual	As written in the description column
Files	Upload the file used in each module.	○	File	Maximum size 4GB
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

Please "Upload in advance (①)" the "template files" before "register".

Please click the "Register" button after checking the Playbook file name displayed in the "Upload status(②)".

Template files*

Choose File sample.yml

Upload in advance ①

Upload status:
Uploaded.
File name sample.yml
Size128bytes ②

① Write Playbook

When describing each modules in the Playbook, write the file embedded variable name.

e.g)

Playbook

Registration content

```
-copy: src='{{△CPF_hosts△}}' dest=/etc/hosts  
△: half-width space
```

File embedded variable name	Files
CPF_hosts	hosts

Please write the file name for dest. If the file name is not specified, the work will be executed with the registered file whose file name is added with the ITA management number in the front of the file name.

For example, in the case of dest=/etc/, the file name will be /etc/10-digit-number_hosts

```
-unarchive src='{{△CPF_tool_tgz△}}' dest=/usr/local/bin  
△: half-width space
```

File embedded variable name	Files
CPF_tool_tgz	tool.tgz

② Write dialog file

In the case of describing the dialog file, write the file embedded variable name.

e.g)

Dialog file

Registration content

```
- expect: '{{△__loginuser_△}}@{{△__loginhostname_△}}'  
  exec: 'scp △ITA user@ITA host name:{{△CPF_hosts△}}△forwarding destination'  
- expect: 'password:'  
  exec: 'exec: ITAuser password'  
  △:half-width space
```

File embedded variable name	Files
CPF_hosts	hosts

Please write the file name in the forwarding destination.

If the file name is not specified, the work will be executed with the registered file whose file name is added with the ITA management number in the front of the file name.

For example, in the case of forwarding destination=/etc/, the file name will be /etc/10-digit-number_hosts

`△CPF_hosts△` will be replaced by the absolute path of forwarding origin during execution.

5.2.6 Collection interface information

In [Collection Interface Information], in order to use the standard RESTAPI of ITA used in the collect function, the connection interface information for RESTAPI access is updated.

For details, please refer to the "Exastro-ITA_User_Instruction_Manual_Collect Function".

5.2.7 Collection item value list

In [Collection item value list], the item to be collected is linked to the item of the parameter sheet.

For details, please refer to the "Exastro-ITA_User_Instruction_Manual_Collect Function".

(1) Clicking the Menu name or the List/Update Menu ID will move the user to that selected menu.

収集項目(FROM)				パラメータシート(TO)				アクセス権	備考	最終更新日時	最終更新者					
履歴	更新	廃止	ID	ベース形式	PREFIX(ファイル名)	変数名	メンバ変数	ID	名前	ID	名前	項目	アクセス許可ロール	備考	最終更新日時	最終更新者
履歴	更新	廃止	1 YAML	prefix	var			2100011611	代入値自動登録用	2 test	パラメータ/項目 1				2021/04/06 16:21:54	システム管理者

フィルタ結果件数: 1

Excel出力

Figure 5.2-11 Submenu screen (Collected item value list)

5.3 Ansible-Legacy／Legacy Role／Pioneer console

The operation of Ansible-Legacy／Legacy Role／Pioneer console.

5.3.1 OS type master

- On the [OS Type master] screen, the OS type of the device to be operated is managed from the ITA Pioneer.
※This menu exists only in the Ansible-Pioneer console.

OS type ID	OS type name*	Device type			Remarks	Last update date/time	Last updated by
		SV	NM	ST			
Auto-input						Auto-input	Auto-input

* * is a required item.

Back Register

Download all and edit file uploads ▽Open

Trace history ▽Open

Figure 5.3-1 Submenu screen (OS type master)

- Click the "Register"- "Start Registration" button to register the OS information.

OS type ID	OS type name*	Device type			Remarks	Last update date/time	Last updated by
		SV	NW	ST			
1						Auto-input	Auto-input

Figure 5.3-2 registration screen (OS type master)

(3) Clicking the Dialogue file material collection button will move the user to the target 5.3.6 Dialogue file collection.

List/Update								
History	Update	Discard	OS type ID	OS type name	Device type	Dialogue file material collection	Access permission	Remarks
History	Update	Discard	1	Test OS	SV	NW	ST	Role to allow access
						Dialogue file material collection		

Figure 5.3-3 Sub menu screen (OS Type master)

(4) The list of items on the registration screen is as follows.

Table 5.3-2 registration screen item list (OS type master)

Item		Description		Input required	Input type	Restrictions
OS type ID		A unique ID that identifies the registration information will be automatically entered.		○	Auto	-
OS type name		Enter any device name.		○	Manual	Maximum length 256 bytes
Model	SV	Select "●" if the equipment type is a server.		-	List	-
	NW	Select "●" if the device type is network device.		-	List	-
	ST	Select "●" if the device type is storage device.		-	List	-
Remarks		Free description field.		-	Manual	

5.3.2 Movement list

- (1) Register/Update/Discard Movement name in "Movement list"

Figure 5.3-4 submenu screen (Movement list)

- (2) Click the "Register" - "Start Registration" button to register the Movement information.

Movement ID	Movement Name*	Delay timer	Dedicated information for ansible		
Auto-input			Host specific format*	WinRM connection	Header section
<					
>					

Figure 5.3-5 Registration screen (Movement list)

- (3) Clicking the Movement-Playbook link (Movement-Dialogue type link, Movement - Role link) button will move the user to the target 5.3.7 Movement-Playbook link (Movement-Dialogue type link, Movement - Role link).

Figure 5.3-6 Submenu screen (Movement list)

(4)The list of registration screen items are as follows.

Table 5.3-2 Registration screen item list (Movement list)

Item	Description	Input required	Input type	Restrictions
Movement name	Enter the name of Movement	<input type="radio"/>	Manual	Maximum length 256 bytes
Delay timer	Enter the specified period (1~) if you want the warning of delay status to display when the scheduled time of Movement has delayed. (Unit:minute) The warning will not display if the column is not entered.	-	Manual	-
Host specific format	Select "Host name" if the user wants to specify the host that is not represented by an IP address. Normally IP is recommended	<input type="radio"/>	List	-
Number of parallel executions ※ Only displayed in the Pioneer Movement list	Enter the number of target hosts that Ansible can execute simultaneously. ■ About the behavior when the column is not entered <ul style="list-style-type: none"> • In the case of Ansible driver, the content of configuration file(/etc/ansible.conf) in the server will be the default values. • In the case of Ansible Automation Controller driver, the default value of Ansible Automation Controller will be used. 	-	Manual	NULL or Integer
WinRM connection	Select "●" if the target host is WindowsServer.	-	List	-
Header section ※ Not displayed in the Pioneer Movement list	Edit the parent Playbook automatically generated by ITA from the beginning to the tasks or roles section. The following will be applied if the column is not entered. Ansible: <pre>- hosts: all remote_user: "{{ __loginuser__ }}" gather_facts: no become: yes</pre> Ansible Automation Controller: <pre>- hosts: all gather_facts: no become: yes</pre> ※ In case of connecting with winrm, become:yes can't be applied.	-	Manual	Maximum length 512 bytes
Optional parameter (Not displayed in the Pioneer)	Enter the Movement-specific optional parameter of Ansible-Playbook command. In the case that the execution engine is Ansible:	-	Manual	Maximum length 256 bytes

Movement list)	<p>Enter the optional parameter of Ansible-Playbook command.</p> <p>The -i option is set by ITA</p> <p>In the case that the execution engine is Ansible Automation Controller:</p> <p>The following option operator can be set</p> <ul style="list-style-type: none"> -verbosity -f FORKS,--forks=FORKS -l SUBSET,--limit=SUBSET -e EXTRA_VARS,--extra-vars=EXTRA_VARS <p>EXTRA_VARS: Variable name=specific value Variable name=specific value.</p> <ul style="list-style-type: none"> -t TAGS,--tags=TAGS -b,--become -D,--diff --skip-tags=SKIP_TAGS --start-at-task=START_AT_TASK <p>The original optional parameters of Ansible Automation Controller are as follows.</p> <ul style="list-style-type: none"> -ufc,--use_fact_cache use fact cache -as,--allow_simultaneous enable simultaneous job execution -jsc,--job_slice_count= job slice count <p>For the original optional parameters of Ansible Automation Controller, please refer to the description of job template in the Ansible Automation Controller user guide.</p>			
Virtualenv ※Displayed when the execution engine is Ansible Automation Controller	Select the Ansible execution environment where virtualenv is constructed. The Ansible execution environment used when installing Tower will be used if this column is not selected.	-	List	
Remarks	free description field	-	Manual	Maximum length 4000 bytes

【Notes】

In the case of selecting "●" in the WinRM connection column, all connection hosts will be considered as WindowsServer.

5.3.3 Playbook file list (Ansible-Legacy only)

- (1) Register/update/discard the Playbooks created by users in the "Playbook files" menu.
 ※This menu only exists in the Ansible-Legacy console.
 Please refer to "[6.1 Write Playbook \(Ansible-Legacy\)](#)" about describing Playbook.

Figure 5.3-7 Submenu screen (Playbook files)

- (2) Click the "Register" - "Start Registration" button to register the Playbook.

Figure 5.3-8 Registration screen (Playbook files)

- (3) Clicking the Movement-Playbook link (Movement-Dialogue type link, Movement - Role link) button will move the user to the target 5.3.7 Movement-Playbook link (Movement-Dialogue type link, Movement - Role link).

List/Update							
History	Update	Discard	Playbook ID	Playbook name	Playbook files	Movement playbook link	Access permission
History	Update	Discard		1 Sample1	Sample1.yml	Movement playbook link	Role to allow access

Filter result count: 1

Figure 5.3-9 Submenu screen (Playbook files)

(4) The list of registration screen items are as follows.

Table 5.3-3 Registration screen item list (Playbook files)

Item	Description	Input required	Input type	Restrictions
Playbook name	Enter the Playbook name to be managed in ITA.	<input type="radio"/>	Manual	Maximum length 256 bytes
Playbook files	Upload the created Playbook file. Please make sure that the playbook file is created with UTF-8 Code when uploading it. Playbook files other than those with a character code of UTF-8 and without BOM will get an error in uploading.	<input type="radio"/>	File	Maximum size 4GB
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

Please "Upload in advance (①)" the "Playbook files" before "register". Please click the "Register" button after checking the Playbook file name displayed in the "Upload status(②)".

The internal process will extract the variables defined in Playbook files. Users can register specific value of the extracted variables in menu "[5.3.9 Substitution value auto-registration setting](#)" and menu "[5.3.11 Substitution value list](#)".

Since the timing of extraction is not in real time, it may take some time^{※1} until the variables can be handled in menu "[5.3.9 Substitution value auto-registration setting](#)" and menu "[5.3.11 Substitution value list](#)".

※1 The timing of extraction is written in "[7.2 About the maintenance method](#)", so please refer to it.

5.3.4 Role package list (Ansible-Legacy Role only)

(1) Register/upload/discard the role package file created by the users.

※This menu only exists in Ansible-Legacy Role console.

Please compress the directory of the hierarchy level which contains "roles" into zip file and register role package file with the zip file.

Please refer to "[6.3 Write role package \(Ansible-Legacy Role\)](#)" for the structure of role package directory.

The screenshot shows the Ansible-LegacyRole interface. On the left, a sidebar menu lists various options: Main menu, Movement list, Role package list (which is highlighted with a red box), Movement details, Nested variable maximum iteration count list, Substitution value auto-registration setting, Target host, Substitution value list, Execution, Check operation status, Execution list, and Contact administrator. The main content area has three tabs: Description (with a ▽Open button), Display filter (with a △Close button), and List/Update. The List/Update tab contains a message: "Record does not exist. New registration can be done as per the following." Below this are three buttons: Register (with a ▽Open button), Download all and edit file uploads (with a ▽Open button), and Trace history (with a ▽Open button). At the top right, there are buttons for User name [System Administrator], Login ID [administrator], Change password, and Logout.

Figure 5.3-10 Submenu screen (Role package list)

(2) Click the "Register" - "Start Registration" button to register the Playbook.

The registration screen is titled "Register". It has fields for Item No. (Auto-input), Role package name* (a required item), Role package file (ZIP format) (with a "Choose File" button and "Upload in advance" button), Remarks (a large text area), and Last update date/time (Auto-input). At the bottom are "Back" and "Register" buttons.

Figure 5.3-8 Registration screen (Role package list)

(3) Clicking the Movement-role link (Movement-Dialogue type link, Movement - Role link) button will move the user to the target 5.3.7 Movement-Playbook link (Movement-Dialogue type link, Movement - Role link).

Figure 5.3-12 Submenu screen (Role package list)

(4) The list of registration screen items are as follows.

Table 5.3-4 Registration screen item list (Role package list)

Item	Description	Input required	Input type	Restrictions
Role package name	Enter the role package name to be managed in ITA.	<input type="radio"/>	Manual	Maximum length 256 bytes
Role package file	Upload the created role package file (zip format). Please make sure that the role package file is created with UTF-8 Code and without BOM when uploading it. If a playbook file other than UTF-8 without BOM is included, an error will occur during registration. For details, please refer to the 6.3 Role package (Ansible-Legacy Role).	<input type="radio"/>	File	Maximum size 20M bytes
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

Please "Upload in advance (①)" the "Role package file" before "register".

Please click the "Register" button after checking the role package file name displayed in the "Upload status (②)".

The internal process will extract the variables defined in Role package files. Users can register specific value of the extracted variables in menu "5.3.9 Substitution value auto-registration setting" and menu "5.3.11 Substitution value list".

Since the timing of extraction is not in real time, it may take some time^{※1} until the variables can be handled in menu "5.3.9 Substitution value auto-registration setting" and menu "5.3.11 Substitution value list".

※1 The timing of extraction is written in "7.2 About the maintenance method", so please refer to it.

5.3.5 Dialog type list (Ansible-Pioneer only)

- (1) Register/update/discard dialog type in the "dialog type list" menu

This menu only exists in the Ansible-Pioneer console

Ansible-Pioneer defines the differences for each OS type in each dialog file, and combines the same purpose dialog file as dialog type to remove (abstract) the device difference.

Figure 5.3-13 Submenu screen (dialog type list)

- (2) Click the "Register" - "Start Registration" button to register the operation information.

Item No.	Dialog type name*	Remarks	Last update date/time	Last updated by
Auto-input			Auto-input	Auto-input

Figure 5.3-14 Registration screen (Dialog type list)

- (3) Clicking the Movement-dialogue type link (Movement-Dialogue type link, Movement - Role link) button will move the user to the target 5.3.7 Movement-Playbook link (Movement-Dialogue type link, Movement - Role link). Clicking the Dialogue file material collection button will move the user to the target 5.3.6 Dialogue file material collection.



Figure 5.3-15 Submenu screen (Dialogue type list)

- (4) The list of registration screen items are as follows.

Table 5.3-5 Registration screen item list (Dialog type list)

Item	Description	Input required	Input type	Restrictions
Dialog type name	Enter the name of dialog type	<input type="radio"/>	Manual	Maximum length 256 bytes
Remarks	Free description field	-	Manual	Maximum length 4000 bytes

5.3.6 Dialog files (Ansible-Pioneer only)

- (1) Register/update/discard the dialog file created by users in "dialog files" menu.
※This menu only exists in Ansible-Pioneer console.
- (2) Please refer to "[6.2 Write Dialog file \(Ansible-Pioneer\)](#)" for describing the dialog file, etc.
Register dialog files for each combination of dialog type and OS type.
Please register dialog file of each "OS type" with the same "dialog type" in the case of supporting multiple OS types with one "dialog type".

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

Description ▽Open

Display filter △Close

Discard	Dialog ID	Dialog type	Last update date/time	Last updated by
Exclude discarded records	~	▼ Search from pulldown	~	▼ Search from pulldown

Filter Clear filter Auto-filter

List/Update △Close

Record does not exist.
New registration can be done as per the following.

Register △Close

Dialog ID	Dialog type	OS type	Dialog file	Last update date/time	Last updated by
Auto-input	▼	▼	Choose File No file chosen Upload in advance <small>Upload status:</small>	Auto-Input	Auto-Input

* is a required item.

Back Register

Download all and edit file uploads ▽Open

Trace history ▽Open

Contact administrator

Figure 5.3-16 Submenu screen (dialog files)

- (3) Click the "Register" - "Start Registration" button to register the dialog files.

Dialog ID Dialog type* OS type* Dialog file Remarks

Auto-input ▼

Choose File No file chosen
Upload in advance
Upload status:

* is a required item.

Figure 5.3-17 Registration screen (dialog files)

- (4) Clicking the list/update's Dialog type will move the user to the 5.3.5 dialog type list.
 Clicking the OS Type will move the user to the 5.3.1 OS Type master.

List/Update							
History	Update	Discard	Dialog ID	Dialog type	OS type	Dialog file	Access permission
							Role to allow access
History	Update	Discard	1	Test	Test OS	top.yml	

Figure 5.3-18 Submenu screen (Dialog files)

- (5) The list of registration screen items are as follows.

Table 5.3-6Registration screen item list (Dialog files)

Item	Description	Input required	Input type	Restrictions
Dialog type	The dialog type registered in the dialog type list menu will be displayed. Select the dialog type of dialog file to be registered.	<input type="radio"/>	List	-
OS type	The OS type registered in the OS type master menu will be displayed. Select the OS type of dialog file to be registered.	<input type="radio"/>	List	-
Dialog file	Upload the dialog file according to the dialog type and OS type. Please make sure that the file is created with UTF-8 Code and without BOM when uploading it. An error will occur if the character code is anything else.	<input type="radio"/>	File	Maximum size 4gb
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

Please "Upload in advance (①)" the "dialog file" before "register".

Please click the "Register" button after checking the dialog file name displayed in the "Upload status(②)".

The screenshot shows a form for uploading a dialog file. At the top, there is a label 'Dialog file*' followed by a 'Choose File' button with the path 'test.txt'. Below this is an orange button labeled 'Upload in advance' with a red circle ① above it. Further down is a box labeled 'Upload status:' containing the text 'Uploaded. File name test.txt Size989bytes' with a red box and circle ② around it.

The internal process will extract the variables defined in dialog files. Users can register specific value of the extracted variables in menu "[5.3.9 Substitution value auto-registration setting](#)" and menu "[5.3.11 Substitution value list](#)".

Since the timing of extraction is not in real time, it may take some time※1 until the variables can be handled in menu "[5.3.9 Substitution value auto-registration setting](#)" and menu "[5.3.11 Substitution value list](#)".

※1 The timing of extraction is written in "7.2 About the maintenance method", so please refer to it.

5.3.7 Movement details

- (1) Register/update/discard the files executed in the Movement in "Movement details" menu.

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

Description ▽Open
Display filter △Close

Discard	Associated item No.	Movement	Playbook files	Last update date/time	Last updated by
Exclude discarded records ▼	~	▼ Search from pulldown			
Filter					
Clear filter					
<input checked="" type="checkbox"/> Auto-filter					

List/Update △Close

Update	Discard	Associated item No.	Movement	Playbook files	Include order	Remark	Last update date/time	Last updated by
Update	Discard	1:testing-one	create_dir		1		2020/02/10 11:44:53	System Administrator

Filter result count: 1

Output Excel

Register ▽Open
Download all and edit file uploads ▽Open
Trace history ▽Open

Contact administrator

Figure 5.3-19 Submenu screen (Movement details)

※The screen is from Ansible Legacy.

- (2) Click the "Register" - "Start Registration" button to register the details of Movement.

Associated item No. Movement Playbook files Include order Remarks

Auto-input

1:testing-one

※ * is a required item.

Table setting

Back Register

Figure 5.3-20 Registration screen (Movement details)

- (3) Clicking the Movement button will move the user to the target 5.3.2 Movement list.
Clicking the Playbook file button will move the user to the target 5.3.3 playbook files.
※For Movement-dialog type link (Ansible-Pioneer), it will be Movement and Dialoge type.
For Movement-Role link (Ansible-Legacy role), it will be Movement and Role package name link.

List/Update							
History	Update	Discard	Associated item No.	Movement	Playbook files	Include order	
History	Update	Discard		1 2:Test Movement	Sample1		1

(4) The list of registration screen items are as follows.

- In Ansible-Legacy

Table 5.3-7 Registration screen item list (Movement details in Ansible-Legacy console)

Item	Description	Input required	Input type	Restrictions
Movement	The Movement registered in the Movement list will be displayed. Select the Movement.	<input type="radio"/>	List	-
Playbook file	The Playbook file registered in "5.3.3 Playbook file list (Ansible-Legacy only)" will be displayed. Select the Playbook file.	<input type="radio"/>	List	-
Include order	Enter the execution order of playbook files (unique value starts from 1). Playbook files will be executed as the entered include order (ascending).	<input type="radio"/>	Manual	Half-width integer
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

- In Ansible-Legacy Role

Table 5.3-8 Registration screen item list (Movement details in Ansible-Legacy Role console)

Item	Description	Input required	Input type	Restrictions
Movement	Same as Ansible-Legacy	<input checked="" type="radio"/>	List	-
Role package name	The role package registered in the role package list menu will be displayed. Select the role package to be executed. Multiple role packages cannot be registered in the same Movement.	<input checked="" type="radio"/>	List	-
Role name	The role names included in role package selected in role package name are displayed. Select the role in the role package to be executed.	<input checked="" type="radio"/>		-
Include order	Same as Ansible-Legacy	<input checked="" type="radio"/>	Manual	Half-width integer
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

- In Ansible-Pioneer

Table 5.3-9 Registration screen item list (Movement details in Ansible-Pioneer console)

Item	Description	Input required	Input type	Restrictions
Movement	Same as Ansible-Legacy.	<input checked="" type="radio"/>	List	-
Dialog type	The dialog type registered in "Dialog type list (Ansible-Pioneer list)" will be displayed. Select the dialog type of dialog file to be executed.	<input checked="" type="radio"/>	List	-

	The dialog file linked with the OS type and dialog type of every host are execution target.			
Include order	Same as Ansible-Legacy	<input type="radio"/>	Manual	Half-width integer
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

5.3.8 Nested variable list (Ansible-Legacy Role only)

- (1) In the "Nested variable list" menu, update the maximum iteration count of member variable array defined as nested array in the nested variable which is defined in the role package registered in "5.3.4 Role package list (Ansible-Legacy Role only)". Click the update button of the member variable that the user want to change and update the maximum iteration count.

Figure 5.3-22 Submenu screen (Nested variable list)

- (2) Click the "List" - "Update" button to update the maximum iteration count.

(※Not the registration button)

Item No.	Variable name	Member variable name (iteration)	maximum iteration count*		Last update date/time	Last updated by
1	VAR_users	-	2		Auto-input	Auto-input

Figure 5.3-23 Registration screen (Nested variable list)

(3) The list of registration screen items are as follows.

Table 5.3-10 Registration screen item list (Nested variable list)

Item	Description	Input required	Input type	Restrictions
maximum iteration count	Enter the maximum iteration count of the array in the range of 1~99,999,999.	○	Manual	Value 1~99,999,999
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

The display of member variable names are the variables of each hierarchy level scoped with ".".

Also, if the first level is nested array, the member variable name will be displayed as "-".

e.g.)

