

ITA\_User\_Instruction\_Manual

Terraform-CLI-driver

*－* Version 1.11*－*

Copyright © NEC Corporation 2023. All rights reserved.

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 Manual are subject to change without notice.

NEC Corporation is not responsible for any technical or editorial errors or Shorteneds 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, and Tomcat are registered trademarks or trademarks of the Apache Software Foundation.
* Terraform is a registered trademark or trademark of HashiCorp.

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 are not specified in this document.

※”Exastro IT Automation” is written as “ITA” in this document.

Table of Contents

[Table of Contents 3](#_Toc126763556)

[Introduction 4](#_Toc126763557)

[1 Terraform driver overview 5](#_Toc126763558)

[1.1 Terraform 5](#_Toc126763559)

[1.2 Terraform-CLI-driver 5](#_Toc126763560)

[2 Handling variables with Terraform-CLI-driver 6](#_Toc126763561)

[2.1 Variable type 6](#_Toc126763562)

[2.2 Extraction of variables and registration of specific values 6](#_Toc126763563)

[2.3 Variable types 7](#_Toc126763564)

[3 Terraform driver console menu structure 12](#_Toc126763565)

[3.1 Menu / screen list 12](#_Toc126763566)

[4 Terraform-CLI-driver workflow 14](#_Toc126763567)

[4.1 Terraform workflow 14](#_Toc126763568)

[5 Terraform driver function and operation method explanation 16](#_Toc126763569)

[5.1 Basic console 16](#_Toc126763570)

[5.1.1 Operation list 16](#_Toc126763571)

[5.2 Terraform-CLI-driver console 17](#_Toc126763572)

[5.2.1 Interface information 17](#_Toc126763573)

[5.2.2 Workspaces list 19](#_Toc126763574)

[5.2.3 Movement list 21](#_Toc126763575)

[5.2.4 Module files 22](#_Toc126763576)

[5.2.5 Movement module link 25](#_Toc126763577)

[5.2.6 Nested variable list 27](#_Toc126763578)

[5.2.7 Substitution value automatic registration 29](#_Toc126763579)

[5.2.8 Substitution value list 33](#_Toc126763580)

[5.2.9 Execution 36](#_Toc126763581)

[5.2.10 Check operation status 38](#_Toc126763582)

[5.2.11 Execution list 41](#_Toc126763583)

[6 How to write construction code 42](#_Toc126763584)

[6.1 Module description 42](#_Toc126763585)

[6.2 BackYard Content 42](#_Toc126763586)

[7 Application operation 44](#_Toc126763587)

[7.1 Maintenance 44](#_Toc126763588)

[7.2 Maintenance and Maintaining 45](#_Toc126763589)

[8 Appendix 46](#_Toc126763590)

[8.1 Module file input example/ register example 46](#_Toc126763591)

[8.2 Nested variable list flow example 60](#_Toc126763592)

Introduction

This document explains ITA’s functions how and how to operate them.

# Terraform driver overview

This chapter describes Terraform and Terraform driver.

## Terraform

Terraform is an orchestration tool that optimizes infrastructures provided by HashiCorp.

Infrastructure configurations coded in HCL language (HashiCorp Configuration Language) generates an execution plan and then runs the construction.