Variable definition	Display of member variable	default value of the maximum iteration count
VAR_users:	-	1
- name: alice		
authorized:		
- /tmp/alice/onekey.pub		
nested:	nested	2
- craete_users:		
Name: root		
password: xxxxxxxx		
- craete_users:		
Name: mysql		
password: xxxxxxxx		

The internal process initially registers the iteration count of member variable defined in the nested variable which is defined in the role package. After the initial registration, the iteration count can be updated in the "Nested variable list" menu.

Also, since initial registration and update of iteration count is not in real time, it may take some time※ 1 until the variables can be handled in menu "5.3.9 Substitution value auto-registration setting" and menu "5.3.11 Substitution value list".

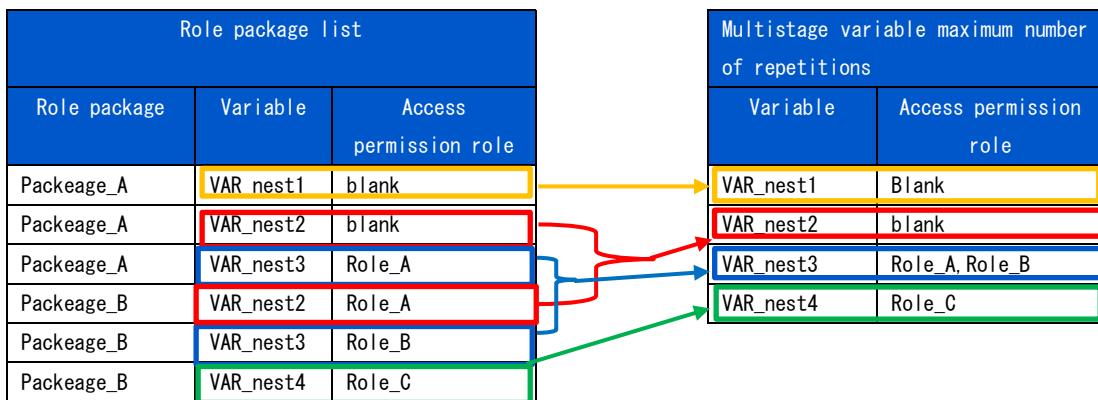
※1 The timing of extraction is writed in "7.2 About the maintenance method", so please refer to it.

(4) Unique list of variable names

Variable name list is unique in all role packages. When using the same variable name across role packages, the number of repetitions set in multistage variable maximum repeat list applies to variables in all role packages.

(5) Access permission role

The permission role is set for multistage variable maximum repeat list is set to the role package management permission role for which the variable will be defined. If a variable is defined in more than one role package list, all permission roles in the role package lists will be set. If the permission role is empty, access to all roles will be treated as accessible. If the access permission role for each role package list is blank, the permission role for multistage variable maximum repeat list will be also set to blank. For more information about access permission roles, please refer to the "User_Instruction_Manual_Role-Based Access Control."



5.3.9 Substitution value auto-registration setting

- (1) Link the parameter sheet created in menu creation function with the variables in the Movement. The registered information will reflected to "substitution value list" menu and "target host" menu by internal process.

The reflection rule is writed in “6.6[BackYard contents](#) (2) Substitution value auto-registration setting”.

The screenshot shows the Ansible-LegacyRole interface. At the top, there is a header with the Exastro logo, the title 'Ansible-LegacyRole', and user information: 'User name [System Administrator]' and 'Login ID [administrator]'. Below the header are two buttons: 'Change password' and 'Logout'. On the left, a sidebar menu lists various options: Main menu, Movement list, Role package list, Movement details, Nested variable maximum iteration count list, Substitution value auto-registration setting (which is highlighted with a red box), Target host, Substitution value list, Execution, Check operation status, and Execution list. The main content area has a title 'Description' with a '▽ Open' button. Below it is a 'Display filter' section containing a table with columns: Discard, Item No., Menu group, Last update date/time, and Last updated by. There are search and filter buttons at the bottom of this section. Further down, there are four buttons: 'List/Update', 'Register', 'Download all and edit file uploads', and 'Trace history', each with a '▽ Open' button. At the bottom of the page is a 'Contact administrator' button.

Figure 5.3-24 Submenu screen (Substitution value auto-registration setting)

※The screen is from Ansible-Legacy Role

- (2) Click the "Register" - "Start Registration" button to set the substitution value auto-registration.

Item No.	Parameter sheet		Registration method*	Movement	IaC variable		
	Menu group:Menu	Item			Key variable		
					Variable name	Member variable name	Substitutio
Auto-input	<input type="button" value="▼"/>	Select menu	<input type="button" value="▼"/>	<input type="button" value="▼"/>	Select Movement	Select variable name	<input type="text"/>

Figure 5.3-25 Registration screen (Substitution value auto-registration setting)

(3) Clicking the "List/Update" Menu ID/Menu link will move the user to the target menu..

Parameter sheet(From)								Registration method
Menu group				Menu		Item		
ID	Name	ID	Name					
History	Update	Discard	Item No.	1	2100011611 Substitution value	2	Test Menu 1 Parameter/Item 1	Key type

Filter result count: 1

Figure 5.3-26 Submenu screen (Substitution value auto registration setting)

Table 5.3-11 corresponding column list (Substitution value auto-registration setting)

Column		Legacy	Legacy Role	Pioneer
Menugroup:Menu		○	○	○
Item		○	○	○
Registration method		○	○	○
Movement		○	○	○
Key variable	Variable name	○	○	○
	Member variable name	—	▲	—
	Substitution name	△	△	△
Value variable	Variable name	○	○	○
	Member variable name	—	▲	—
	Substitution name	△	△	△
NULL link		●	●	●

○: Required

●: Optional

△: Required only if multiple specific value can be set to the selected variable.

▲: Required only if the selected variable is nested variable

—: Not displayed

(4) The list of registration screen items is as follows.

Table 5.3-12 Registration screen item list (Substitution value auto-registration setting)

Column	Description	Input required	Input type	Restrictions
Menugroup:Menu	The menu of parameter list is displayed. Select the menu of association target.	<input type="radio"/>	List	-
item	The item of selected parameter list menu is displayed. Select the item of association target.	<input type="radio"/>	List	-
Registration method	Value type: Select to set the setting value of item as the specific value of the linked variable. Key type : Select to set the name of item as the specific value of the linked variable. If the setting value of the item is blank, it cannot be linked. Key-Value type : Select to set the name(Key) and setting value(Value) of item as the specific value of the linked variable.	<input type="radio"/>	List	-
Movement	The Movement registered in the Movement list will be displayed. Select the Movement.	<input type="radio"/>	List	-

Column		Description	Input required	Input type	Restrictions
Key variable	Variable name	The variables used in the file registered in Movement details menu are displayed. Select the variable to associate with its specific value in key type.	<input type="radio"/> or <input checked="" type="checkbox"/>	List	Required if the registration method is key type of key-value type.
	Member variable name	If nested variable is selected in the variable name column, the member variable of nested variable will be displayed. Select the member variable.	<input type="radio"/> or <input checked="" type="checkbox"/>	List	
	Substitution name	Required only if multiple specific value can be set to the selected variable Enter the substitution order (1~) of specific value. Value will be substituted in ascending order following the entered value. Please enter the substitution order (from 1) even if there are no more specific value.	<input type="radio"/> or <input checked="" type="checkbox"/>	Manual	Blank or positive integer.
Value variable	Variable name	The variables used in the file registered in Movement details menu are displayed. Select the variable to associate with its specific value in value type.	<input type="radio"/> or <input checked="" type="checkbox"/>	Manual	Required if the registration method is key type of key-value type.
	Member variable name	If nested variable is selected in the variable name column, the member variable of nested variable will be displayed. Select the member variable.	<input type="radio"/> or <input checked="" type="checkbox"/>	List	-
	Substitution name	Required only if multiple specific value can be set to the selected variable Enter the substitution order (1~) of specific value. Value will be substituted in ascending order following the entered value. Please enter the substitution order (from 1) even if there are no more specific value.	<input type="radio"/> or <input checked="" type="checkbox"/>	Manual	Blank or positive integer.
NULL link		Set whether to register NULL (blank) value to substitution value list menu if the specific value in parameter sheet is NULL (blank). • If the "Valid" is set, any value in the parameter sheet will be registered in the substitution value list menu. (NULL value will be registered) • If the "Invalid" is set, only specific value in the parameter sheet will be registered in the substitution value list menu (NULL value will not be registered) • If the column is blank, the "NULL link" value in Ansible interface information menu will be applied.	-	List	-
Remarks		Free description field.	-	Manual	Maximum length 4000 bytes.

※ Please refer to “[5.3.11 Substitution value list](#)” for the description of member variable name.

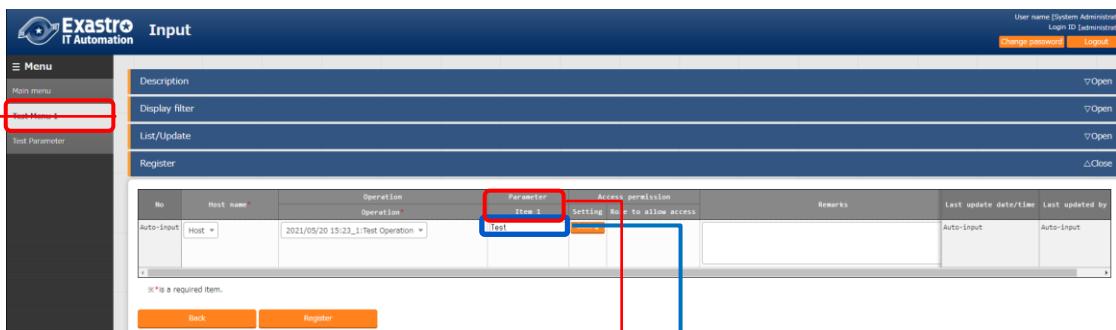
(5) If the "Ansible Common : Template list:Template Variable name / Ansible Common: File list:File embedded variable names." are used as Parameter sheet items in the Substitute value automatic registration settings,

The selected item setting value's linked Variable's specific value (Variable name) will be displayed in the Substitute value list as "{{ Variable name }}"

Parameter sheet definition



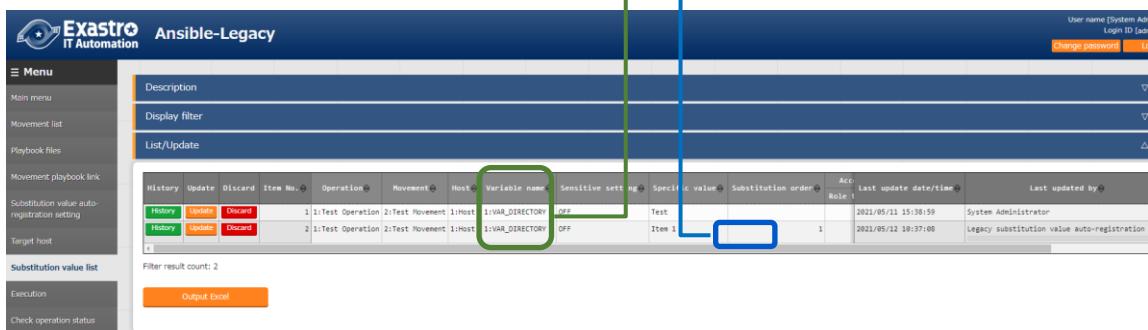
Parameter sheet



Substitution value auto-registration setting

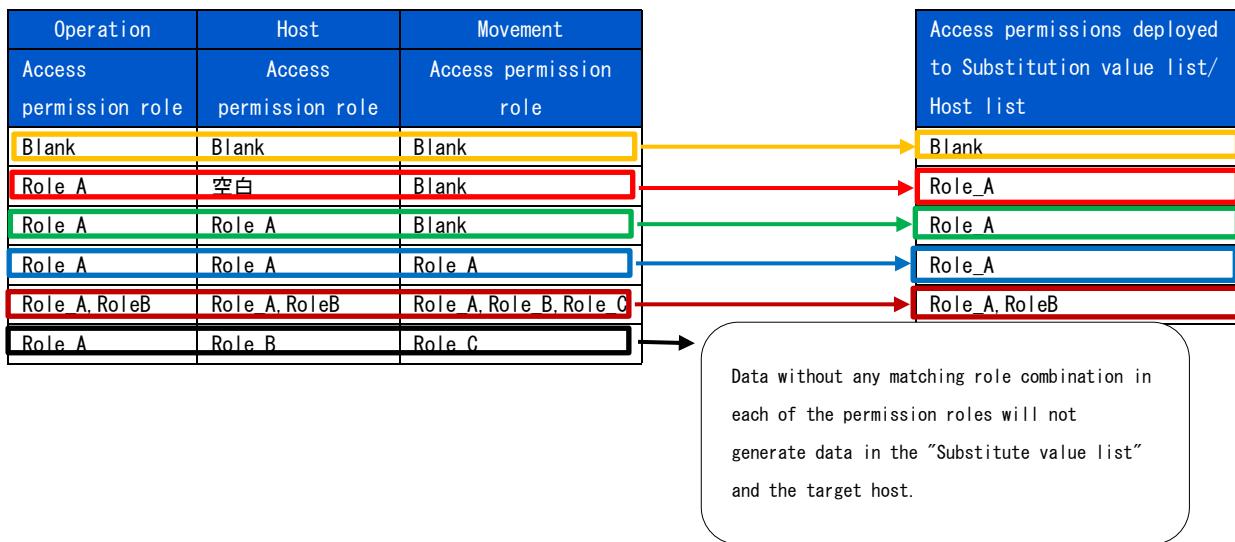


Substitution value list



(6) Access permission role

The access permission roles set for the "Substitute value list" and "Operation target hosts" generated from the information in the "Automatic substitute value registration" will be set to the roles that match the access permission roles for each of the hosts (device list) and Operations set in the Movement and parameter sheets set in the "Automatic substitute value registration". If the permission role is empty, access to all roles will be treated as accessible. Data without any matching role combination in each of the permission roles will not generate data in the "Substitute value list" and the target host. For more information about access permission roles, please refer to the "User Instruction Manual Role-Based Access Control."



(7) Parameter sheet (Vertical menus)

When using parameter sheets (Vertical menus), repeated items with BLANK values will not be displayed in the Substitute value list, even if the NULL settings in the Substitute value auto registration settings are Valid.

Create menu

The screenshot shows the 'Create Menu' interface. On the left, there's a sidebar with options like 'Main menu', 'Create - Define menu', 'Menu definition information', etc. The main area displays a vertical menu structure. A red box highlights a section where 'Repeat Item 1' and 'Repeat Item 2' are defined. Both items have 'String' type, 'Maximum number of bytes* 256', and 'Regular expression' fields. Under 'Default value', there are checkboxes for 'Required' and 'Unique constraint'. Below these are 'Explanation' and 'Remark' sections. The 'Repeat Item 1' and 'Repeat Item 2' sections are grouped under a 'REPEAT 1' header.

Parameter sheet

The screenshot shows the 'Input' interface. On the left, there's a sidebar with 'Main menu', 'sampleA', 'Directory settings', and 'Example menu'. The main area has tabs for 'Description', 'Display filter', and 'List/Update'. Under 'List/Update', there's a table showing history entries. To the right, a red box highlights a 'Parameter' section. It contains four rows: 'Normal Item 1' with 'ValueA', 'Repeat Item 1' with 'ValueB', 'Repeat Item 2' with 'ValueC', and 'Normal Item 2' with 'ValueD'. Below this is a 'NULL link' dropdown set to 'VALID'.

Substitute value auto registration settings

The screenshot shows the 'Movement playbook link' interface. On the left, there's a sidebar with 'Variable name list', 'Movement variable association list', 'Substitution value auto-registration setting', 'Target host', and 'Substitution value list'. The main area shows a table of movement items. A red box highlights the 'IaC variable(To)' column, which lists variables like '1:VAR_test1' through '6:VAR_test6'. To the right, another red box highlights the 'NULL link' column, which is also set to 'VALID'.

Substitution value list

The screenshot shows the 'Substitution value list' interface. On the left, there's a sidebar with 'Substitution value auto-registration setting', 'Target host', 'Substitution value list', 'Execution', and 'Check operation status'. The main area is a table of substitution entries. A red box highlights the '値' (Value) column, which shows 'valueA' for most rows and 'valueC' for one row. The '代入順序' (Assignment Order) column shows values 1, 1, 1, 3, and 3 respectively.

The repeat items in the ValueB row all have blank values, meaning that they will not display in the Substitute value list, even if the NULL link is set to Valid.

5.3.10 Target host

- (1) Register/update/discard the Movement and host linked with Operation in the "Target host" menu.

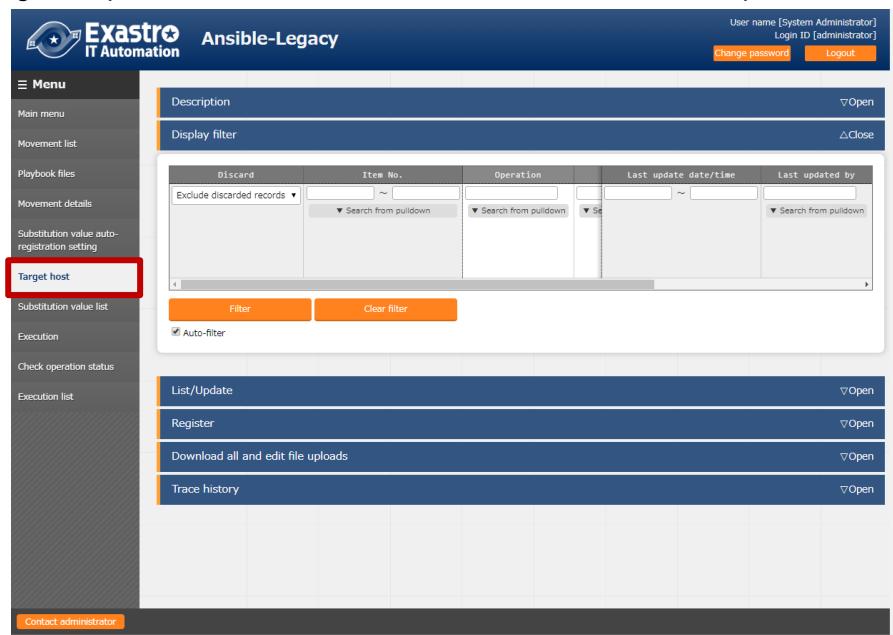


Figure 5.3-27 Submenu screen (Target host)

- (2) Click the "Register" - "Start Registration" button to register the operation target host.

Item No.	Operation*	Movement*	Host*	Remarks
Auto-input				

※ * is a required item.

Back Register

Figure 5.3-28 Registraton screen (Target host)

- (2) Clicking the Movement link in the “List/Update” submenu will move the user to the target 5.3.7 Movement-Playbook Link. Pressing the Substitution value management button will take the user to the target 5.4.11 Substitution value management

History	Update	Discard	Item No. ▾	Operation ▾	Movement ▾	Host ▾	Substitution value management	Access permission	Remarks ▾
							Role to allow access		
History	Update	Discard	1	1:Test Operation	2:Test_Movement	1:Host	Substitution value management		Test

Filter result count: 1

Figure 5.3-29 Registraton screen (Target host)

(3) The list of registration screen items are as follows.

Table 5.3-13 Registration screen item list (Target host)

Item	Description	Input require	Input type	Restrictions
operation	The Operation registered in the input operation list is displayed. Select the Operation.	<input type="radio"/>	List	-
Movement	The Movement registered in the Movement list is displayed. Select the Movement to associate with Operation.	<input type="radio"/>	List	-
Host	The host name registered in the device list will be displayed. Select the host to be linked with the Operation.	<input type="radio"/>	List	-
Remarks	Free description field.	-	Manual	Maximum length 4000 bytes

5.3.11 Substitution value list

(1) Register/update/discard the substitution value of variable.

Users can perform maintenance (view/register/update/discard) of the specific values that are substituted with variable "VAR_" of Playbook and template file to be used in target Movement for each operation.

Users can also maintain the specific values that are substituted with variable "LCA_" other than "VAR_" according to the definition of translation table. Please refer to "[6.5 Write translation table \(Ansible-Legacy Role only\)](#)" for details.

The registered variable information will be output into host variable file (under host_vars/) during operation execution.

Figure5.3-21 Submenu screen (Substitution value list)

※The screen is from Ansible-Legacy Role.

(2) Click the "Register" - "Start Registration" button to manage the substitution value.

Figure 5.3-22 Registration screen (Substitution value list)

The variable name in substitution value list menu is reflected from the uploaded Playbook and the information registered in the substitution value auto-registration setting menu.

※ The timing of extraction is written in "7.2 About the maintenance method", so please refer to it.

(3) The list of registration screen items are as follows.

Table 5.3-14 corresponding column list (Substitution value list)

column	Legacy	Legacy Role	Pioneer
Operation	○	○	○
Movement	○	○	○
host	○	○	○
variable name	○	○	○
Member variable name	—	▲	—
Substitution order	△	△	△
Default value(display only)	—	○	—

○: Required

△: Required only if multiple specific value can be set to the selected variable

▲: Required only if the selected variable is nested variable

—: Not displayed

Table 5.3-15 Registration screen item list (Substitution value list)

Item	Description	Input required	Input type	Restrictions
Operation	The Operation registered in the operation target host is displayed. Select the Operation.	○	List	-
Movement	The Movement linked with the Operation selected from the data registered in the target host menu is displayed. Select the Movement.	○	List	-
Host	The host linked with the Operation and Movement selected from the data registered in the target host menu is displayed. Select the host.	○	List	-
Variable name	The variable name attached with the Movement selected from the data registered in the Movement-Playbook link menu is displayed. Select the variable.	○	List	-
Member variable name	If nested variable is selected in the variable name column, the member variable of nested variable will be displayed. Select the member variable.	○ or /	List	-
Sensitive settings	Select "OFF" or "ON". If "ON" is selected, the specific value will be encrypted and will not be displayed on ITA.		Button	

	<ul style="list-style-type: none"> For Legacy/Legacy-Role Host variable files passed to Ansible will be set with contents encrypted in Ansible-Vault. For Pionner Host variable files passed to Ansible will be set with ITA's original encrypted content. 			
Specific value 値※1	<p>Enter the specific value used in Operation / Movement / Host. File embedded variable "CPF_" and template embedded variable "TPF_" can be entered in the specific value column. When describing the variable, enclose the variable name in {{}} as describing them in the Playbook. e.g.) Entering TPF_sample as specific value. '{{△TPF_sample△}}' △: Half-width space ': recommended</p>	<input type="radio"/>	Manual input	Maximum length 1024 bytes
Substitution order	<p>Required only if multiple specific value can be set to the selected variable. Enter the substitution order (1~) of specific value. Value will be substituted in ascending order following the entered value. Please enter the substitution order (1~) even if there are no multiple specific value.</p>	<input type="radio"/> or <input type="checkbox"/>	Manual input	Blank or positive integer
Default value	<p>The specific value of variable selected in the variable name or member variable name column set in the default variable definition file(defaults->main.yml) is displayed Please refer to "6.4 Write ITA readme (Ansible-Legacy Role only)" for details. True is displayed when the specific value is "Yes","Y", or "y". False is displayed when the specific value is "No","N", or "n".</p>	-	Display only	-
Remarks	Free description field.	-	Manual input	Maximum length 4000 bytes

※1 If you are going to set filed embedded variables (CPF) or template embedded variables (TPF) to the specific values, make sure that the Sensitive settings are set to "OFF".

If the Sensitive settings are set to "ON", the variables will not be used.

【The display content of member variable name】

Selecting member variable is required only if the variable is nested variable.

Only the variable that requires specific value is displayed in the member variables.

The display of variable names of each hierarchy level is scoped with "." .

If the variable is in nested array, the variables are scoped with "[]" at the iteration position (0~).

The iteration array count is set in "["5.3.8 Nested variable list \(Ansible-Legacy Role only\)"](#)".

e.g.)

Variable definition	Display of member variable
VAR_users:	
- name: alice	[0].name
authorized:	[0].authorized
- /tmp/alice/onekey.pub	
mysql:	
password: mysql-password	[0].mysql.password
hosts:	[0].mysql.hosts
- "127.0.0.1"	
- "localhost"	
- name: bob	[1].name
authorized:	[1].authorized
- /tmp/alice/onekey.pub	
mysql:	
password: mysql-password	[1].mysql.password
hosts:	[1].mysql.hosts
- "127.0.0.1"	※ mysql is the variable which indicates the hierarchy
- "localhost"	directory, so is not displayed in member variable.

The information registered in "substitution value auto registration setting" menu is reflected to "substitution value list" menu and "target host" menu by internal process.

※ The timing of extraction is writed in "["7.2 About the maintenance method"](#)", so please refer to it.

① Entering the substitution order

In Ansible-Legacy, if the substitution order is not entered, the variable will be treated as normal variable.

If the substitution order is entered, the variable will be handled as multiple specific value variable.

Please enter the substitution order although multiple specific value is not required (one specific value is sufficient) if the variable is multiple specific value variable.

In Ansible-Legacy Role, by selecting variable name or member variable name, it is possible to enter substitution order only for multiple specific value variables.

Please enter if the variable is multiple specific value variables.

In Ansible-Pioneer, if the substitution order is not entered, the variable will be handled as normal variable.

If the substitution order is entered, the variable will be handled as multiple specific value variable

Please enter the substitution order although multiple specific value is not required (one specific value is sufficient) if the variable is multiple specific value variable.

In each mode, it is no problem although the substitution order is not consecutive for specific multiple concrete value variables.

e.g.)

**Registration in substitution
value list menu**

Host	Variable	Specific Value	Substitution order
HOST_A	VAR_std	value1	
HOST_A	VAR_list_a	value2	10
HOST_A	VAR_list_b	value3	100
HOST_A	VAR_list_b	value4	200

The content output to the host variable file of HOST_A

```
VAR_std: value1
VAR_list_a:
- value2
VAR_list_b:
- value3
- value4
```

② Output to the host variable file

The specific value of variable registered in substitution value list menu will be output to host variable file.

In Ansible-Legacy and Ansible-Pioneer, if the specific value of variable used in Playbook or dialog file is not registered in substitution value list menu during operation execution, unexpected error will occur.

In Ansible-Legacy Role, only the variable registered in substitution value list menu the will be output to host variable file during operation execution.

It is same for nested variables that only the member variable registered specific value will be output.

e.g.)

Variable definition

```
VAR_users:
  - name: alice
    authorized:
      - /tmp/alice/onekey.pub
  mysql:
    password: mysql-password
    hosts:
      - "127.0.0.1"
      - "localhost"
  - name: bob
  omitted
```

Registration in substitution value list menu

Host	Variable	Member variable	Specific value	Substitution order
HOST_A	VAR_users:	[0].name	value1	
HOST_A	VAR_users	[1].authorized	value2	

The content output to the host variable file of HOST_A

```
VAR_users:
  - name :value1
  - .authorized: value2
```

③ Default value check option

In the "System settings" of "ITA Management console", users can set the parameter to display warning message and not register the specific value when registering the specific value of the variable whose default value does not match between multiple roles.

This parameter is not registered by default. Please register if necessary.

The content to register in system settings is as follows.

Also, please refer to "User instruction manual_Ansible-Management console" for system settings.

Table 5.3-16 Registration content in system settings

Item	Input value	Input required
ID	ANSIBLE_DEF_VAL_CHK	<input type="radio"/>
Item name	Any desired string	-
Setting value	1: Parameter enabled Contents other than 1 or record not registered: Parameter disabled.	<input type="radio"/>
Remarks	Any desired string	

5.3.12 Check operation status

- (1) Monitor the status of operation execution.

The screenshot shows the Exastro IT Automation Ansible-Legacy web interface. On the left, there is a vertical navigation menu with the following items:

- Main menu
- Movement list
- Playbook files
- Movement details
- Substitution value auto-registration setting
- Target host
- Substitution value list
- Execution
- Check operation status** (highlighted with a red box)
- Execution list

The main content area has a header "Ansible-Legacy" and two tabs: "Description" and "Target Operation". Below these tabs is a large table containing various configuration parameters for an operation. The table has columns for "Item" and "Value". Some rows have additional columns for "Host specific format" and "Host name". The table includes the following data:

Item	Value												
Execution No.	3												
Execution type	Normal												
Status	Completed												
execution engine	Ansible												
Caller symphony	test												
Execution user	System Administrator												
Movement	<table border="1"><tr><td>ID</td><td>1</td></tr><tr><td>Name</td><td>testing-one</td></tr><tr><td>Delay timer (minutes)</td><td></td></tr><tr><td>Dedicated information for ansible</td><td></td></tr><tr><td>Host specific format</td><td></td></tr><tr><td>WinRM connection</td><td></td></tr></table>	ID	1	Name	testing-one	Delay timer (minutes)		Dedicated information for ansible		Host specific format		WinRM connection	
ID	1												
Name	testing-one												
Delay timer (minutes)													
Dedicated information for ansible													
Host specific format													
WinRM connection													
Operation	<table border="1"><tr><td>No.</td><td>1</td></tr><tr><td>Name</td><td>execution</td></tr><tr><td>ID</td><td>1</td></tr></table>	No.	1	Name	execution	ID	1						
No.	1												
Name	execution												
ID	1												
Host management	<table border="1"><tr><td>confirmation</td></tr><tr><td>confirmation</td></tr></table>	confirmation	confirmation										
confirmation													
confirmation													
Substitution value	<table border="1"><tr><td>Populated data</td><td>InputData_000000003.zip</td></tr><tr><td>Result data</td><td>ResultData_000000003.zip</td></tr></table>	Populated data	InputData_000000003.zip	Result data	ResultData_000000003.zip								
Populated data	InputData_000000003.zip												
Result data	ResultData_000000003.zip												
Scheduled date/time													
Operation status	<table border="1"><tr><td>Start date/time</td><td>2020/02/10 11:59:04</td></tr><tr><td>End date/time</td><td>2020/02/10 11:59:24</td></tr></table>	Start date/time	2020/02/10 11:59:04	End date/time	2020/02/10 11:59:24								
Start date/time	2020/02/10 11:59:04												
End date/time	2020/02/10 11:59:24												

At the bottom of the page, there is a "Progress status(Execution log)" section and an orange "Contact administrator" button.

Figure 5.3-23 Submenu screen (Check operation status)

① Display of execution status

"Status" is displayed according to the execution status.

Also, the details of the execution status is displayed in execution log and error log

In the "execution type", "Dry run" is displayed when performing dry run, "Normal" will be displayed for other cases

If the status ends with an unexpected error, the cause is incomplete registration of web contents, message will be displayed in error log.

In addition, in the case that communication with Ansible RestAPI fails due to incomplete registration in "[5.2.1 Interface information](#)", message will not be displayed in error log.

In this case, error information will be record in application log. Please check the application log if necessary.

The symphony which the operation is executed from is displayed in "Caller symphony"

The column will be blank if the operation is executed directly from Ansible-Legacy, Pioneer, LegacyRole driver.

The login user when clicking the "execute" or "dry run" button in the "exeuction" menu will be displayed in "Execution User".

② Host management

By clicking the "confirmation" button, "[5.3.10 Target host](#)" will display and the host filtered by the operation and Movement of operation target will be displayed.

③ Substitution value confirmation

By clicking the "confirmation" button, "[5.3.11 Substitution value list](#)" will display and the substitution value filtered by the operation and Movement of operation target will be displayed.

④ Emergency stop/ Schedule cancellation

It is possible to stop the construction operation by clicking the "Emergency stop" button

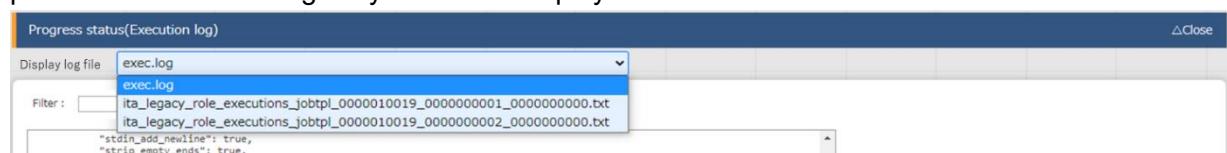
In addition, for the "scheduled execution" operation before execution, the "schedule cancellation" button will display. Cancel the scheduled execution by clicking the "schedule cancellation" button.

⑤ Display of execution log

When Ansible Automation Controller is executed, the Playbook is executed in units of the device to be built grouped by which item value, such as the user password instance group in the list of devices to be built and the ansible execution log is split.

In addition, by specifying the number of job slices in the optional parameters in the Movement list, the grouped device to be built is further divided by the number of job slices, the playbook is executed and the ansible execution log is also divided.

When the execution log is splitted, the pull-down of the display log file will be displayed and possible to select the log file you want to display.



There are two types of log file names displayed in the pull-down of the display log file.

exec.log: This is a log file that summarizes all execution logs.

Without exec.log: Splited execution log file. The file naming conventions is as follows.

Ita_<mode name>_executions_jobtpl_<work number>_<group number>_<serial number>

Table 5.3-17 Naming elements for split execution log files.

Element	Content
Modename	Executed mode name legacy/pioneer/legacy_role
Execution number	Executionnumber of execution list menu.
Group number	Serial number from 1 that is grooved by the item value of the user, password, instance groove etc. of the device list and the device to be built.
Serial number	Serial number from 1 that divides the group by setting the number of job slices. If 0, no division of job slicing was done.

⑥ Log filter

Execution log and error log can be filtered. By entering the string that the user wants to search in the filter box of each log and checking the "Display only corresponding lines" checkbox, only the corresponding line will be displayed.

The display refresh cycle and the maximum display line count of execution and error log can be set in "Status monitoring cycle (milliseconds)" and "Number of rows to display progress status" of "5.2.1 Interface information" menu.

⑦ Input data

Users can download files such as the executed Playbook.

Please refer to "8.1 The linkage between the input data used during Ansible execution and ITA menu" for the configuration of input data.

⑧ Result data

Users can download files such as execution log and error log.

5.3.13 Execution list

- (1) The history of operation can be viewed here.

The operation list table and graph will display by specifying criteria and clicking the "filter" button.

By clicking the "Check execution status" button, the screen will transit to "[5.3.12 Check operation status](#)" and the details of execution status can be viewed.

Discard	Execution No.	Execution type	Status	execution engine	virtualenv	Caller symphony	Last update date/time	Last updated by
All records	~	~	~	~	~	~	~	~
75	Check execution status	Normal	Completed	Ansible Engine		Sample1	System	2021/02/01 13:13:20
74	Check execution status	Normal	Completed	Ansible Engine		Sample1	System	2021/01/27 09:11:00
73	Check execution status	Normal	Completed	Ansible Engine	Workflow1	Sample1	System	2021/01/26 16:59:51
72	Check execution status	Normal	Unexpected error	Ansible Engine		Sample1	System	2020/12/18 15:49:43
71	Check execution status	Normal	Unexpected error	Ansible Engine		Sample1	System	2020/12/18 15:48:58

Figure 5.3-24 Submenu screen (Execution list)

5.3.14 Execution

- (1) Indicate Operation execution. Select the radio button from the Movement list and operation list and click the execution button, the screen will transit to "5.3.12 Check operation status" and the operation will be executed.

The screenshot shows the Ansible-Legacy interface with the 'Execution' submenu selected. The 'Movement [List]' section shows one entry:

Select	Movement ID	Movement Name	Orchestrator	Delay timer	Host specific format	Last update date/time	Last updated by
<input checked="" type="radio"/>	1	testing-one	Ansible Legacy		Host name	2020/02/10 10:52:16	System Administrator

The 'Operation [List]' section shows two entries:

Select	No.	Operation ID	Operation name	Scheduled date for execution	Last ex	Last update date/time	Last updated by
<input checked="" type="radio"/>	1	1	1 execution	2020/02/10 10:43	2020/02/10 10:43	2020/02/12 08:33:05	Legacy execution procedure
<input type="radio"/>	2	2	2 legacy	2020/02/10 14:31	2020/02/10 14:31	2020/02/10 14:31:41	System Administrator

At the bottom, there are buttons for 'Dry run' and 'Execute'.

Figure 5.3-25 Submenu screen (Execution)

※The screen is from Ansible Legacy

① Dry run

By clicking the "Dry run" button, dry run can be executed without actually constructing the target device. In the case of dry run, the operation of each mode is as follows.

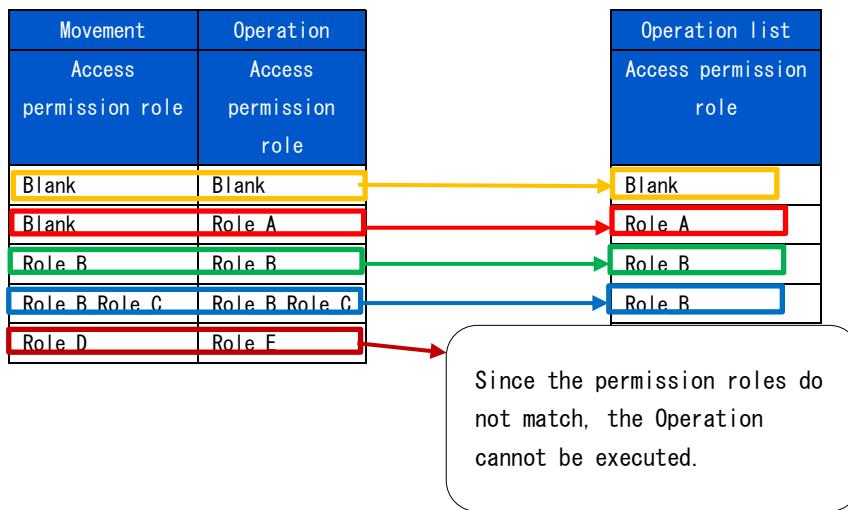
Driver	Action
Ansible-Legacy	Execute the playbook by specifying the -check parameter to the Ansible-Playbook command.
Ansible-Legacy Role	Execute the role by specifying the - check parameter to the Ansible-Playbook command.
Ansible-Pioneer	Only perform the connection check to target device

② Specify scheduled date/time

Execution can be scheduled by entering "Scheduled date/time" column.
Only future date/time can be registered for "Scheduled date/time"

(2) Determining access permission role compatibility when executing

Determines whether there are matching roles for each access permission role in the Movement and Operations selected in the Movement list and Operations list. If there are no matching roles, an error message will be displayed and the operation cannot be executed. Matching roles will be set to have access to Operation lists. If the permission role is blank, all the roles will be handled as accessible. If each access permission role is blank, the operation list access permission role will be also set to blank. For more information about access permission roles, please refer to the "User Instruction Manual_Role-Based Access Control."



6 How to write construction code

6.1 Write Playbook (Ansible-Legacy)

Playbooks uploaded to [5.3.3 Playbook file list](#) (Ansible-Legacy only) are included in the Playbook file generated by ITA and excuted in Include format. The Master playbook created by ITA are constructed by the Header section and the Tasks section.

(1) Header section

The playbook does not require a header section when being uploaded.

The header section has a default value, but you can change it in the header section of ["5.3.2.Movement List"](#).

```
Defult value of header section
•Ansible Core           •Ansible Automation Controller
- hosts: all           - hosts: all
  remote_user: "{{ __loginuser__ }}"   gather_facts: no
  gather_facts: no           become: yes
  become: yes
```

(2) tasks section

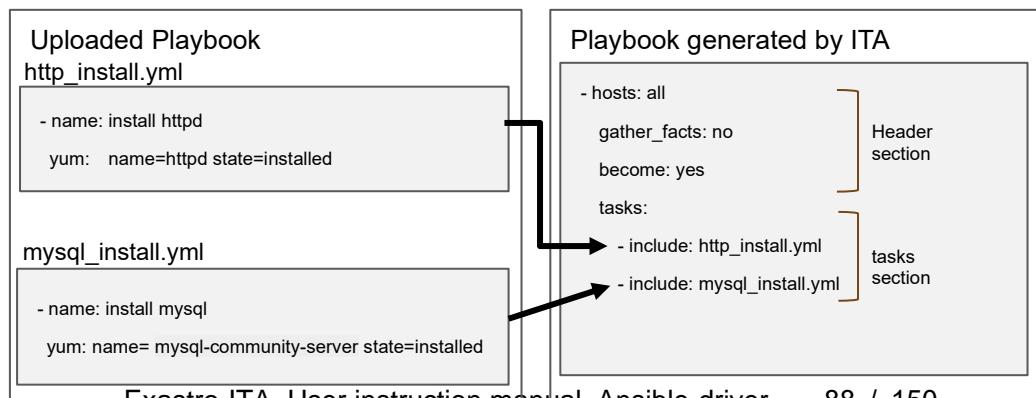
Please refer to the official manual for the basic Playbook format.

Please adjust the indent in the Playbook to multiple of 2.

Make sure the character code is UTF-8 without BOM.

```
e.g.)
-△name: comment
△△template:
△△△△src: "{{ item.src }}"
△△△△dest: "{{ item.dest }}"
△△△△owner: "{{ item.owner is none |ternary('root', item.owner) }}"
△△△△group: "{{ item.group is none |ternary('bacula', item.group) }}"
△△△△mode: "{{ item.mode is none |ternary('0654', item.mode) }}"
```

Uploaded Playbooks are included in the include order of ["5.3.7 Movement details"](#).



6.2 Write Dialog file (Ansible-Pioneer)

The dialog file in Ansible-Pioneer incorporates an ITA-specific module into Ansible.

The dialog file is in ITA-specific format.

Please use UTF-8 for the character encoding.

(1) Structure of dialog file

The dialog file is constructed with 2 types of section.

Section name	Usage
Conf	Specify the timeout value according to timeout parameter. timeout value: 1~3600(unit:second)
exec_list	Construct the target host with 4 kinds of dialog command.

Write the timeout parameter in the beginning of dialog file then write the dialog comment in the later part.

Comments can be written in the same format with Ansible basic format.

```
e.g.)  
# Comment  
conf:  
△△timeout: 10  
exec_list:  
※△: Half-width space  
Please enter 2 half-width space before "timeout:"
```

(2) Dialog command

There are 4 kinds of dialog commands as follows.

Module	Use
exec	Input command to the target host
expect	Waiting for the output of expected string (prompt) from the contents that the target host outputs to the standard output.
state	Input the command to target host. The contents of the standard output until the prompt is output to the standard output are analyzed by external shell, and the result is determined.
command	Loops and conditional branching can be performed before and after inputting commands to the target host.

① expect module

Waiting for the output of expected string (prompt) from the contents that the target host outputs to the standard output.

Write the expected string **in regular expression**.

When the expected string is received, proceed to the next. In addition, if the string is not received within the time specified by the time out parameter, the dialog file will terminate abnormally.

```
e.g.) Waiting for the prompt of password entry via telnet connection  
△△-△expect:△'Password'  
※△: half-width space  
Please enter 2 half-width space before "- expect:"  
It is recommended to enclose the waiting string with quotation.
```

② exec module

Input command to the target host.

exec module and expect module are used in pairs.

e.g.) Wait for the password entry prompt via telnet connection and input password.

△△-△expect:△'Password'

△△△△exec:△itapassword

※△: Half-width space

Please enter 4 half-width space before the description of "exec:"

It is recommended to enclose the waiting string with quotation if necessary.

③ state module

Input the command to target host. The contents of the standard output until the prompt is output to the standard output are analyzed by external shell, and the result is determined.

The format of state module

Parameter	Required/ Optional	Description
△△-△state:△xxx	Required	Specify the input command.
△△△△prompt:△xxx	Required	Specify the waiting prompt. The prompt can be writed in regular expression.
△△△△shell:△xxx	Optional	Specify the shell file name to check the result with the created shell. If the exit code of the created shell is 0, the result is determined as normal, and the others are abnormal. This parameter is not required when checking the result with default shell. The default shell will grep the contents of standard output with the string specified by parameter (-). If there is at least one matching row, the result is determined normal, and if there is no matching row, the result is determined to be abnormal. In addition, if the parameter is not specified, the result will be determined as abnormal. In the case of using the parameter to save the command result (standard output) to the file specified by stdout_file, please specifiy yes for ignore_errors.
△△△△parameter: △△△△△△-△xxx △△△△△△-△xxx	Optional	Specify the string to search for the result (standard output) of the input command. If the shell is specified, the strings will become the parameters during shell execution. Enumerate the criteria strings if there are multiple of them.
△△△△stdout_file:△xxx	Optional	The file to save the result (standard output) of the input command.
△△△△success_exit:△xxx	Optional	Please set this file to the shell parameter if the shell parameter if specified. Specify "yes" to exit the dialog normally if the search result is normal and specify no to proceed to the next. "no" is set on default.

Parameter	Required/ Optional	Description
△△△△ignore_errors:△ xxx ※△:Half-width space	Optional	Specify "yes" to proceed to the next even if the search result is abnormal. "no" is set on default.

Exp2-3)

cat the hosts file and grep the displayed results with parameter value. If there is line containing 139.0.0.1 or lalhost, the result is determined as normal and proceed to the next. If there is no such line, the result is determined as abnormal and the dialog file is terminated abnormally.

exec_list:

```
- state: 'cat /etc/hosts'
  prompt: 'root@{{ __loginhostname__ }}'
  parameter:
    - '139.0.0.1'
    - 'lalhost'
  - expect: root@{{ __loginhostname__ }}
  exec: exit
```

Exp2-4)

cat the hosts file and grep the displayed results with parameter value. If there is line containing 139.0.0.1 or lalhost, the result is determined as normal and terminate normally according to the success_exit:yes setting.

If there is no such line, the result is determined as abnormal and the dialog file is terminated abnormally

exec_list:

```
- state: 'cat /etc/hosts'
  prompt: 'root@{{ __loginhostname__ }}'
  parameter:
    - '139.0.0.1'
    - 'lalhost'
  success_exit: yes
  - expect: root@{{ __loginhostname__ }}
```

Exp2-5)

cat the hosts file and grep the displayed results with parameter value. If there is line containing 139.0.0.1 or lalhost, the result is determined as normal and proceed to the next. If there is no such line, the result is determined as abnormal and proceed to the next according to the ignore_errors:yes setting.

exec_list:

```
- state: cat /etc/hosts
  prompt: root@{{ __loginhostname__ }}
  parameter:
    - 139.0.0.1
    - lalhost
  ignore_errors: yes
  - expect: root@{{ __loginhostname__ }}
```

Exp2-6)

cat the hosts file and use the user created shell to grep the displayed results with parameter value. If there is line containing 139.0.0.1 or lalhost, the result is determined as normal and proceed to the next. If there is no such line, the result is determined as abnormal and the dialog file is terminated abnormally

exec_list:

```
- state: cat /etc/hosts
  prompt: root@{{ __loginhostname__ }}
  shell: /tmp/grep.sh
  stdout_file: /tmp/stdout.txt
  parameter:
    - 139.0.0.1
    - lalhost
```

User created shell(/tmp/grep.sh)

```
#!/bin/bash
STDOUT=/tmp/STDOUT.tmp
STDERR=/tmp/STDERR.tmp
cat /tmp/stdout.txt|grep $1|grep $2 | wc -l >${STDOUT} 2>${STDERR}
RET=$?
if [ $RET -ne 0 ]; then
  EXIT_CODE=$RET
else
  if [ -s ${STDERR} ]; then
    EXIT_CODE=1
  else
    CNT=`cat ${STDOUT}`
    if [ ${CNT} -eq 0 ]; then
      EXIT_CODE=1
    else
      EXIT_CODE=0
    fi
  fi
fi
```

Exp2-7)

cat the hosts file and save the displayed result to the file specified by stdout_file then proceed to the next. If the “no” parameter is set to the default shell, the result will be determined as abnormal.

Set ignore_errors:yes to proceed to the next. exec_list:

```
- state: cat /etc/hosts
  prompt: root@{{ __loginhostname__ }}
  stdout_file: {{ __symphony_workflowdir__ }}/hosts
  ignore_errors: yes
  - expect: root@{{ __loginhostname__ }}
  exec: exit
```

④ command module

Loops and conditional branching can be performed before and after inputting commands to the target host.

Command module format

Parameter	Required/ Optional	Description
$\Delta\Delta - \Delta\text{command}:\Delta\text{xxx}$	Required	Specify the input command.
$\Delta\Delta\Delta \Delta\text{prompt}:\Delta\text{xxx}$	Required	Specify the waiting prompt. It can be written in regular expression.
$\Delta\Delta\Delta \Delta\text{timeout}:\Delta\text{xxx}$	Optional	Specify the timer to wait for the prompt after the command is sent. If the parameter is omitted, then conf->timeout is used.
$\Delta\Delta\Delta \Delta\text{register}:\Delta\text{xxx}$	Optional	<p>Save the information of the standard output to any string specified after sending the command.</p> <p>When using with_items to loop, the information of the standard output after the last command input is saved. This variable can be used in condition judgment (can only be used in condition judgment). However, saving the information of standard output for every variable name is not possible. Previous information will be overwritten.</p> <pre> 1 exec_list: 2 - expect: 'assword' 3 exec: '{{ __loginpassword__ }}' 4 - command: 'systemctl status httpd' 5 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 6 register: httpd_status_register 7 - command: 'systemctl restart httpd' 8 when: 9 - httpd_status_register no match(running) 10 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 11 - command: 'systemctl status mysql' 12 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 13 register: mysql_status_register 14 - command: 'systemctl restart mysql' 15 when: 16 - mysql_status_register no match(running) 17 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 18 - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 19 exec: exit </pre> <p>Since the 6th line, httpd_status_register, has a different value set to the Register variable in the command module on line 11 (mysql_status_register), it is valid all up until line 10.</p>
$\Delta\Delta\Delta \Delta\text{with_items}:$ $\Delta\Delta\Delta \Delta\Delta - \Delta\{{\text{VAR}_x}\}$ $\Delta\Delta\Delta \Delta\Delta - \Delta\{{\text{VAR}_y}\}$ Enclose the defining variables in single quotation marks.	Optional	<p>If you're looping and inputting commands to with_items, configure multiple variable names.</p> <p>The scope of each variable is "item.X(X is 0-99).</p> <p>If you are using with_items with prompt, timeout, refer to the following for variable names</p> <pre> prompt: {{ΔVAR_prompt_XXXΔ}} timeout: {{ΔVAR_timeout_XXXΔ}} </pre>

Parameter	Required/ Optional	Description
		<p>(Δ =half-width space. XXX=any half-width alphabetic characters and underscore)</p> <p>If the number of variable's specific values set to with_items are not the same, it will loop at the maximum number of specific values.</p> <p>If there are any variables that does not have any specific values, they will have their specific value set to "blank".</p> <p>Additionally, if you're using with_items with Prompt or Timeout, pay attention to the numbers of specific values.</p> <p>prompt->command->prompt->command->prompt...(Loop)</p> <p>As it will loop like this, you will need to set 1 additional specific value (Number of commands+1).</p> <p>If there are any prompt or timeout variables that does not have specific values set, an error will occur when the operation is executed.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p>If you want to run the following command with Command module,</p> <ul style="list-style-type: none"> • systemctl start httpd • systemctl start mysql <p>The dialogue file and the specific values of the variables used in with_items are as follows.</p> <pre> - command: "systemctl {{ item.0 }} {{ item.1 }}" prompt: '{{ item.2 }}' timeout: '{{ item.3 }}' with_items: - '{{ VAR_status_list }}' # item.0 - '{{ VAR_service_list }}' # item.1 - '{{ VAR_prompt_list }}' # item.2 - '{{ VAR_timeout_list }}' # item.3 VAR_status_list: VAR_service_list: - start - httpd - start - mysql VAR_prompt_list: VAR_timeout_list: - Command prompt - 10 - Command prompt - 10 - Command prompt - 10 </pre> </div> <p>The variables defined with with_items can be scoped with item.X (other than register/when)</p> <p>Exp)</p> <pre> with_items: - '{{ VAR_item1 }}' #item.0 - '{{ VAR_item2 }}' #item.1 exec_when: - '{{ item.0 }} == active' - '{{ item.0 }} == {{ VAR_status }}' - 'register 変数 match({{ item.0 }})' </pre>

Parameter	Required/ Optional	Description						
		<p>failed_when:</p> <ul style="list-style-type: none"> — ‘stdout match('{{ item.1 }}’) 						
△△△△when: △△△△△△-△xxx △△△△△△-△xxx	Optional	<p>The condition judgement before command executes. Execute command if the condition matches. Move to the next “command” line if the condition doesn't match.</p> <p>Conditional expression</p> <p>Judging variable definition</p> <table> <tr> <td>VAR_xx is define</td> <td>Variable defined</td> <td>true</td> </tr> <tr> <td>VAR_xx is undefined</td> <td>Variable undefined</td> <td>true</td> </tr> </table> <p>Exp)</p> <ul style="list-style-type: none"> — ‘VAR_status is define’ — ‘VAR_status is undefined’ <p>※define/undefine can be specified only for ITA variable(VAR_xx)</p> <p>Judging variable specific value</p> <p>VAR_xx/register variable relational_operator string VAR_xx/register variable relational_operator VAR_xx VAR_xx/register variable match(Regular expression string /VAR_xx) VAR_xx/register variable no match(Regular expression string /VAR_xx)</p> <p>※Relational operators are 「==」、「!=」、「>」、「>=」、「<」、「<=」</p> <p>※The 「>」、「>=」、「<」、「<=」 relational operators are assumed to be used for numerical values.</p> <p>※It is not required to enclose string and regular expression string with single or double quotations.</p> <ul style="list-style-type: none"> — '{{ VAR_status }} match(active)' — '{{ VAR_status }} == active' — ‘register variable match(active)’ <p>Compound condition with and / or</p> <p>When processing with or condition, add OR between judge conditions</p> <p>Exp)</p> <ul style="list-style-type: none"> — '{{ VAR_status }} == 1 OR {{ VAR_status }} == 2' <p>When processing with and condition, write the statement in multiple lines.</p> <p>Exp)</p> <ul style="list-style-type: none"> — '{{ VAR_status }} == 1 OR {{ VAR_status }} == 2' — '{{ VAR_sub_status }} == 1' 	VAR_xx is define	Variable defined	true	VAR_xx is undefined	Variable undefined	true
VAR_xx is define	Variable defined	true						
VAR_xx is undefined	Variable undefined	true						
△△△△exec_when: △△△△△△-△xxx △△△△△△-△xxx	Optional	<p>Judge condition for every loop (continue condition)</p> <p>Perform condition judgement if with_items is writed.</p> <p>If the condition matches, execute command of the corresponding loop.</p> <p>If the condition doesn't match, move on to the next loop.</p> <p>Conditional expression</p> <p>Same format as “when:”</p>						
△△△△failed_when: △△△△△△-△xxx △△△△△△-△xxx	Optional	<p>Condition judgment for the stdout content after command execution(for every loop)</p> <p>Perform condition judgment even if with_items is not writed</p>						

Parameter	Required/ Optional	Description
※△: half-width space		<p>If the condition matches, the result is normal If the condition doesn't match, the result is abnormal and the dialog file is terminated abnormally Conditional expression Judging variable specific value stdout relational_operator string stdout relational_operator VAR_xx stdout match(regular expression string/VAR_xx) stdout no match(regular expression string/VAR_xx) ※Relational operators are 「==」、「!=」、「>」、「>=」、「<」、「<=」 ※The 「>」、「>=」、「<」、「<=」 relational operators are assumed to be used for numerical values. ※It is not required to enclose string and regular expression string with single or double quotations. VAR_status match(active) VAR_status == active Compound condition with and / or Same format as "when:"</p>

```

Exp3-1)
conf:
    timeout: 30

exec_list:
# If waiting for strings other than prompt is required, use the combination of expect/exec.
# In the case that the password is required.
    - expect: 'password:'
      exec:   '{{ __loginpassword__ }}'

# If the ITA variable, VAR_hosts_make, is writed in host variable file, cat the host file.
# If the variable is not writed, skip the command.
    - command: cat /etc/hosts
      prompt: root@{{ __loginhostname__ }}
      when:
        - VAR_hosts_make is define
    - expect: root@{{ __loginhostname__ }}
      exec: exit

```

```

Exp3-2)
conf:
    timeout: 30

exec_list:
# If waiting for strings other than prompt is required, use the combination of expect/exec.
# In the case that the password is required.
    - expect: 'password:'
        exec:  '{{ __loginpassword__ }}'

# If the ITA variable, VAR_hosts_make, is writed in host variable file, cat the host file.
# If the variable is not writed, skip the command."
# Use cat to save the contents of the standard output hosts file to result_stdout.
    - command: cat /etc/hosts
        prompt: root@{{ __loginhostname__ }}
        register: result_stdout
        when:
            - VAR_hosts_make is define

# If the ITA variable, VAR_hosts_make, is writed in host variable file, cat the host file.
# If the variable is not writed, skip the command.
# Execute the command for the numbers of the specific values of the multiple specific value
# variable set in the with_items.
# From the result of condition judgment for each loop, if "ip address host name" does not
# correspond to the hosts file, execute command.
# Add "IP_address host_name" to the last line of hosts file by using echo.
    - command: 'echo {{ item.0 }}  {{ item.1 }} >> /etc/hosts'
        prompt: 'root@{{ __loginhostname__ }}'
        when:
            - VAR_hosts_make is define
        with_items:
            - '{{ VAR_hosts_ip }}'      # item.0
            - '{{ VAR_hosts_name }}'   # item.1
        exec_when:
            - result_stdout no match('{{ item.0 }} *{{ item.1 }}')

    - expect: root@{{ __loginhostname__ }}
        exec: exit

```

```

Exp3-3)
conf:
    timeout: 30

exec_list:
# If waiting for strings other than prompt is required, use the combination of expect/exec.
# In the case that the password is required.
    - expect: 'password:'
      exec:   '{{ __loginpassword__ }}'

# Execute the command for the numbers of the specific values of the multiple specific value
# variable set in the with_items
# Execute auto startup configuration.
    - command: 'systemctl enable {{ item.0 }}'
      prompt: 'root@{{ __loginhostname__ }}'
      with_items:
        - '{{ VAR_service_name_list }}' # item.0

# Execute the command for the numbers of the specific values of the multiple specific value
# variable set in the with_items
# Execute service startup
    - command: 'systemctl start {{ item.0 }}'
      prompt: 'root@{{ __loginhostname__ }}'
      with_items:
        - '{{ VAR_service_name_list }}' # item.0

Execute the command for the numbers of the specific values of the multiple specific value
variable set in the with_items.
Output the service status to standard output.
If the content of result output to standard output contains the regular expression of item.1, the
result is right.
For example, in the case that the specific value of VAR_service_status_list is set to running and
the service is running, "running" in "Active: active(running)" matches so the result is right. (Move
on to the next loop)
In the case that condition doesn't match, the result is determined as abnormal and the dialog
file terminates abnormally.
    - command: 'systemctl status {{ item.0 }}'
      prompt: 'root@{{ __loginhostname__ }}'
      with_items:
        - '{{ VAR_service_name_list }}' # item.0
        - '{{ VAR_service_status_list }}' # item.1
      failed_when:
        - stdout match('{{ item.1 }}')

    - expect: root@{{ __loginhostname__ }}
      exec: exit

```

```

Exp3-4)
conf:
    timeout: 30

exec_list:
# If waiting for strings other than prompt is required, use the combination of expect/exec.
# In the case that the password is required.
- expect: 'password:'
  exec:   '{{ __loginpassword__ }}'

# Execute the command for the numbers of the specific values of the multiple specific value
# variable set in the with_items.
# When describing the command with "{{item.0}}" only, enclose it with double-quotation.
# Please note the numbers of specific value when using with_items in prompt or timeout.
# prompt→command→prompt→command→prompt…(loops thereafter), it is required to plus 1
# to the command count. (Same for timeout)
- command: "{{ item.0 }}"
  prompt: '{{ item.1 }}'
  timeout: '{{ item.2 }}'
  with_items:
    - '{{ VAR_command_list }}' # item.0
    - '{{ VAR_prompt_list }}' # item.1
    - '{{ VAR_timeout_list }}' # item.2
- expect: root@{{ __loginhostname__ }}
  exec: exit

```

Variable name	Specific value
VAR_command_list	systemctl status {{ item.1 }}
VAR_service_name_list	ky_pioneer_execute-workflow.service

exec_list:

If waiting for strings other than prompt is required, use the combination of expect/exec.

In the case that the password is required.

- expect: 'password:'

```
exec: '{{ __loginpassword__ }}'
```

Execute the command for the numbers of the specific values of the multiple specific value
variable set in the with_items.

Setting the specific value of substitution value list to {{item.X}} is possible. In that case, please
make the numerical value writed in the specific value of item.X which is writed in the
dialog file grows.

The command executed in this example is "systemctl status ky_pioneer_execute-workflow.service".

- command: "{{ item.0 }}"

```
prompt: 'root@{{ __loginhostname__ }}'
```

with_items:

- '{{ VAR_command_list }}' # item.0

- '{{ VAR_service_name_list }}' # item.1

- expect: root@{{ __loginhostname__ }}

```
exec: exit
```

```

Exp6)
conf:
    timeout: 30

exec_list:
# If waiting for strings other than prompt is required, use the combination of expect/exec.
# In the case that the password is required.
    - expect: 'password:'
      exec:   '{{ __loginpassword__ }}'

# Example of compound condition using and / or.
# When processing with or condition, write the if statement horizontally.
# When processing with and condition, describing the statement in multiple lines.
# "when" is used as the example here but the same applies to exec_when and failed_when.
    - command: echo aaa
      prompt: 'root@{{ __loginhostname__ }}'
      when:
        - 10 == 9 OR 10 != 9 OR 10 >= 9
        - 10 > 9 OR 10 <= 9 OR 10 < 9

    - expect: root@{{ __loginhostname__ }} and condition
      exec: exit

```

⑤ localaction module

Execute command on Ansible/Ansible Automation Controller server.

Localaction module format

Parameter	Required/ Optional	Description
$\Delta\Delta - \Delta\text{localaction}:\Delta\text{xxx}$	Required	Specify the command to be executed.
$\Delta\Delta\Delta\Delta\text{ignore_errors}:\Delta\text{xxx}$ ※ Δ :Half-width space	Optional	Specify “yes” to continue if the execution result of the command is abnormal. If “no” is specified, the dialog will end if the result of execution is abnormal. Default is “no”.

Exp4-1)

During Symphony execution, create a directory to output the hosts file for every host in the shared directory (`__symphony_workflowdir__`) of each Movement.

exec_list:

```
- localaction: mkdir -p 755 {{ __symphony_workflowdir__ }}/{{ __loginhostname__ }}  
ignore_errors: yes  
- state: cat /etc/hosts  
prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}'  
stdout_file: {{ __symphony_workflowdir__ }}/{{ __loginhostname__ }}/hosts  
ignore_errors: yes  
- expect: root@{{ __loginhostname__ }}  
exec: exit
```

(3) Regular expression

The strings writed in the following command and parameter are evaluated in regular expression.

- expect module
- The prompt parameter of state module
- The prompt parameter of command module

When the string is writed in regular expression contains metacharacter "(){}.", etc., inserting escape character "\ \$" before metacharacters is required.

Exp1)

When waiting for the following command, the red characters are metacharacters.

XAMPP Developer Files [Y/n] exec_list:

Inserting escape character "\ \$" before metacharacters is required.

XAMPP Developer Files \\${Y\\$\\$n\\$} exec_list:

State module and command module extracts the result (standard ouput) of the executed command. The notes of the extraction are as follows.

- ① The delimitation between the result (standard ouput) of the executed command and the prompt.

The delimitation between the result (standard ouput) of the executed command and the prompt is performed by the string specified in the prompt parameter. When judging the result of the executed command (standard output) or saving it to a file, please do not write a preceding match with .* in regular expression. The result (standard ouput) of the executed command can not be extracted.

.* Example of preceding match with .* in regular expression.

'.*[\\$\\$\\$\\$\\$\\$] \\$'

- ② Support of escape sequence

Depending on the target device, an Operating System Command sequence may be added immediately before the prompt sent from the target device.

Escape sequences immediately before the string specified by the prompt parameter are excluded.

(4) Notes when using multiple specific value variable

The only parameter in the dialog file that can use multiple specific value variable is the with_items parameter of command module. If multiple specific value variable is used in other cases, the operation execution will turn out to be error.

(5) Things to be aware of when processing prompts other than command prompts.

If you want to process prompts other than command prompts, combine exec module and expect module and create a dialogue file.

Command and State modules cannot be processed.

Exp)	Variable	Specific Value
conf:	VAR_id_rsa_path	Set file path of the secret key
timeout: 10	VAR_passphrase	Set passphrase
exec_list:		If you want it to return If you are using variables, leave the specific values blank. If you are not using variables, input an empty string (with two quotations)..
# ssh connection Password authentication - expect: 'assword:' exec: '{{ __loginpassword__ }}'		exec: ''
# ssh-keygen command execution - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}' exec: ssh-keygen		
# The following is the process for prompts other than the command prompt. # Set file path of the secret key # Since expect is evaluated in regular notation, the escape character (\\$) must be inserted for meta characters that need to be escaped. - expect: 'id_rsa\\$': exec: '{{ VAR_id_rsa_path }}'		
# Set passphrase - expect: ' passphrase\\$': exec: '{{ VAR_passphrase }}'		
# Confirm passphrase - expect: ' passphrase again:' exec: '{{ VAR_passphrase }}'		
# Confirm the created secret key file. - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}' exec: 'ls -al {{ VAR_id_rsa_path }}'		
# Close ssh connection - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}' exec: exit		

(6) Things to be aware of when ending dialogue files.

Make sure to input a command that ends the session at the end of the dialogue file.

Ending the last line of the module closes the session.

If the final line is a file copy or any other process that takes time, the session will close before the command is finished and may end up ending abnormally.

```
Exp)
conf:
    timeout: 10

exec_list:
    # ssh connection password authentication
    - expect: 'assword:'
      exec: ' {{ __loginpassword__ }}'

    # File copy
    - expect: ' {{ __loginuser__ }}@{{ __loginhostname__ }}'
      exec: 'cp -rfp {{ VAR_src_path }} {{ VAR_dest_path }}'

    # Write a line that waits for the previous command to end in the command prompt and inputs
    # an exit command at the end of the dialogue file.
    - expect: ' {{ __loginuser__ }}@{{ __loginhostname__ }}'
      exec: exit
```

(7) Things to keep in mind when writing dialogue files in yaml format.

Dialogue files are treated as yaml format files. If there are lines or descriptions that do not follow the YAML format, an error will occur when uploading the dialogue module or when executing the operation. See example below.

- If a variable is written in the parameter of each module and the whole parameter is not enclosed in quotation marks.
- If the parameters are only written in constants, the constant doesn't end in ":" , or other cases where parameters are not enclosed in quotations.

We recommend to enclose all the module parameters in quotation marks.

Example when not written in YAML Format (Red text)

```
- expect: assword:  
  exec: {{ __loginpassword__ }}  
- expect: {{ __loginuser__ }}@{{ __loginhostname__ }}  
  exec: ls  
- command: echo {{ item.0 }}  
  prompt: {{ __loginuser__ }}@{{ __loginhostname__ }}  
  exec_when:  
    - {{ item.1 }} = run  
  with_items:  
    - {{ VAR_echo }}  
    - {{ VAR_exec_when }}  
    - {{ VAR_failed_when }}  
failed_when:  
  - stdout == match('{{ item.2 }}')  
- state: {{ VAR_command }}  
  prompt: {{ __loginuser__ }}@{{ __loginhostname__ }}  
parameter:  
  - {{ VAR_p1 }}  
  - {{ VAR_p2 }}
```

Closing the parameters when the description is not written in YAML format.

```
- expect: 'assword':  
  exec: '{{ __loginpassword__ }}'  
- expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'  
  exec: 'ls'  
- command: 'echo {{ item.0 }}'  
  prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}'  
  exec_when:  
    - '{{ item.1 }} = run'  
  with_items:  
    - '{{ VAR_echo }}'  
    - '{{ VAR_exec_when }}'  
    - '{{ VAR_failed_when }}'  
failed_when:  
  - stdout == match('{{ item.2 }}')  
- state: '{{ VAR_command }}'  
  prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}'  
  parameter:  
    - '{{ VAR_p1 }}'  
    - '{{ VAR_p2 }}'  
  success_exit: 'yes'
```

Variables listed in
with_items should be
enclosed in single quotes.

(8) LANG for the construction device's login user

The Login user's "LANG" supports the following: UTF-8/euc/shift_jis.

The Login user's "LANG" settings must be configured from the device list.

If "euc" or "shift_jis" is used, the dialogue file may not be processed correctly due to the characteristics of the decoding process to UTF-8 of the pexpect module used in the communication control with the device.

• Some full-width characters(such as "①②" cannot be used with the UTF-8 decoder. Characters that are not decodable will be displayed as "??".

(3) • When some full-width characters (such as " - , etc") are used in prompt waiting,

The waiting will end successfully if the LANG is "UTF-8". If it is "euc/shift_jis" the wait will time out.

(9) Termination codes for commands input to the Construction device.

The Command's termination codes is "LF".

If it is "CRLF", add "/r" to the end of the command.

```
conf:  
    timeout: 10  
exec_list:  
    - expect: 'password:'  
      exec:  'XXXXXXXX\r'  
    - command: ' {{ VAR_command }}\r'  
      prompt: ' {{ __loginuser__ }}@{{ __loginhostname__ }}'  
    - state:  ' {{ VAR_state }}\r'  
      prompt: ' {{ __loginuser__ }}@{{ __loginhostname__ }}'  
      parameter:  
        - ' {{ VAR_parameter1 }}'  
        - ' {{ VAR_parameter2 }}'
```

6.3 Write role package (Ansible-Legacy Role)

Please refer to the Ansible best practices official manual for the basic format.

Please use UTF-8 for the character encoding.

This section writes the directory that is required to be in the zip file of role package file uploaded in "5.3.4 Role package list (Ansible-Legacy Role only)" and the process in ITA.

(Parent directory)	
---site.yml	site.yml(master Playbook) is created in ITA. The file will be overwritten if exists.
---hosts	hosts file is created in ITA. The file will be overwritten if exists.
---group_vars	host group variables are not handled. The group_vars directory will be removed if exists.
---host_vars	Host variables are created in ITA. This directory will be overwritten if it exists.
---ITA readme	ITA readme is defined for every role. Error doesn't occur even if the file doesn't exist The naming rule of ITA readme file name: ita_readme_[role_name].yml e.g.) Role name: mysql File name: ita_readme_mysql.yml Role name: mysql/install File name: ita_readme_mysql%install.yml ※If role is in a deep directory hierarchy, replacing "/" in the role name with "%" is required.
---translation table	Translation table is defined for every role. Error doesn't occur even if the file doesn't exist The naming rule of translation table file name ita_translation-table_[role_name].txt e.g.) Role name: mysql File name: ita_translation-table_mysql.txt Role name: mysql/install File name: ita_translation-table_mysql%install.txt ※If role is in a deep directory hierarchy, replacing "/" in the role name with "%" is required.
---ITA does not concern if other directory or file exists other the directory or file above.	
---roles	Error occur during upload if the roles directory does not exist.
---[role name①]	Error occur during upload if the role name directory does not exist.

		Handle the directory containing tasks directory as a role.
		Errors doesn't occur even if the directory hierarchy is deep.
	└ readme.md	ITA does not concern with the file.
	└ tasks	The tasks directory is required.
	└ main.yml	Error occur during upload if main.yml does not exist.
	└ user_files	Files other than main.yml can be placed here.
	└ user.yml	Files other than main.yml can be placed in subdirectory.
	└ handlers	Doesn't concern if handlers directory exists or not.
	└ main.yml	Doesn't concern if main.yml exists or not
	└ user_files	Files other than main.yml can be placed here.
	└ user.yml	Files other than main.yml can be placed in subdirectory.
	└ templates	Doesn't concern if templates directory exists or not.
	└ hosts.j2	Files can be placed in subdirectory.
	└ user_files	
	└ user.j2	
	└ files	Doesn't concern if files directory exists or not.
	└ sudoers	Doesn't concern if file and subdirectory exists or not.
		The file content is not concerned.
	└ vars	Doesn't concern if vars directory exists or not.
	└ main.yml	Doesn't concern if file and subdirectory exists or not.
		The file content is not concerned.
	└ defaults	Doesn't concern if defaults directory exists or not.
	└ main.yml	Doesn't concern if main.yml exists or not.
	└ user_files	Files other than main.yml can be placed here.
	└ user.yml	Files other than main.yml can be placed in subdirectory.
	└ meta	Doesn't concern if meta directory exists or not.
	└ main.yml	Doesn't concern if file and subdirectory exists or not.
		The file content is not concerned.
	└ [role name②]	ITA does not concern if other directory or file exists other the directory or file above.
	└ [role name②]	There is no specific limit on the number of roles.

(1) Master playbook

The master Playbook you create in ITA consists of a header section and a roles section.

① header section

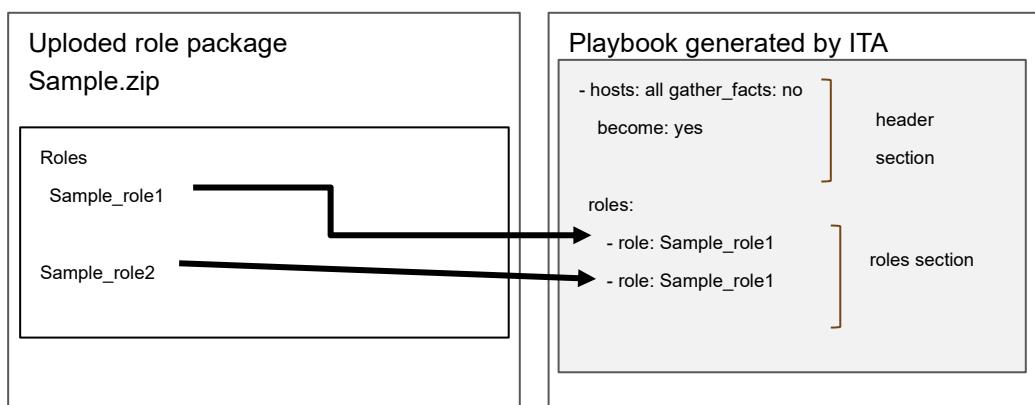
The default value of the header section is fixed, but you can change it in the header section of ["5.3.2. Movement list"](#).

Defult value of vector section

• For Ansible engine	• For AnsibleTower
- hosts: all	- hosts: all
remote_user: "{{ __loginuser__ }} "	gather_facts: no
gather_facts: no	become: yes
become: yes	

② roles section

Execute the roles in the uploaded role package in the role according to the included order in ["5.3.7Movement details"](#).



(2) Unique management of variable name

The variable information registered in the substitution value list of ITA is handled as host variable.

Variable names in all role packages of each drivers are uniquely managed.

When using same the variable name between roles with different variable structure, error will occur during upload

For example, in the case that the normal variable and nested variable or the nested structure is different between nested variables, etc.

(3) ITA original specification of default variable definition file (defaults-> main.yml)

There is ITA original specification of the description (variable definition) of default variable definition.

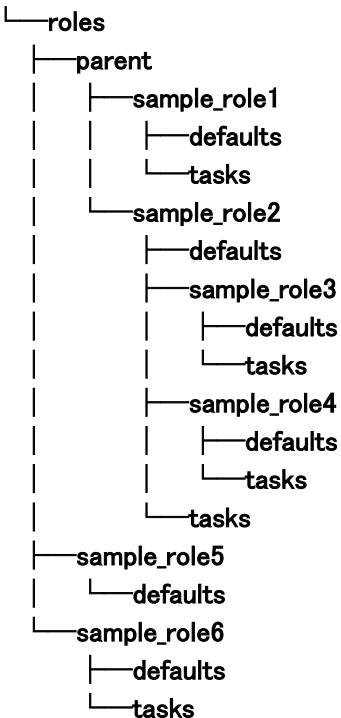
Please refer to the attachment file "User Instruction Manual - Ansible-driver attachment Ansible usage guideline with additional rules" for details.

(4) Notes on subdirectories of a predetermined directory in Ansible Role Directory Structure

If a subdirectory that has the same name of the predetermined directory is created under the predetermined directory in Ansible Role Directory structure (For example, a "files" directory under the "files" directory, etc.), error will occur during operation execution.

- (5) Points to note when the role name in the role package is set to the directory hierarchy.

The following directory hierarchy role package will be explained as an example.



- ① The directory recognized as a role is the directory containing the tasks directory.

In this example. There are three directory hierarchies (role names) to be handled by roles.

In this example. There are three directory hierarchies (role names) to be handled by roles.

- parent/sample_role1
- parent/sample_role2
- sample_role6

- ② Exclude directory hierarchies with multiple tasks directories

There are tasks directories in parent/sample_role2/sample_role3 and

parent/sample_role2/sample_role4, but parent/sample_role2 has a tasks directory and
recognizes it as a role, so it is not handled as a role.

6.4 Write ITA readme (Ansible-Legacy Role only)

The substitution value management function interprets the variable type defined in defaults variable definition file and sets the variable value of each variable and its' member variable.

In the cases such as not wanting to define variable directly in the Playbook, etc. if variable is not defined in defaults variable definition file, variable value can be specified in the substitution value management function by setting the variable definition in ITA readme file.

(1) Naming rule of file name of ITA readme

ita_readme_[role name].yml

e.g.)

Role name: mysql File name: ita_readme_mysql.yml

Role name: mysql/install File name: ita_readme_mysql%install.yml

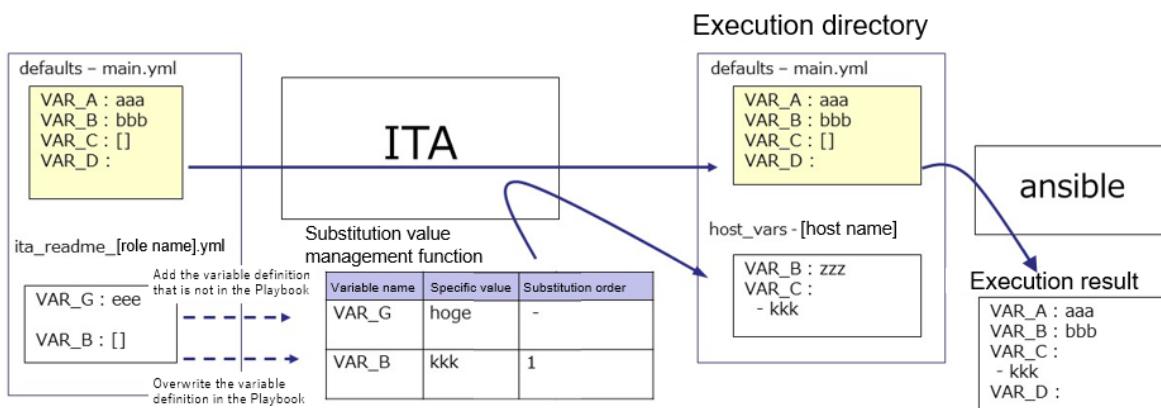
※If the role's directory hierarchy is deep, it is necessary to replace "/" in the role name with "%".

(2) The format of translation table

The format is YAML format.

Make sure the character code is UTF-8 without BOM.

The relation between ITA readme file and substitution value management function is as the following figure.



If the variable that is not in Playbook is defined in ITA readme file, the value of defined variable can be set in the substitution value management function.

Furthermore, if a type different from the variable in the Playbook is defined in the ITA readme file, the value can be registered in the substitution value management function with the overwritten variable type.

The value set in the substitution value management function is output to the variable definition file (**host_vars**) of each host, and is executed on each host by using the original Playbook and variable definition file as input in Ansible.

ITA readme file is only used to provide variable definition to substitution value management function, the variable and variable value defined in ITA readme don't affect the execution of Ansible.

It is optional to create ITA readme. If the variable definition in ITA readme and defaults variable definition file overlaps, the following rules will be used to handle the situation.

Table 6.4-1 variable adoption rule

defaults variable definition file	ITA readme	Adoption source of variable definition
defined	undefined	defaults variable definition file
undefined	defined	ITA readme
defined	defined	ITA readme

In addition, the default value displayed in "["5.3.11 Substitution value list"](#)" is processed following the rule below.

Table 6.4-2 default value display rule

defaults variable definition file	ITA readme	The method to handle the default value
defined	undefined	Adopt the defaults variable definition file.
undefined	defined	Handle as no default value.
defined	defined	Adopt the defaults variable definition file. However, the rule is applied only when the definition variable matches. If the variable definition doesn't match, the variable is handled as no default value.

ITA readme is separated from role package during work execution.

The variable and specific value registered in ITA readme can't be applied.

6.5 Write translation table (Ansible-Legacy Role only)

The translation table is a file set for making setting the specific value of variable other than "VAR_xxx" defined in defaults variable definition file or ITA readme in "5.3.11 Substitution value list" possible.

Define the link between the "arbitrary variable" defined in the default variable definition file or ITA readme with the "substitution variable" handled in substitution management function.

(1) Naming rule of file name of ITA readme

ita_readme_[role name].txt

e.g.)

Role name: mysql File name: ita_translation-table_mysql.txt

Role name: mysql/install File name: ita_translation-table_mysql%install.txt

※If the role's directory hierarchy is deep, it is necessary to replace "/" in the role name with "%".

(2) The format of translation table

The format is as follows in text format.

Make sure the character code is UTF-8 without BOM

The combination of substitution variable and arbitrary variable has to be unique within single role.

Substitution variable(\$s*):(\$s+)arbitrary variable

substitution variable:LCA_***

***:half-width alphanumeric character and underscore(_) can be used.

(Minimum length: 1byte, Maximum length: 256 bytes)

Arbitrary variable: (Minimum length: 1byte, Maximum length: 256 bytes)

(\$s*): More than 0 half-width space

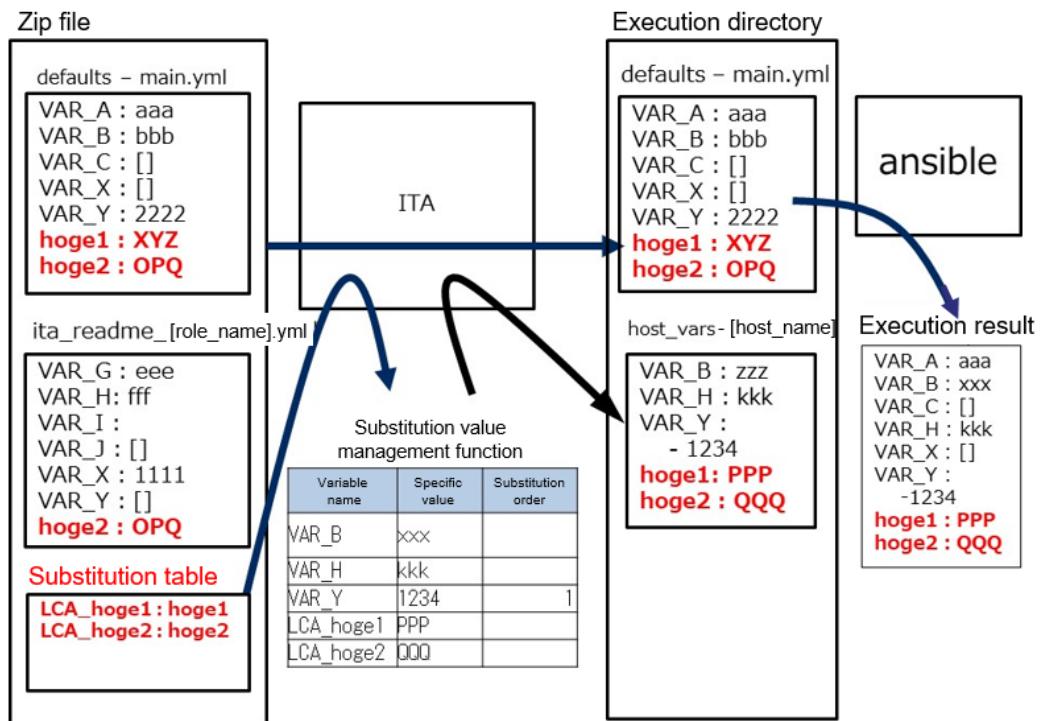
(\$s+): More than 1 half-width space

e.g.)

LCA_var1: var1

The line starts with # is comment line

The relationship of substitution value management function is as the following figure.

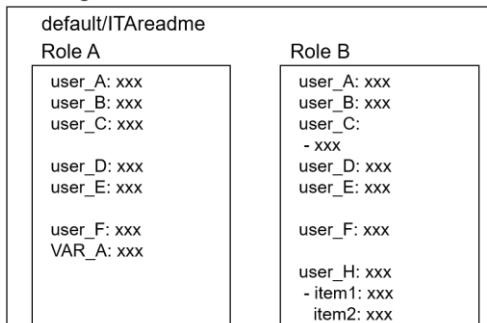


(2) Notes

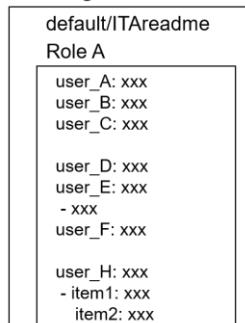
Listing the notes when creating translation table.

Case	ITA behaviour	Remarks
The translation table exists but defaults variable definition file and ITA readme doesn't exist. (For every role)	Translation table can't be read.	
The variable begins with VAR_ is defined as arbitrary variable	Error occurs when uploading role package.	
Uses any variables not defined in the defaults variable definition file and ITA readme	Error occurs when uploading role package.	
Definition of substitution variables are duplicated in the role	Error occurs when uploading role package.	Package A->Role A LCA_A: user_A/LCA_A: user_B
Definition of arbitrary variables are duplicated in the role	Error occurs when uploading role package.	Package A->Role B LCA_A: user_A/LCA_B: user_A
The structure of arbitrary variable differs between roles	Error occurs when uploading role package.	Package A->Role A/B LCA_C: user_C
The combination of substitution variable and arbitrary variable is not unique in role package	Error occurs when uploading role package.	Package A Role A LCA_D: user_D Role B LCA_D: user_E
The structure of arbitrary variable differs between role packages	Error doesn't occurs when uploading role package, but the substiution variable is not displayed in substitution value list.	Package A->Role A LCA_F: user_F Package B->Role A LCA_F: user_F
Nested arbitrary variable is defined between role packages	The nested structure is the same so error does not occur but the setting of nested iteration count is common setting for each package.	Package A->Role B LCA_H: user_H Package B->Role A LCA_H: user_H

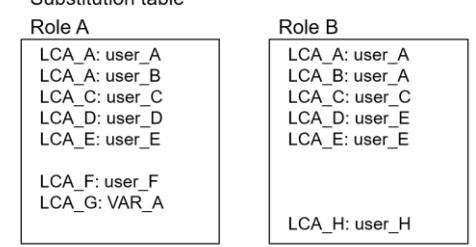
Package A



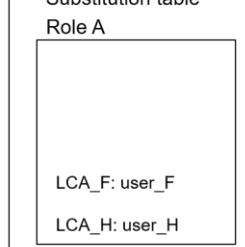
Package B



Substitution table



Substitution table



6.6 ita_readme file and translation table (Ansible-Legacy Role only)

This chapter lists 9 different ita_readme and translation table usecases.

This section presumes that the Ansible-Legacy Role (roles directory) is acquired from an external source.

The following figure illustrates the process of using the ita_readme file and translation table to upload and checking the results.

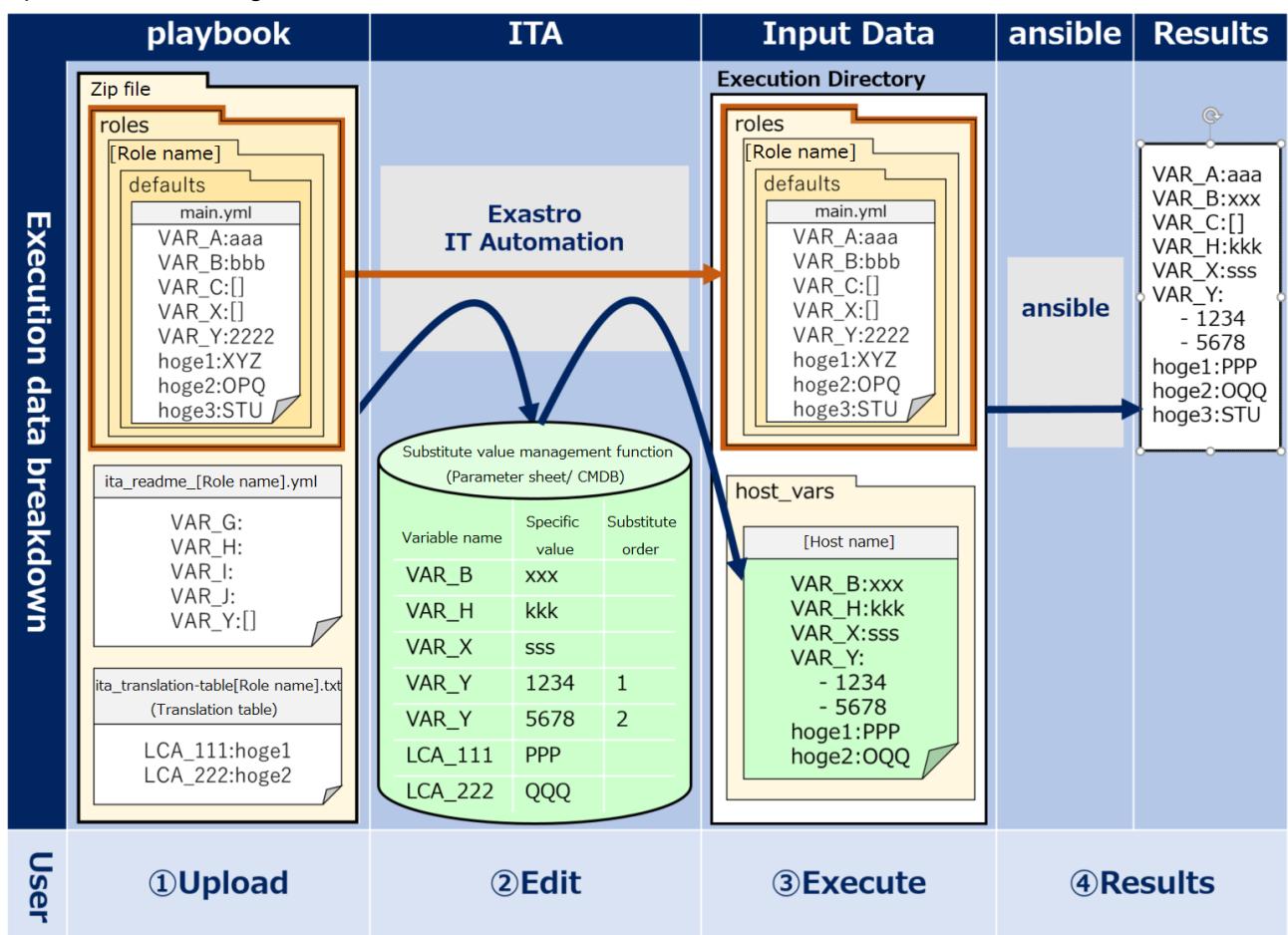


Figure 6.6-1 Overview image

The 9 usecases introduced in this chapter uses the figure above as a base.

No.	Usecase
1	Using Ansible-Legacy Role without modifying it.
2	ita_readme and translation table role
3	Variable definitions and default values described in the "defaults/main.yml" file.
4	"host_vars" files and ITA/CMDB
5	Adding variables to "defaults/main.yml"
6	"VAR_" Prefix
7	Linking "ita_readme" and translation table
8	Applying Playbook Length evaluation
9	Applying Playbook Defined evaluation

- **Case 1. Using Ansible-Legacy Role without modifying it**

Users can use Ansible-Legacy Role (roles directory) acquired from an external source without modifying it.

Therefore, users can put the `ita_readme` file and/or substitute table in the “roles” directory and assign parameters to the variables used inside the directory.

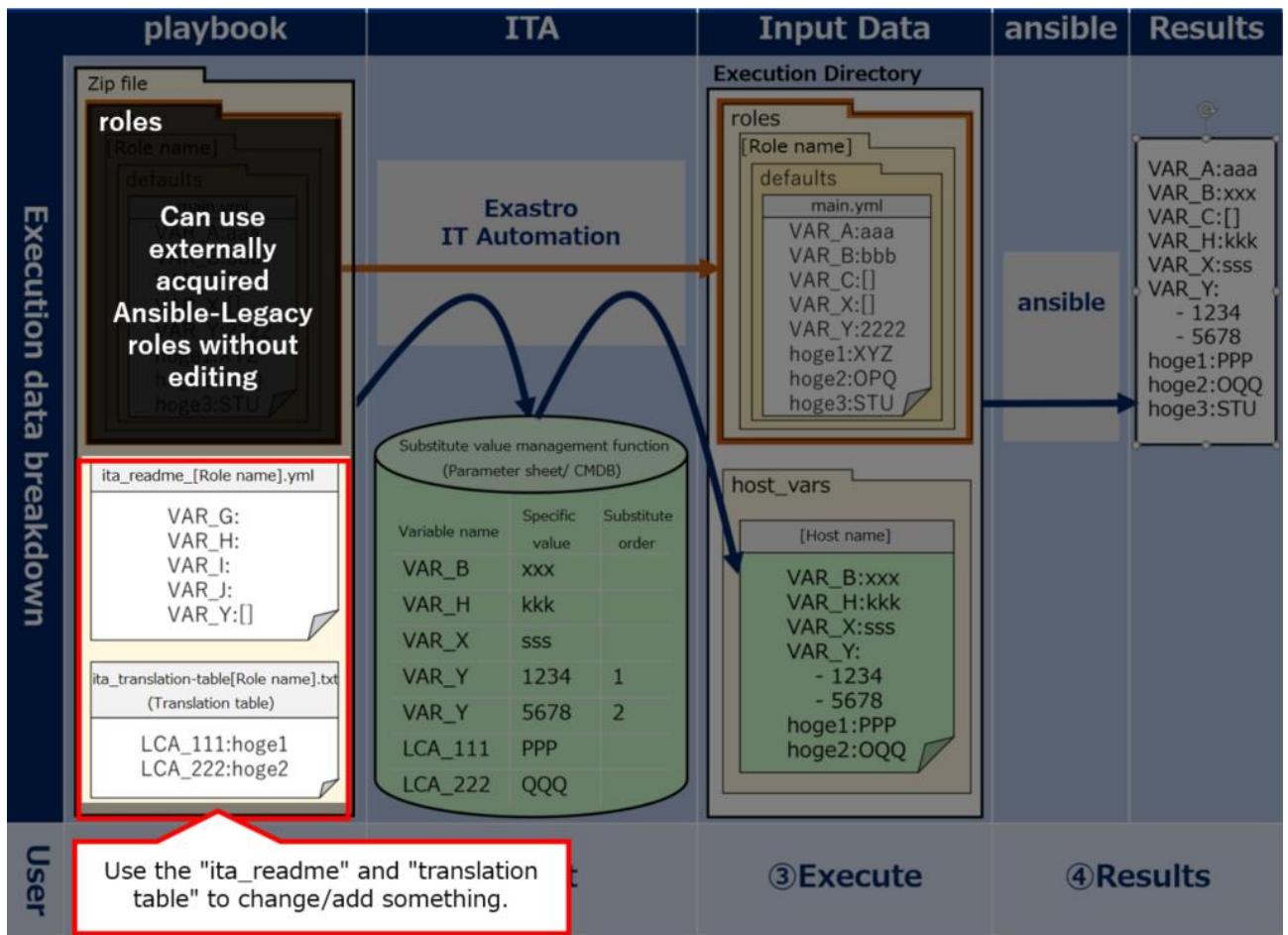


Figure 6.6-2 Case 1

- **Case 2. ita_readme and translation table role**

Both the `ita_readme` file and translation table are used to send variables/variable types to ITA.
In other words, they are not used to define specific values (Parameters).
ITA will not be able to read any specific values written in them.

Please see the other cases below for information on how to assign specific values.

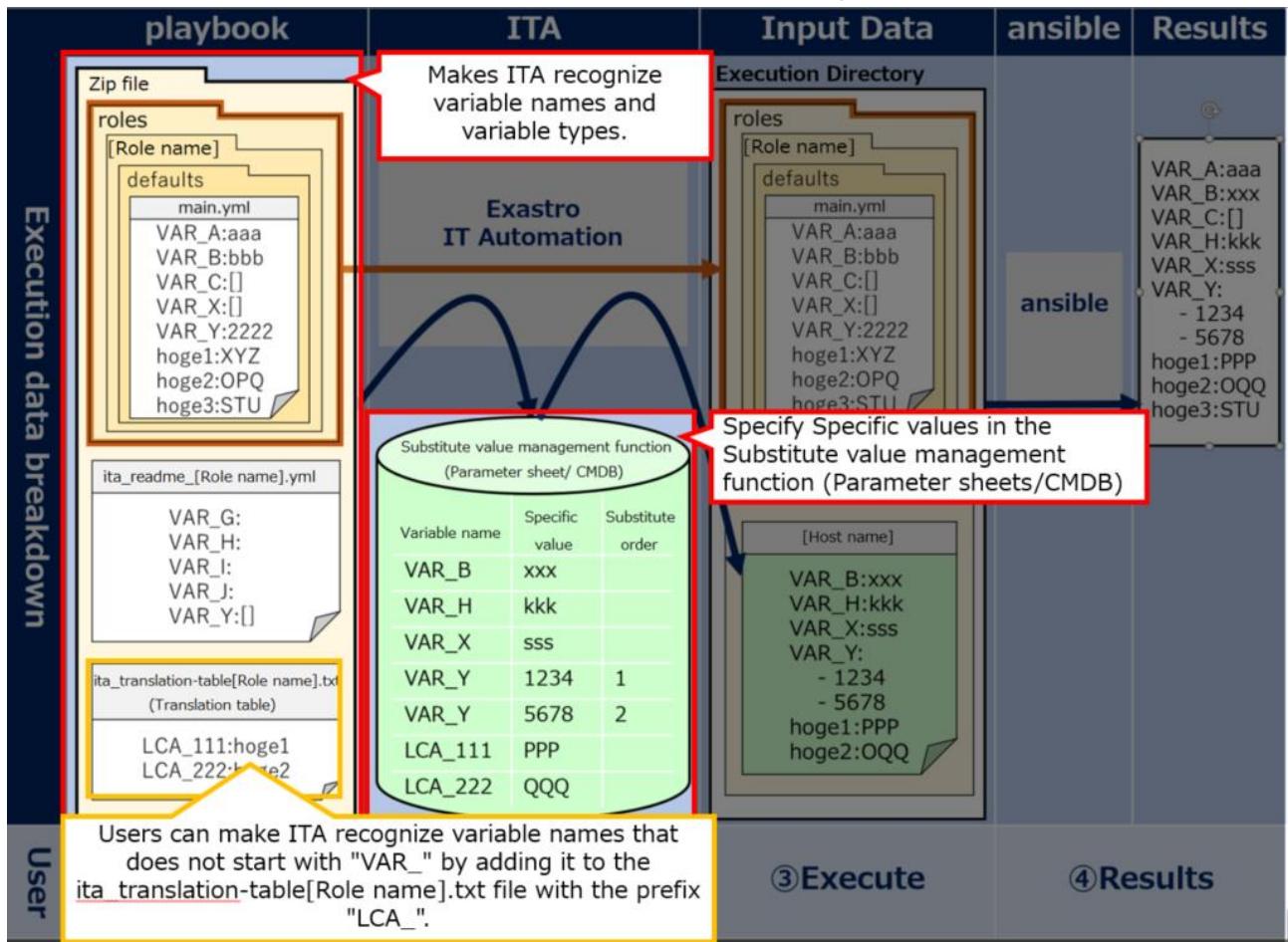


Figure 6.6-3 Case 2

- Case 3. Variable definitions and default values described in the "defaults/main.yml" file.

The "defaults/main.yml" file stored under "roles" is automatically passed to ansible.

The file will be automatically sent as long only if no variables or default values are defined in host_vars.(E.g: "VAR_A:aaa").

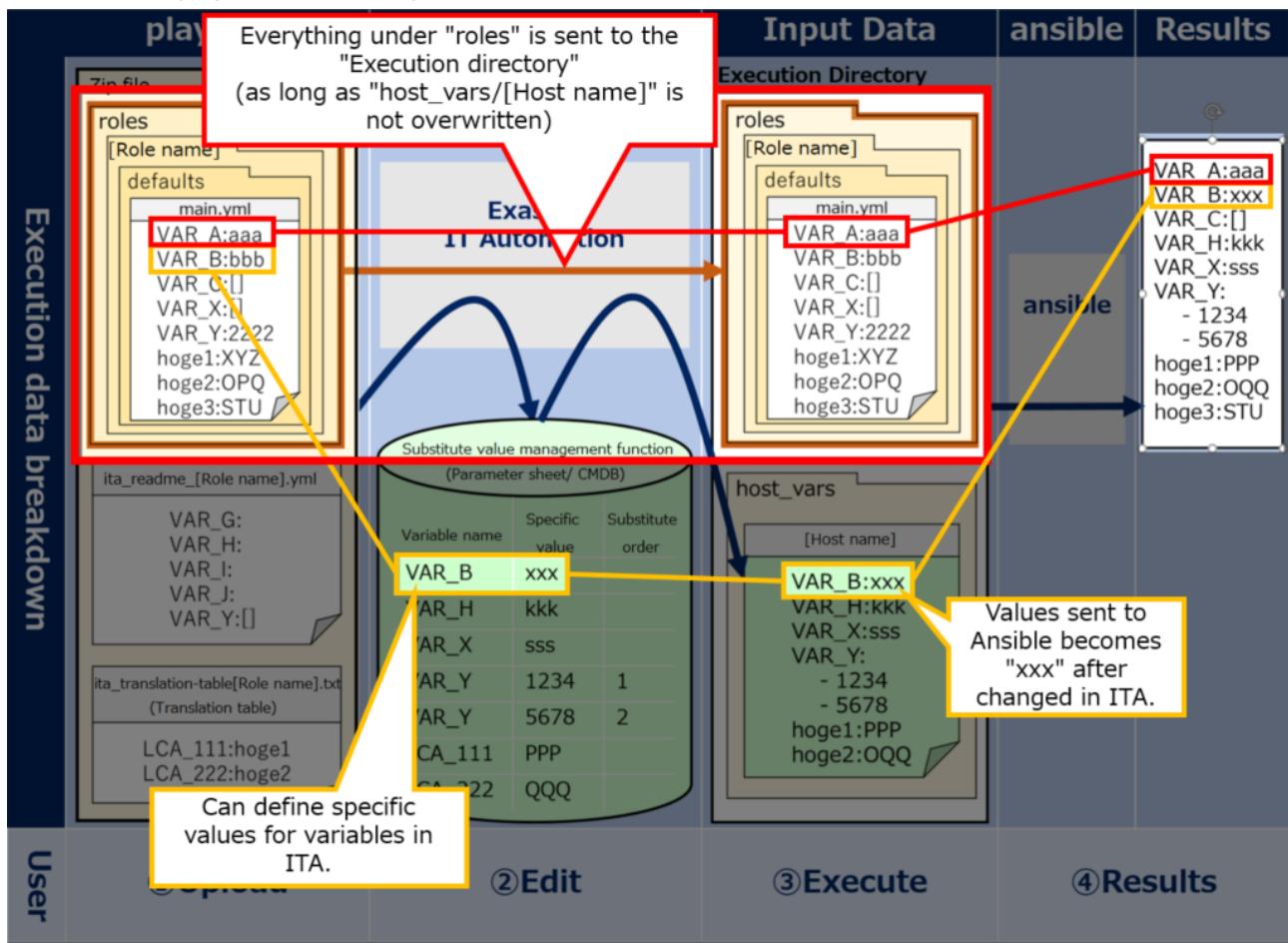


Figure 6.6-4 Case 3

- **Case 4. "host_vars" files and ITA/CMDB**

Host_vars files are automatically created everytime ITA's CMDB (parameter sheet) executes something.

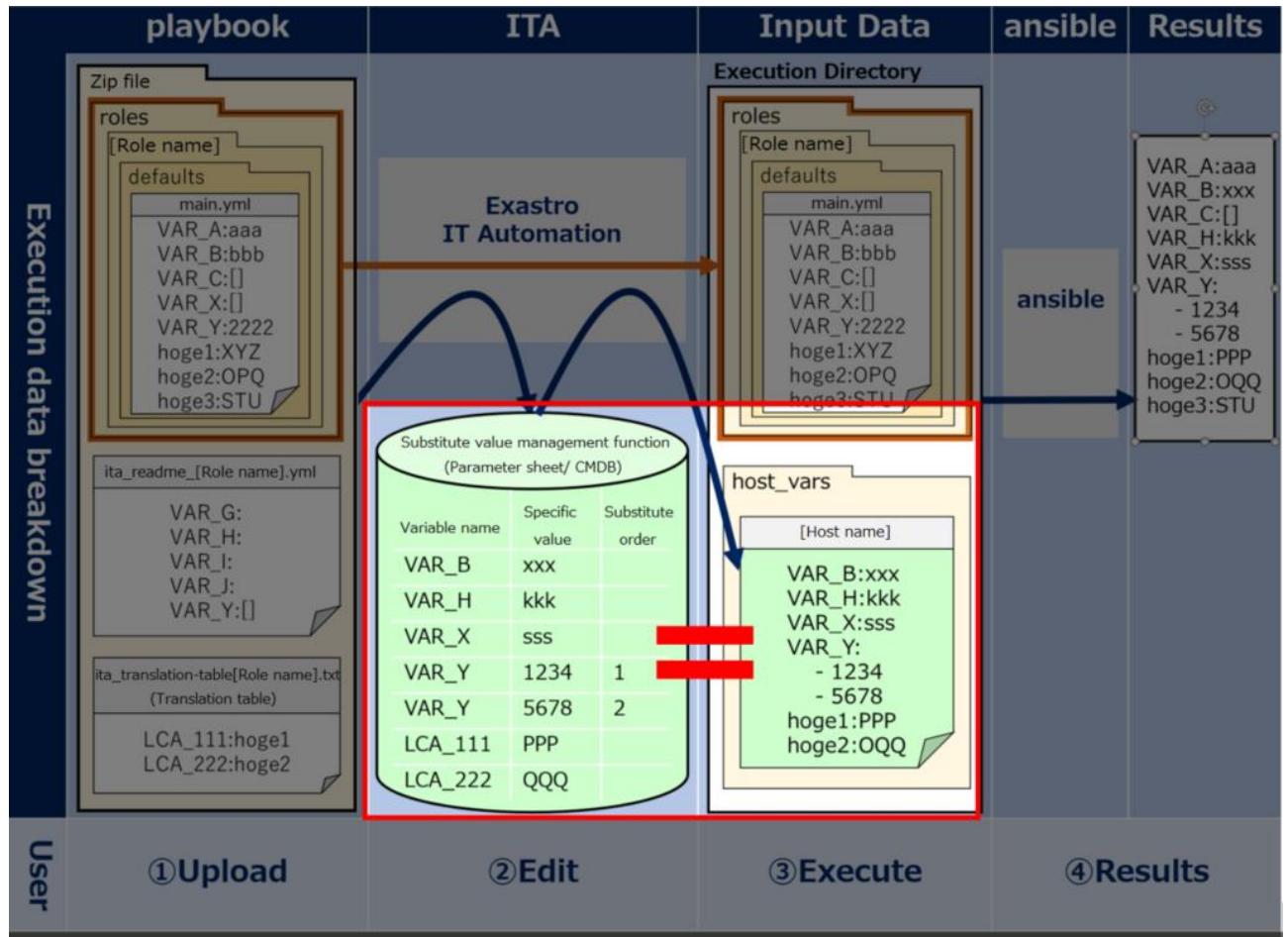


Figure 6.6-5 Case 4

- **Case 5. Adding variables to “defaults/main.yml”**

If you want to add any changes to Ansible-Legacy Role ("roles" directory), users can describe variable names/types in the "ita_readme" file.

Users do not have to define any variables in the ita_readme file that are already defined in the "defaults/main.yml" file.

If there are different definitions for the same variables in the files, the ones in the "ita_readme" file will be prioritized.

※The figure below illustrates that it is possible to add variables by describing a variable(VAR_H) in the ita_readme file

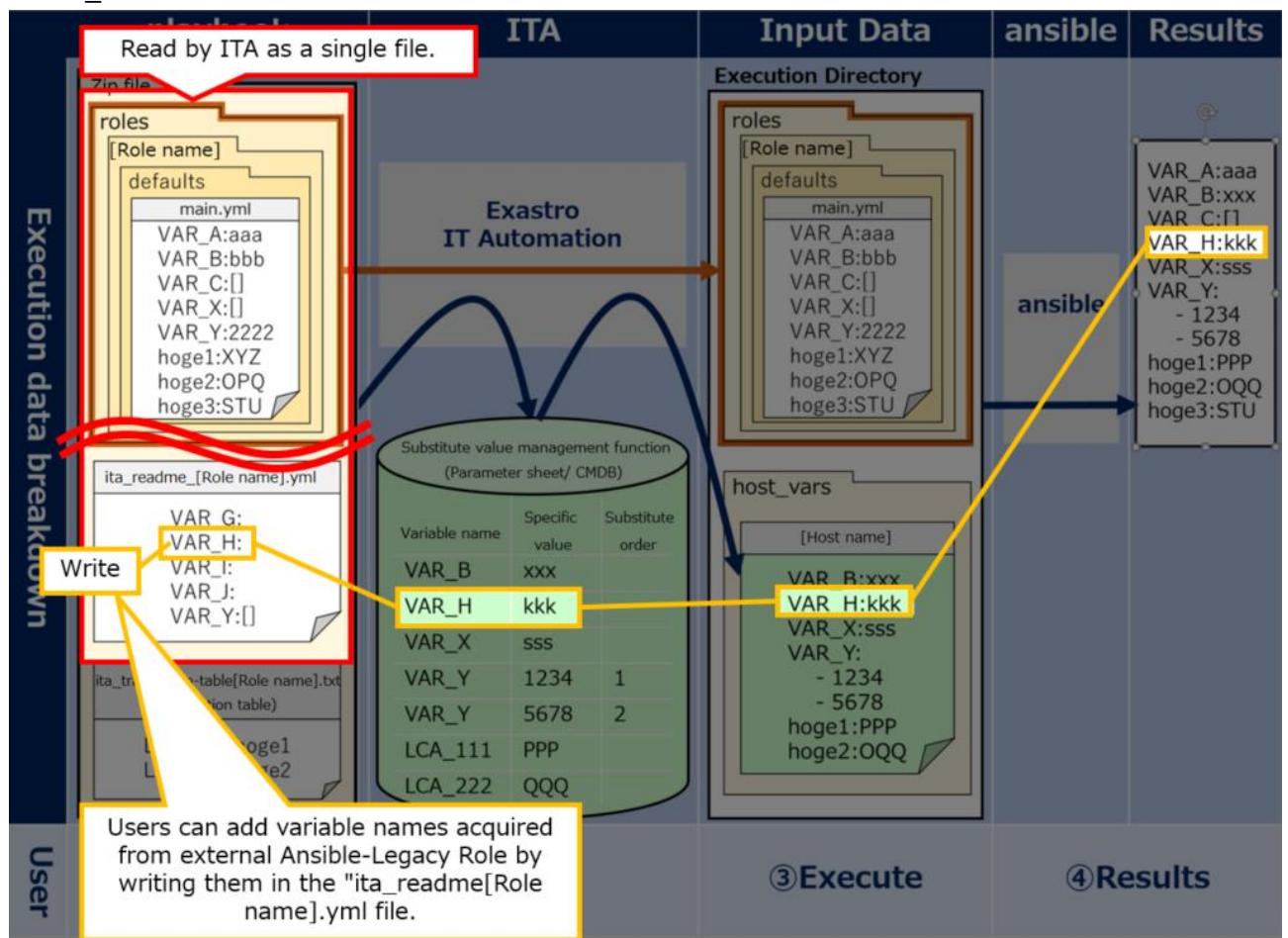


Figure 6.6-6 Case 5

- **Case 6. "VAR_" Prefix**

ITA manages only variables in the "defaults/main.yml" that starts with the prefix "VAR_".

If the user want to manage variables that does not start with "VAR_", use the "Translation table".

Users can define variables that does not start with "VAR_" by writing them in the Translation table and adding prefix "LCA_".

If the user can of course refrain from using the translation table if they want to execute an operation without giving parameters from ITA to "defaults/main.yml" variables (those without the "VAR_" prefix).
※See the variable "hoge" in the figure below

※Translation tables are only active when they contain a definition that starts with "LCA_".

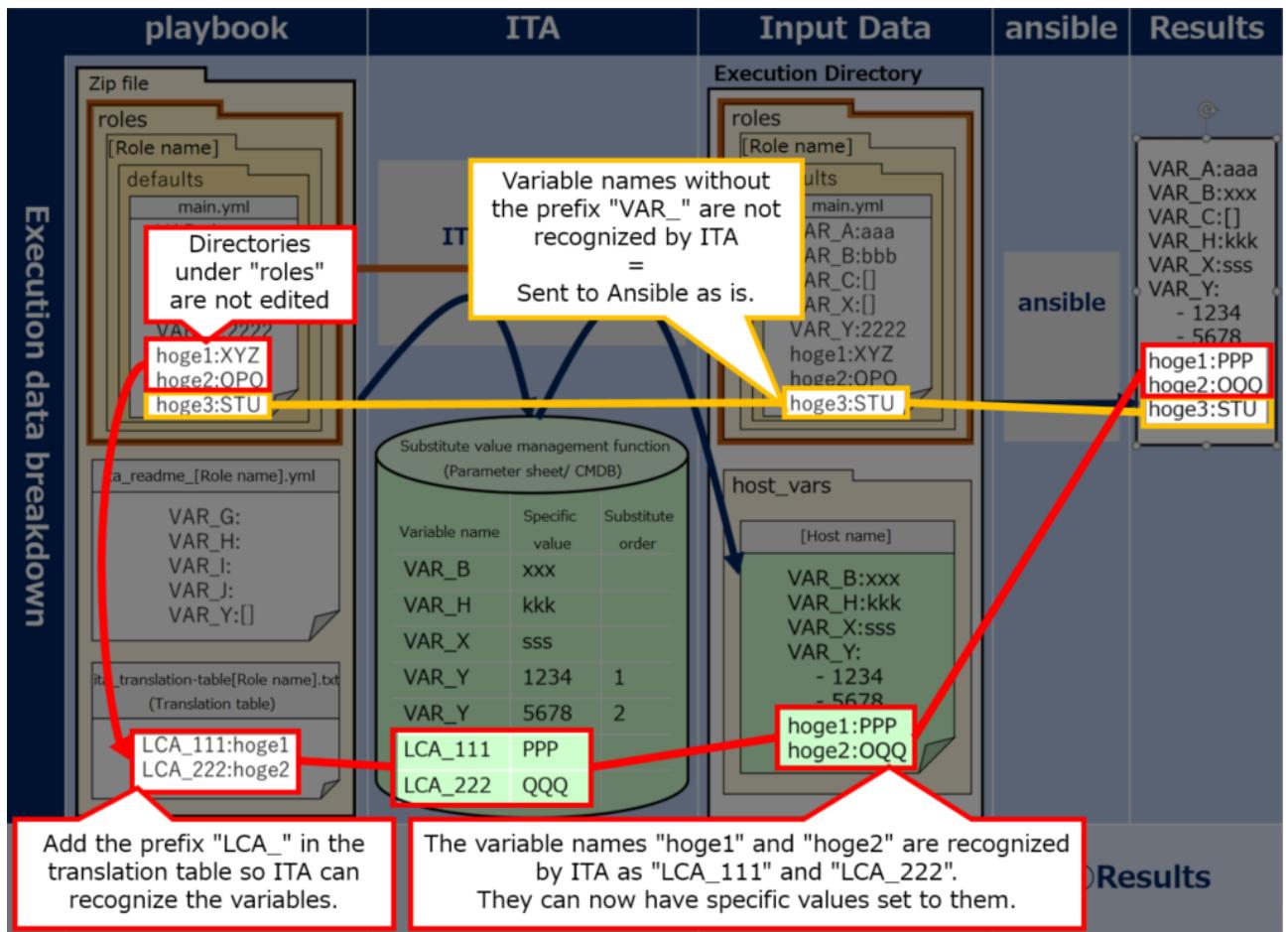


Figure 6.6-7 Case 6

- **Case 7. Linking "ita_readme" and translation table**

Users can give Parameters from ITA to variables in "tasks/main.yml"(Playbook) that does not start with "VAR_" and are not defined in "default/main.yml" by using both the "ita_readme" file and a "translation table".

For example as shown in the figure below, if the "hoge" variable under the "tasks/main.yml" is used, users can follow the following steps in order to send it to ITA.

- ① Add the variable name "hoge" to the "ita_readme" file.
- ② Add the "hoge" to the "Translation table" with the "LCA_" prefix.

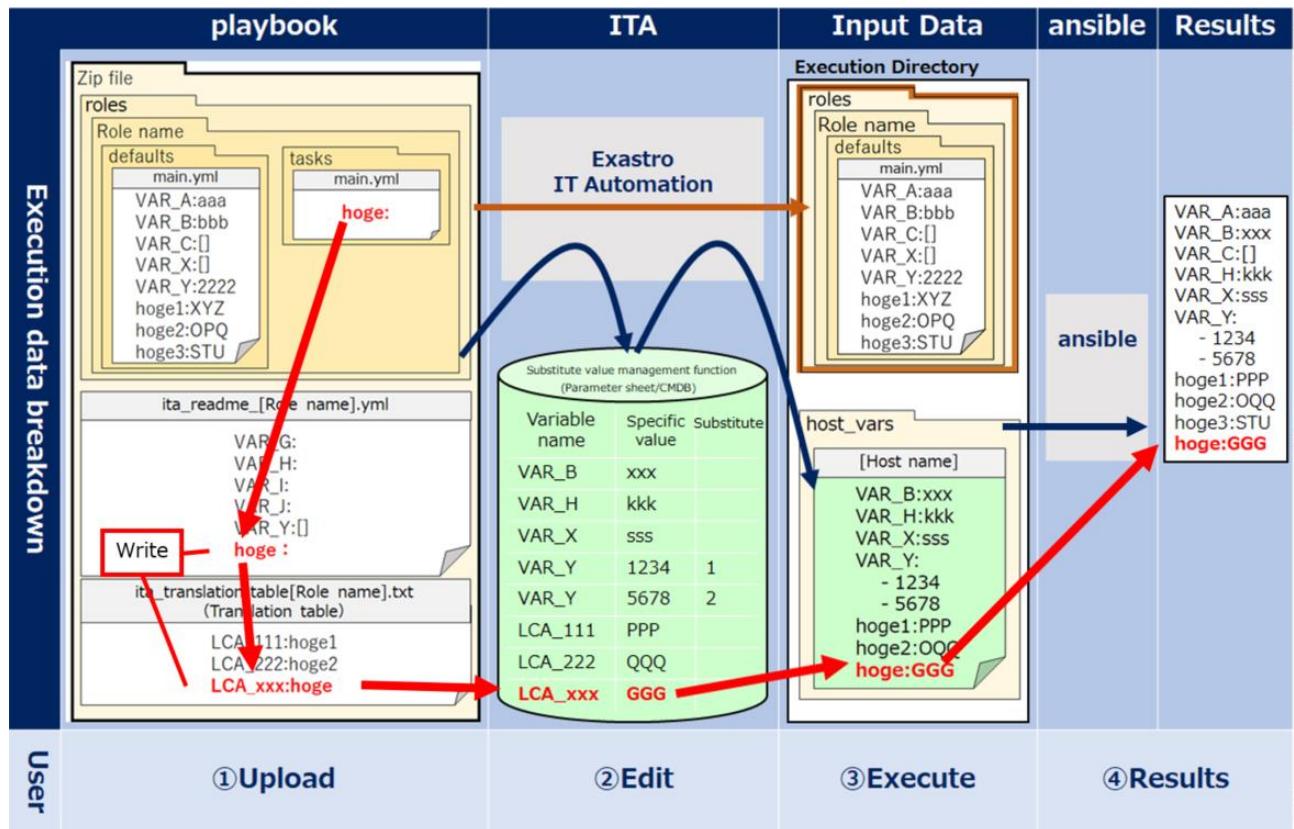


Figure 6.6-8 Case 7

- **Case 8. Applying Playbook Length evaluation**

Depending on whether a variable has a concrete value or not, it can be used as a conditional branch for length evaluation.

For example if "VAR_C:[]" is written in "defaults/main.yml", the length will equal 0 if the operation is executed with no specific value set to "VAR_C".

On the other hand, doing the same with a specific value set will have length be <0 (length<0). (E.g.: VAR_X:sss)

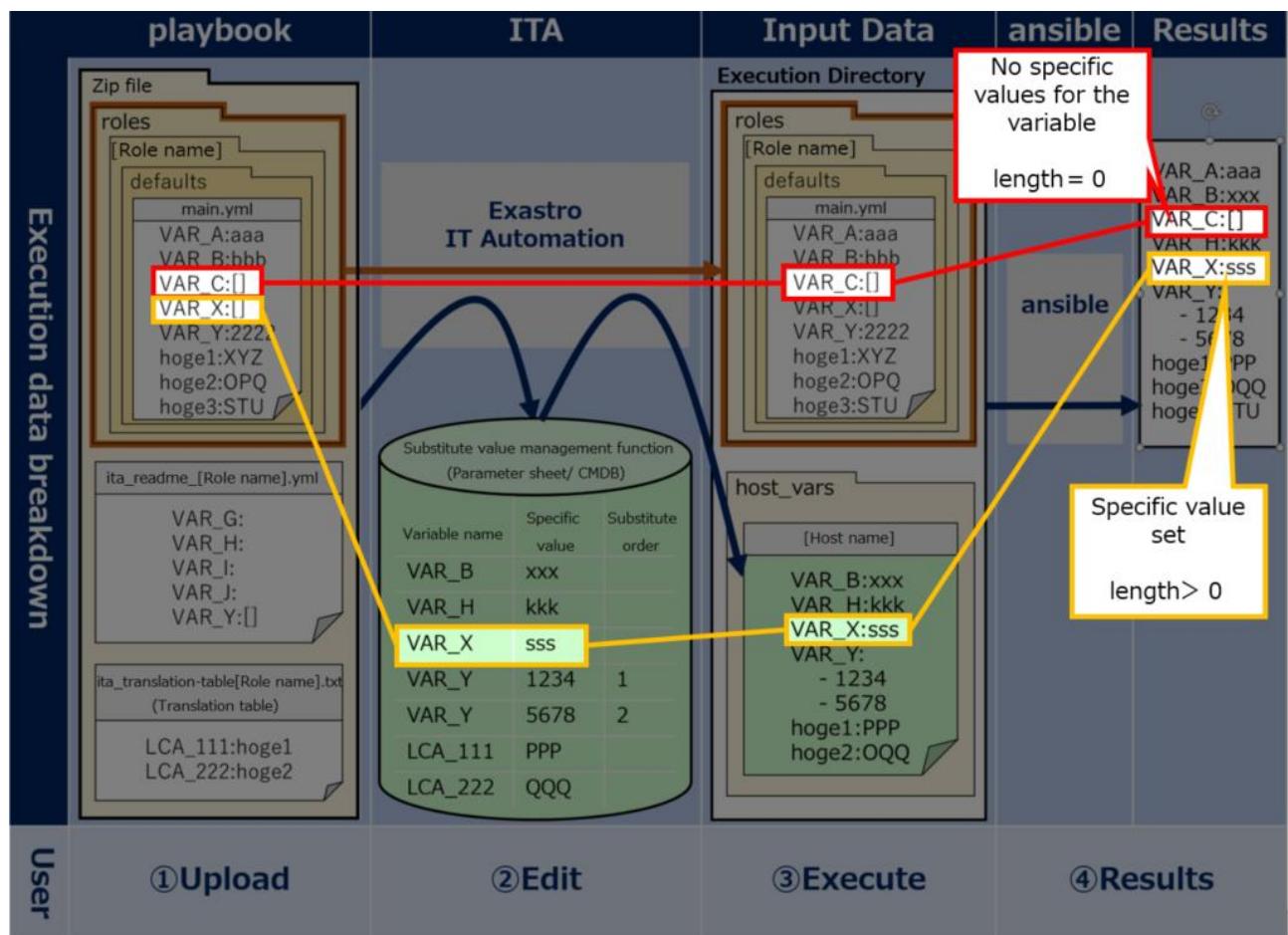


Figure 6.6-9 Case 8

- Case 9. : Applying Playbook Defined evaluation

Depending on whether a variable has a concrete value or not, it can be used as a conditional branch for defined valuation.

For example, first write a definition for the variables "VAR_G" and "VAR_H" in the "ita_readme" file. By doing so, they can be used by ITA's CMDB.

Running an operation without giving a specific value to "VAR_G" while it is not defined in "defaults/main.yml" or "host_vars" will turn "defined" to "false".

On the other hand, if the specific value "kkk" is added to "VAR_H", "defined" will turn into "true".

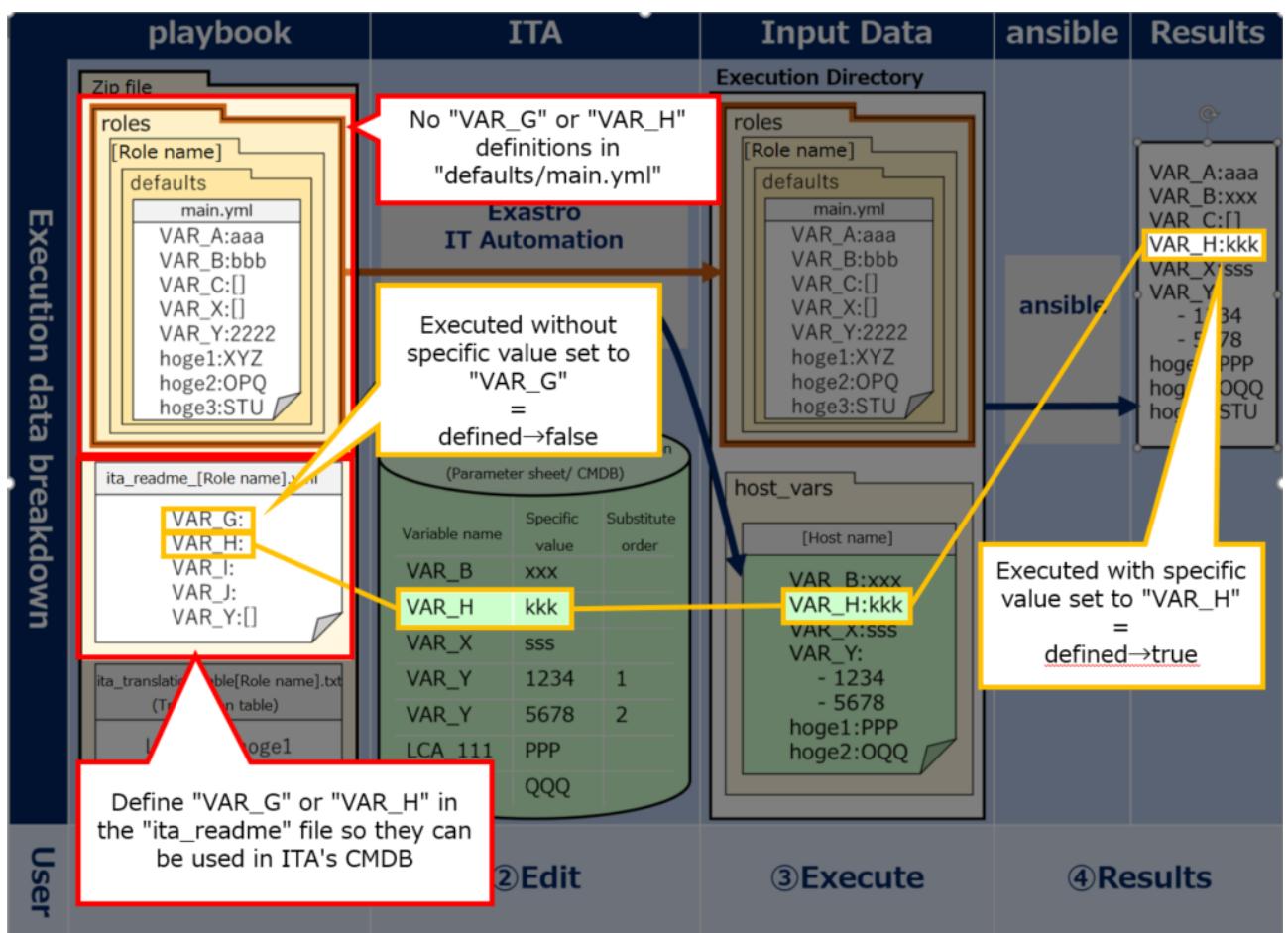


Figure 6.6-10 Case 9

6.7 BackYard contents

(1) Variable auto registration

When the variable analysis target file is uploaded, extract variable from the uploaded file.

Table 6.6-1 Variable handling of the file uploaded in each mode

Menu	Legacy	Legacy Role	Pioneer
Playbook files	○	×	×
Role package list	×	○	×
Dialog files	×	×	○

The extract timing depends on the startup cycle of the automatic process.

※Unique management of variable name

The extracted variable names in all role packages of each mode are uniquely managed.

Since the variable structure is defined in the default variable definition file, the notes when the variable structure is different in each file are as follows

- Single role package

When using same the variable name between roles with different variable structure.

※In the case that the normal variable and nested variable or the nested structure is different between nested variables, etc.

=> Error occurs during file upload.

- All role package

When using same the variable name between role packages with different variable structure.

=> Error occurs during file upload.

(2) Substitution value auto-registration setting

The Operation of parameter sheet and Movement linked with the item setting value of every host and the variable information are reflected to Target host menu and Substitution value list menu as the association target.

The reflection timing also depends on the startup cycle of the automatic process as written above. Target host and substitution value list menu is updated by multiple operators. Reflection will not be performed if the last updater is other operator (not Backyard).

When user wants to reflect the data in substitution value auto-registration setting menu, please perform operations such as discard the applicable record in substitution value list value or disable the applicable record in other BackYard process.

The reflection rules to Target host menu and Substitution value list menu are as follows.

- ① When reflecting the information registered in substitution value auto-registration setting to substitution value list.

The status of substitution value list	Doesn't exist applicable record	Exist applicable record			Applicable record is being discarded	
		=	≠ Specific value			
		Specific value	Last updated by	BackYard process		
Reflection in substitution value list	Add new record	-	Update the specific value of the applicable record	-	Restore the discarded record	

※Applicable record: The record that has same Operation+host+Movement+variable name + (member variable)+(include order)

- ② When reflecting the information that is not registered in "substitution value auto-registration setting" menu (registered only in "substitution value" menu) to substitution value list

The status of substitution value list	Exist applicable record		
	Last updated by		
	BackYard process	Other operators	
Reflection in substitution value list	Discard the applicable record		-

- ③ When reflecting the information registered in substitution value auto-registration setting to target host menu

The status of operation target host	Doesn't exist applicable record	Exist applicable record	Applicable record is being discarded
Reflection in operation target host	Add new record	-	Restore the discarded record

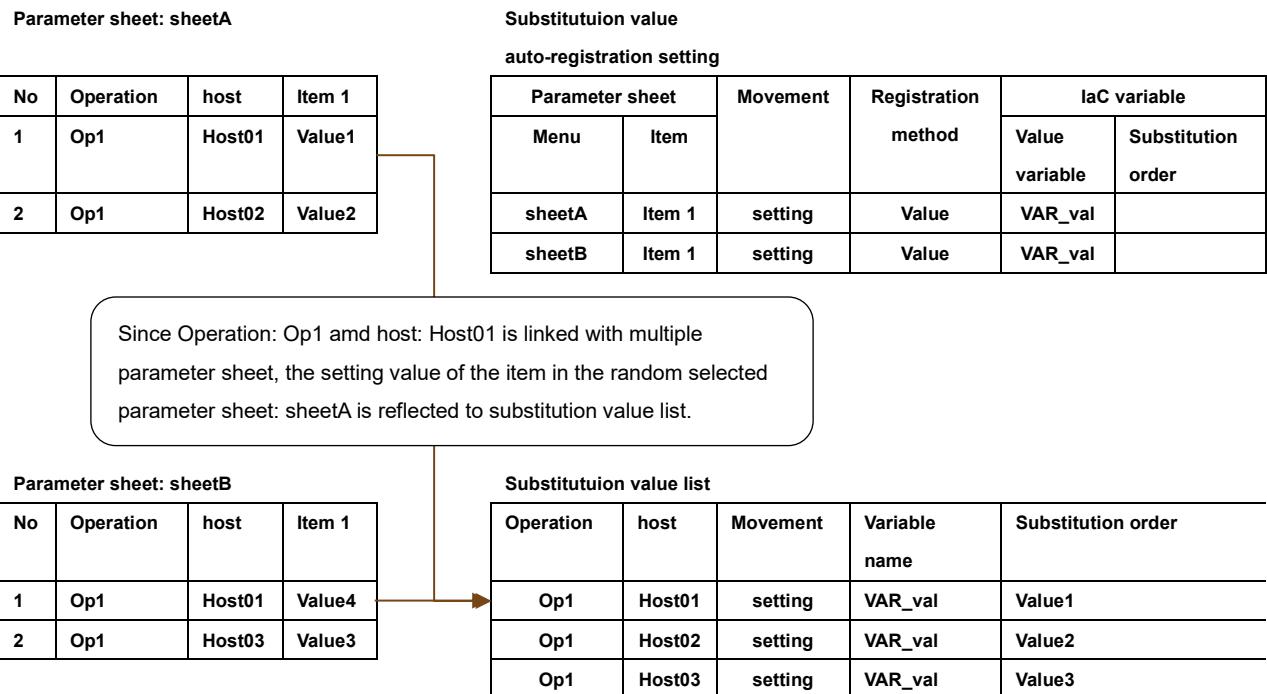
※ Applicable record: The record that has same Operation+host+Movement

- ④ When reflecting the information that is not registered in substitution value auto-registration setting(only registered in target host menu) to target host menu

The status of substitution value list	Exist applicable record		
	Last updated by		
	BackYard process	Other operators	
Reflection in substitution value list	Discard the applicable record		-

- ⑤ When link to multiple items is registered to for the same Movement, variable, and substitution order.

When multiple parameter sheets are linked with same operation and host, one item is selected randomly and reflected to substitution value list.



6.8 Ansible usage guideline ITA additional rules

A Playbook creation guideline for using ITA to execute on Ansible.

Please refer to the attachment "User Instruction Manual - Ansible-driver attachment- Ansible usage guideline with additional rules" for details.

7 Application operation

The operation to utilize ITA system contains not only inputs by user from the browser screen of client PC but also operations according to system operation and maintenance. The available operation and maintenance are as follows.

7.1 Maintenance

The required file to start/stop/restart Ansible driver independent processes are as follows.

Description	Target file name
Legacy/pioneer/legacyRole execution monitor Execute the unexecuted Operation.	ky_legacy_execute-workflow.service
legacy variable automatic registration Extract variable from the uploaded file.	ky_legacy_varsautolistup-workflow.service
legacy automatic registration setting Reflect the information set in auto-registration setting to substitution value list and operation target host menu.	ky_legacy_valautostup-workflow.service
pioneer automatic registration setting Reflect the information set in auto-registration setting to substitution value list and operation target host menu.	ky_pioneer_valautostup-workflow.service
legacyRole variable automatic registration Extract variable from the uploaded file.	ky_legacy_role_varsautolistup-workflow.service
legacyRole automatic registration setting Reflect the information set in auto-registration setting to substitution value list and operation target host menu.	ky_legacy_role_valautostup-workflow.service
Ansible Automation Controller data synchronization Obtain setting information from Ansible Automation Controller.	ky_ansible_towermasterSync-workflow.service

The target files are stored in “/usr/lib/system/system”.

The method to start/stop/restart process are as follows.

Please execute the command with root privilege.

- ① Start process

```
# systemctl start ky_legacy_execute-workflow.service
```

- ② Stop process

```
# systemctl stop ky_legacy_execute-workflow.service
```

- ③ Restart process

```
# systemctl restart ky_legacy_execute-workflow.service
```

Please substitute each target file name to start/stop/restart the process.

7.2 About the maintenance method

- ① Change to NORMAL level

Rewrite the 8th line of the following file from “DEBUG” to “NORMAL”.

Log level setting file: <installation directory>/ita-root/confs/backyardconfs/ita_env

- ② Change to DEBUG level

Rewrite the 8th line of the following file from “NORMAL” to “DEBUG”.

Log level setting file: <installation directory>/ita-root/confs/backyardconfs/ita_env

- ③ Change the startup cycle

Change the 5th parameter of ExecStart in each target file. (Unit: second)

Please use the default value of startup cycle excluding exceptions.

```
ExecStart=/exastro/ita-root/backyards/common/ky_lopcall-php-procedure.sh  
/usr/local/bin/php /usr/local/bin/php /exastro/ita-  
root/backyards/ansible_driver/ky_pioneer_varsautolistup-workflow.php /exastro/ita-  
root/logs/backyardlogs 10 NORMAL > /dev/null 2>&1
```

After rewriting the file, the change takes effect after restarting the process.

Log file output destination: <installation directory>/ita-root/logs/backyardlogs

8 Appendix

8.1 The linkage between the input data used during Ansible execution and ITA menu

Extract information from each ITA menu and create the "input data" that is required for Ansible execution. At that time, the password in device list menu is encrypted. Ansible-Legacy and Ansible-LegacyRole encrypts with ansible-vault, while pioneer encrypts with the original method of ITA. The "Input data" can be downloaded from "[5.3.12 Check operation status](#)" in ZIP format. Executing the following command in Ansible directly is also possible.

The relationship between various data and the ITA menu is as follows.

8.1.1 Ansible-Legacy input data

【Parent directory】

child_playbooks	The directory containing user created Playbooks.		
	Ansible-Legacy	Playbook files	Playbook file
	Ansible-Legacy	Movement details	Include order
template_files	The directory containing the template file used in Playbook that is going to be executed.		
	Ansible-Legacy	Template list	Template file
	Ansible-Legacy	Movement details	Include order
copy_files	The directory containing the file that is going to be deployed on operation target server.		
	Ansible-Legacy	Contents list	Files
	Ansible-Legacy	Movement details	Include order
host_vars	The directory containing the host information of the operation target host and the definition file of various variable.		
	Ansible common	Global variable list	Variable name/specific value
	Ansible-Legacy	Substitution value list	Variable name/specific value
	Ansible-Legacy	Template list	Template file
	Ansible-Legacy	Contents list	File variable name
	Ansible-Legacy	Movement details	Include order
	Ansible-Legacy	Interface information	Data relay storage path(ITA)
	Ansible-Legacy	Interface information	Symphony instance data relay storage path(Ansible)
	Basic console	Device list	Protocol
	Basic console	Device list	Login user ID
	Basic console	Device list	Login password ※Encrypted with ansible-vault
	Basic console	Device list	Host name

ssh_key_files	The directory in which the specified ssh authentication key file is stored when using key authentication as the authentication method.		
	Basic console	Device list	ssh authentication key file
winrm_ca_files	The directory in which the file that defines the connection information when connecting to WinRM is stored.		
	Basic console	Device list	WinRM connection information
AnsibleExecOption.txt	Parameter for AnsiblePlaybook execution		
	Ansible common	Interface information	Option parameter
	Ansible-Legacy	Movement list	Number of parallel executions
hosts	The file describing the operation execution target host		
	Basic console	Device list	host name
	Basic console	Device list	IP address
	Basic console	Device list	Login user ID
	Basic console	Device list	Login password
	Basic console	Device list	※Encrypted with ansible-vault
	Basic console	Device list	Connection options ※The parameter of ansible_ssh_extra_args
	Basic console	Device list	ssh authentication key file
	Basic console	Device list	Server certificate
	Basic console	Device list	Inventory file additional option
playbook.yml	The file calls the whole information of playbook and host information and executes Ansible.		
	Ansible-Legacy	Playbook files	Playbook file
	Ansible-Legacy	Movement details	Include order
	Ansible-Legacy	Movement details	gather_facts

8.1.2 Ansible-Pioneer input data

【Parent directory】			
template_files	The directory containing the template file used in Playbook that is going to be executed.		
	{ Ansible-Pioneer Template list Template file Ansible-Pioneer Movement details Include order		
copy_files	The directory containing the file that is going to be deployed on operation target server.		
	{ Ansible-Pioneer Contents list Files Ansible-Pioneer Movement details Include order		
ssh_key_files	The directory in which the specified ssh authentication key file is stored when using key authentication as the authentication method.		
	{ Basic console Device list ssh authentication key file		
winrm_ca_files	The directory in which the file that defines the connection information is stored when connecting to WinRM.		
	{ Basic console Device list WinRM connection information		
host_vars	The directory in which the host information of the operation target host and the definition file of various variable is stored.		
	{ Ansible common Interface information Data relay storage path(ITA) Ansible common Interface information Symphony instance data relay storage path(Ansible) Ansible common Global variable list Variable name/specific value Ansible-Pioneer Substitution value list Variable name/specific value Ansible-Pioneer Template list Template file Ansible-Pioneer Movement details Include order Ansible-Pioneer Contents list File variable name Ansible-Pioneer Movement details Include order Basic console Device list Login password Basic console Device list Host name Basic console Device list Connection options Basic console Device list Protocol Basic console Device list Login user ID ※Encrypted with ITA original method		
dialog_files	The directory in which user created dialog files is stored.		

	Ansible-Pioneer	Dialog files	Dialog file																					
	Ansible-Pioneer	Movement details	Include order																					
AnsibleExecOption.txt	Parameter for AnsiblePlaybook execution.																							
	<table border="0"> <tr> <td>Ansible common</td> <td>Interface information</td> <td>Option parameter</td> </tr> </table>			Ansible common	Interface information	Option parameter																		
Ansible common	Interface information	Option parameter																						
hosts	The file describing the operation execution target host																							
	<table border="0"> <tr> <td>Basic console</td> <td>Device list</td> <td>Host name</td> </tr> <tr> <td>Basic console</td> <td>Device list</td> <td>IP address</td> </tr> <tr> <td>Basic console</td> <td>Device list</td> <td>Login user ID</td> </tr> <tr> <td>Basic console</td> <td>Device list</td> <td>Login password</td> </tr> <tr> <td>Basic console</td> <td>Device list</td> <td>※Encrypted with ITA original method</td> </tr> <tr> <td>Basic console</td> <td>Device list</td> <td>ssh authentication key file</td> </tr> <tr> <td></td> <td></td> <td>Inventory file additional option</td> </tr> </table>			Basic console	Device list	Host name	Basic console	Device list	IP address	Basic console	Device list	Login user ID	Basic console	Device list	Login password	Basic console	Device list	※Encrypted with ITA original method	Basic console	Device list	ssh authentication key file			Inventory file additional option
Basic console	Device list	Host name																						
Basic console	Device list	IP address																						
Basic console	Device list	Login user ID																						
Basic console	Device list	Login password																						
Basic console	Device list	※Encrypted with ITA original method																						
Basic console	Device list	ssh authentication key file																						
		Inventory file additional option																						
playbook.yml	The file calls the whole information of playbook and host information and executes Ansible.																							
	<table border="0"> <tr> <td>Ansible-Pioneer</td> <td>Interface information</td> <td>Data relay storage path(ITA)</td> </tr> </table>			Ansible-Pioneer	Interface information	Data relay storage path(ITA)																		
Ansible-Pioneer	Interface information	Data relay storage path(ITA)																						

8.1.3 Ansible-LegacyRole input data

【Parent directory】		
copy_files	The directory in which the file that is going to be deployed on operation target server is stored.	
	Ansible-LegacyRole	Contents list
	Ansible-LegacyRole	Movement details
		Files
		Include order
roles	The directory containing the user created role.	
	Ansible-LegacyRole	Role package list
		Role package file (ZIP format)
ssh_key_files	The directory in which the specified ssh authentication key file is stored when using key authentication as the authentication method.	
	Basic console	Device list
		ssh authentication key file
winrm_ca_files	The directory in which the file that defines the connection information when connecting to WinRM is stored.	
	Basic console	Device list
		WinRM connection information
host_vars	The directory containing the host information of the operation target host and the definition file of various variable.	
	Ansible common	Interface information
	Ansible common	Interface information
	Ansible common	Global variable list
	Ansible-LegacyRole	Substitution value list
	Ansible-LegacyRole	Template list
	Ansible-LegacyRole	Movement details
	Ansible-LegacyRole	Contents list
	Ansible-LegacyRole	Movement details
	Basic console	Device list
		Data relay storage path(ITA)
		Symphony instance data relay storage path(Ansible)
		Variable name/specific value
		Variable name/specific value
		Template file
		Include order
		File variable name
		Include order
		Protocol
		Login user ID
		Login password
		※Encrypted with ansible-vault
		Host name

AnsibleExecOption.txt Parameter for AnsiblePlaybook execution.

Ansible common	Interface information	Option parameter
Ansible-LegacyRole	Movement list	Number of parallel executions

hosts The file describing the operation execution target host.

Basic console	Device list	host name
Basic console	Device list	IP address
Basic console	Device list	Login user ID
Basic console	Device list	Login password
Basic console	Device list	※Encrypted with ansible-vault
Basic console	Device list	Connection options
Basic console	Device list	※The parameter of ansible_ssh_extra_args
Basic console	Device list	ssh authentication key file
Basic console	Device list	Server certificate
Basic console	Device list	Inventory file additional option

site.yml The file calls the whole information of playbook and host information and executes Ansible.

Ansible-Legacy	Playbook files	Playbook file
Ansible-Legacy	Movement details	Include order
Ansible-Legacy	Movement details	gather_facts

8.1.4 Directly executing the input data

(1) Create a directory where the input data will be decompressed.

Create the two following directories and extract the input data into directory 1

1. /Base Directory/Driver path/Operation No./in

2. /Base Directory/Driver path/Operation No./out

Base Directory: Interface information=>Data relay storage path(Ansible/Ansible Automation Controller)

Driver path: legacy: legacy/ns Legacy-role: legacy/rl pioneer: pioneer/ns

Operation No.: Number of the Operation when executed. The whole number is 10 numbers.

The operation number is then moved to the right and the rest of the numbers are filled with "0".

Operation No.: 12345 => 0000012345

The input file does not include the secret key file uploaded to the device list. If chosen authentication method requires a secret key file, open the inventory file "hosts" included in the input data and copy the path of the secret key file set to "ansible_ssh_private_key_file".

```
Inventory file"hosts"
all:
children:
hostgroups:
hosts:
target_host_1:
ansible_user: keyauth_user
ansible_ssh_private_key_file: /exastro/data_relay_storage/ansible_driver/legacy/ns/0000000060/in/ssh_key_files/000000006-keyauth_user_id_rsa
```

(2) Commands that directly executes input data.

Ansible-Legacy

ansible-playbook (Option) –i hosts --vault-password-file Password file playbook.yml

Ansible-Pioneer

ansible-playbook (Option) –i hosts --vault-password-file Password file playbook.yml

Ansible-LegacyRole

ansible-playbook (Option) –i hosts --vault-password-file Password site.yml

The Password file name can be whatever you want.

The password set in the password file should be the value of the contents of ITA-Installation-directory/ita-root/confs/commonconfs/ansible_vault_accesskey.txt, decoded in the order rot13, base64.

8.2 Result data created during Ansible execution

The result of executing [Input data] with ansible is saved as [Result data] in ZIP format. [Result data] can be downloaded in ZIP format from "[5.3.12 Check operation status](#)".

8.2.1 Legacy/LegacyRole List of files saved in result data

Table 8.2.1-1 Legacy/LegacyRole List of files where result data is saved

File name	Record content	In Ansible Engine	In Ansible Tower
result.txt	Record the execution checking of Ansible	<input type="radio"/>	
xxx.pid	A file that records the process ID of the Ansible-playbbok command. xxx: Process ID of the Ansible-playbbok command	<input type="radio"/>	
error.log	Error output destination file when ITA outputs an error message due to some error when executing work, or when the Ansible-playbbok command outputs an error message due to some error. The contents displayed in the error log of work execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log	A log file that is a partial processing of the execution log output by Ansible-playbbok. Contents displayed in the execution log of work execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log.org	Execution log output by ansible-playbook	<input type="radio"/>	<input type="radio"/>
Ita_<mode name>_executions_jobtpl_<execution no.>_<group no.>	Splited execution log file. For the naming convention of the file name, please refer to ⑥ Execution log display of 5.3.12 Check operation status.		<input type="radio"/>
forced.txt	Record file in emergency stop	<input type="radio"/>	
user_files	The directory where the file is recorded when some kind of file is output to the ITA original variable <code>__workflowdir__</code> in the executed playbook.	<input type="radio"/>	<input type="radio"/>

8.2.2 List of files saved in Pioneer result data

Table 8.2.1-2 List of files for which Pioneer results data is stored

File name	Record content	In Ansible Engine	In Ansible Tower
result.txt	Record the execution result of Ansible	<input type="radio"/>	
xxx.pid	A file that records the process ID of the Ansible-playbook command. xxx: Process ID of the Ansible-playbook command	<input type="radio"/>	
error.log	The error destination file when the ITA outputs an error message due to some error when executing or if the Ansible-playbook command outputs an error message due to some error. The contents will be displayed in the error log of execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log	A log file that is a partial processing of the execution log output by Ansible-playbook. Contents displayed in the execution log of execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log.org	Execution log output by ansible-playbook	<input type="radio"/>	<input type="radio"/>
Ita_<mode name>_executions_jobtpl_<execution no.>_<group no.>	Split execution log file. Please refer to (6) execution log display of 5.3.12 check operation status for the naming convention of the file name.		<input type="radio"/>
forced.txt	Record file in case of emergency stop.	<input type="radio"/>	
user_files	A directory where files are recorded when some file is output to ITA's original variable "__workflowdir__" in the playbook executed.	<input type="radio"/>	<input type="radio"/>

(1) Legacy-Role

Table 9.2.1-3 Legacy-Role list of files where result data is saved

File name	Record content	In Ansible Engine	In Ansible Tower
result.txt	Record the execution result of Ansible	<input type="radio"/>	
xxx.pid	A file that records the process ID of the Ansible-playbook command. xxx: Process ID of the Ansible-playbook command	<input type="radio"/>	
error.log	The error destination file when the ITA outputs an error message due to some error when executing or if the Ansible-playbook command outputs an error message due to some error. The contents will be displayed in the error log of execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log	A log file that is a partial processing of the execution log output by Ansible-playbook. Contents displayed in the execution log of execution confirmation.	<input type="radio"/>	<input type="radio"/>
exec.log.org	Execution log output by ansible-playbook	<input type="radio"/>	<input type="radio"/>
Ita_<mode name>_executions_jobtpl_<execution no.>_<group no.>	This is a divided execution log file. Please refer to (6) execution log display of 5.3.12 check operation status for the naming convention of the file name.		<input type="radio"/>
forced.txt	Record file in case of emergency stop.	<input type="radio"/>	
user_files	A directory where files are recorded when some file is output to ITA's original variable "__workflowdir__" in the playbook executed.	<input type="radio"/>	<input type="radio"/>

8.3 Option Parameter list

The following section explains the option parameters for the Interface information and the Movement list.

In ITA, the ansible-playbook option parameters are configured in the following order.

If multiple parameters that only allow a single values are defined, the parameters in the Movement list=>optional parameters will be activate.

- Ansible common=> Interface information=> Option parameter
- Movement list=> Option parameter

For Ansible Core

If the Ansible Core is set as the Execution engine, input the ansible-playbook command's option parameter.

For more information regarding the ansible-playbook commands' option parameter, please run the commands below and see the help information displayed.

```
「 ansible-playbook -h 」
```

For execution engines other than Ansible Core

The following table displays the specifyable option parameters for execution engines other than Ansible Core.

Table 8.3.1 List of specifyable option parameters for execution engines other than Ansible Core

Option Parameter	Specification method	Ansible Automation Controller settings location	Remarks
-v --verbose	-v -vv -vvv -vvvv -vvvvv --verbose	Configure number of "v"s specified to the Template screen's "More information"	<ul style="list-style-type: none">Uses the total amount of "v"s--verbose" is the same as "-v" <p>Example: "--verbose -vvv" is the same as "-vvvv"</p> <ul style="list-style-type: none">If more than 6 "v"s are specified, it will be specified as 5.
-f --forks	-f FORKS --forks=FORKS	The "FORKS" specified in the template's "Fork" is set.	Specify numeric values for FORKS If multiple are defined, the last parameter defined will be active. Example: If -f 1 --forks=10, then --forks=10 will be active. If something other than a numeric value is specified, an error will occur.
-I	-I SUBSET	The SUBMIT	<ul style="list-style-type: none">SUBSET: Host name in the

Option Parameter	Specification method	Ansible Automation Controller settings location	Remarks
--limit	--limit=SUBSET	specified to the Template's "Limit" is set	Device list • If multiple are defined, the last parameter defined will be active.
-e --extra-vars	-e EXTRA_VARS --extra-vars=EXTRA_VARS	テンプレートの[追加変数]に変数名:具体値の形式で設定される	• EXTRA_VARS: Variable name, Specific values are either json or yaml format. Example: For json format: --extra-vars={"VAR_1":"directory","VAR_2":"0755"} For yaml format: --extra-vars=VAR_1: directory VAR_2: 0755 • If multiple are defined, the last defined parameter will be active.
-t --tags	-t TAGS --tags=TAGS	テンプレートの[ジョブタグ]に設定した TAGS が設定される	• TAGS: Tag name • Allows for multiple parameters
-b --become	-b --become	テンプレートのオプション[権限昇格の有効化]がチェックされる	• Parameters are active as long as at least one is specified
-D --diff	-D --diff	テンプレートの[変更]の表示が有効化される	• Parameters are active as long as at least one is specified
--skip-tags	--skip-tags=SKIP_TAGS	テンプレートの[スキップタグ]に設定した SKIP_TAGS が設定される	• SKIP_TAGS: Skip tag name • Allows for multiple parameters
--start-at-task	--start-at-task=START_AT_TASK	※Ansible Automation Controller の Web UI には --start-at-task の表示はない。	• If multiple are defined, the last parameter defined will be active.
-ufc --use_fact_cache	-ufc --use_fact_cache	テンプレートのオプション[ファクトキャッシュの有効化]がチェックされる	• Parameters are active as long as at least one is specified
-kkoas --allow_simultaneous	-as --allow_simultaneous	テンプレートのオプション[同時実行ジョブの有効化]がチェックされる	• Parameters are active as long as at least one is specified
-jsc --job_slice_count	-jsc Number of job slices --	テンプレートの[ジョブスライス数]に指定し	• Specify numeric value for amount of Job slices

Option Parameter	Specification method	Ansible Automation Controller settings location	Remarks
	job_slice_count=Nakumber of job slices	たジョブスライス数が設定される	<ul style="list-style-type: none"> If multiple are defined, the last parameter defined will be active.

※For more information regarding Ansible Automation Controller's option parameters, please refer to the explanation for Ansible Automation Controller user guide's job template.

8.4 Using ITA proprietary variables in Ansible Automation Controller.

The following is a list of points to keep in mind when executing operations with Ansible Automation Controller in a cluster configuration for Movements including Playbooks, which uses the following ITA proprietary variables to output files.

ITA Proprietary variables

- __workflowdir__
- __symphony_workflowdir__
- __conductor_workflowdir__
- __movement_status_filepath__
- __parameters_dir_for_epc__
- __parameters_file_dir_for_epc__
- __parameter_dir__
- __parameters_file_dir__

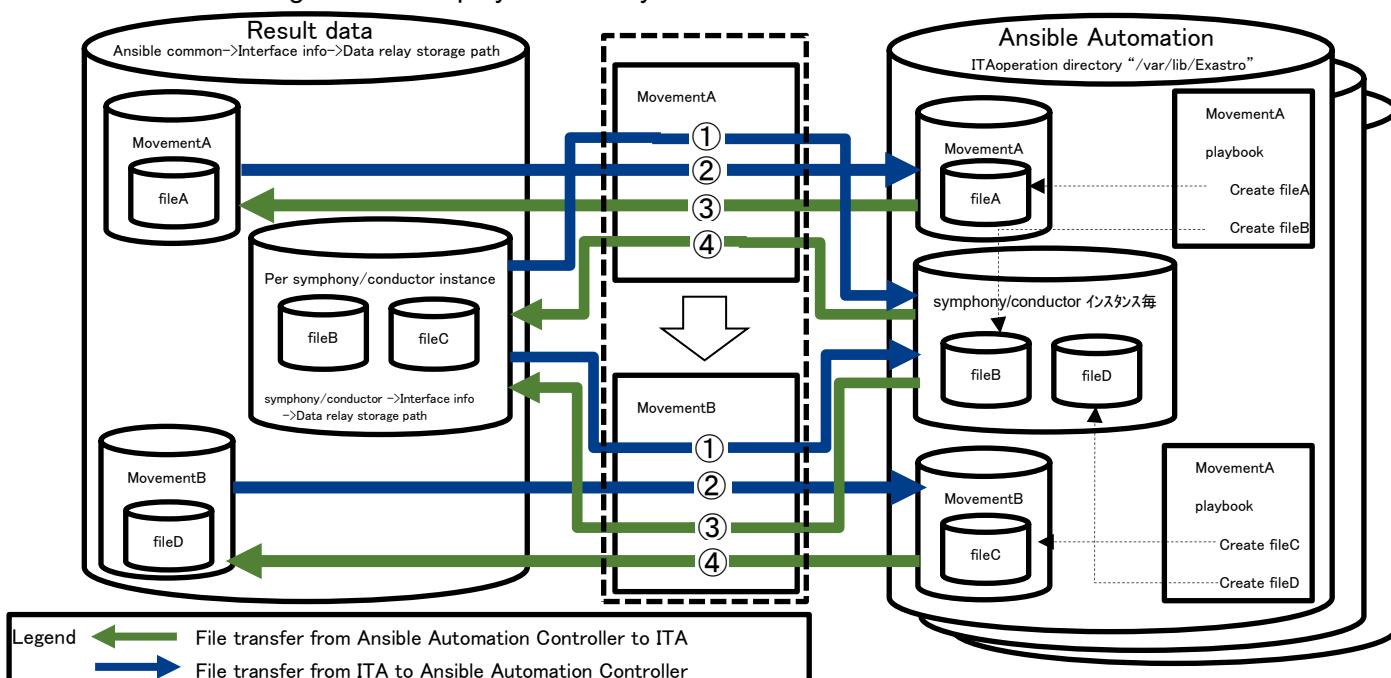
(1) Managing files created using ITA's proprietary variables

The Output directory of the files created using the ITA proprietary values is set to the Ansible Automation Controller's ITA Operation directory, "/var/lib/exastro".

Before Movements are executed, the result data is transferred as a file to "/var/lib/exastro" under the Ansible Automation Controller's ITA operation directory.

The file created when the Movement is executed is transferred from the Ansible Automation Controllers to the result data in overwrite mode.

If a file with the same name as an already existing file is created, the result data of the updated symphony/conductor file might not be displayed correctly.



- ① If a Movement is executed from symphony/conductor The files placed under the corresponding symphony/conductor instance is transferred to the Ansible Automation Controller's ITA operation directory
- ② The files under the corresponding Movements are transferred to the Ansible Automation Controller's ITA operation directory before the Movement is executed.
- ③ If a Movement is executed from symphony/conductor,

the file created by the corresponding movement in the ITA operation directory in the Ansible Automation Controller is transferred to the result data.

- ④ The file created under the corresponding symphony/conductor instance under the Ansible Automation Controller's ITA operation directory is transferred to the result data after the Movement is executed.

(2) Important notes

① Make sure that the file name includes `ansible_loginhostname` for each target host linked to the movement in order to avoiding overlapping of file names.

② If executing from symphony/conductor, make sure that the movement file names don't overlap.

8.5 Data resources deleted when executing.

The following list contains data resources that will be deleted if the user selects "Delete" when executing interface information.

Table 8.5-1 List of data resources deleted when running data deletion.
(Ansible Automation Controller side)

Data resources	Resource type	Execution engine		Remarks
		Ansible Tower3.x	Ansible Automation Controller4.x	
ITA operation directory /var/lib/exastro/ita_< section >_executions_10 digit operation number	Directory	○	○	
SCM management directory /var/lib/awx/projects/ita_< section >_executions_10 digit operation number	Directory	○	※	※ Deleted by Project resource deletion
Inventory Resource name: ita_< section >_executions_inventory_10 digit operation number	Resource	○	○	
Authentication information Resource name:: ita_< section >_executions_credential_10 digit operation number_serial number ita_< section >_executions_vault_credential_10 digit operation number ita_< section >_executions_git_credential_10 digit operation number	Resource	○	○	
Project ↳ Resource name:: ita_< section >_executions_project_10 digit operation number	Resource	○	○	
Job template Resource name:: ita_<section>_executions_jobtpl_10 digit operation number_serial number	Resource	○	○	
Workflow job template Resource name:: ita_< section >_executions_workflowtpl_10 digit operation number	Resource	○	○	
Job Resource name:: Job number—ita_< section >_executions_workflowtpl_10 digit operation number Job number—ita_< section >_executions_jobtpl_10 digit operation number	Resource	○	○	

Table 8.5-2 List of data resources deleted when running data deletion.
 (ITA side)

Data resources	Resource type	Execution engine		Remarks
		Ansible Tower3.x	Ansible Automation Controller4.x	
Git repository ITAinstall directory/ita-root/repositorys/ansible_driver/<Section>_10 digit operation number	Git repository	<input checked="" type="radio"/>	<input checked="" type="radio"/>	Deletion per directory

section: legacy/legacy_role/pioneer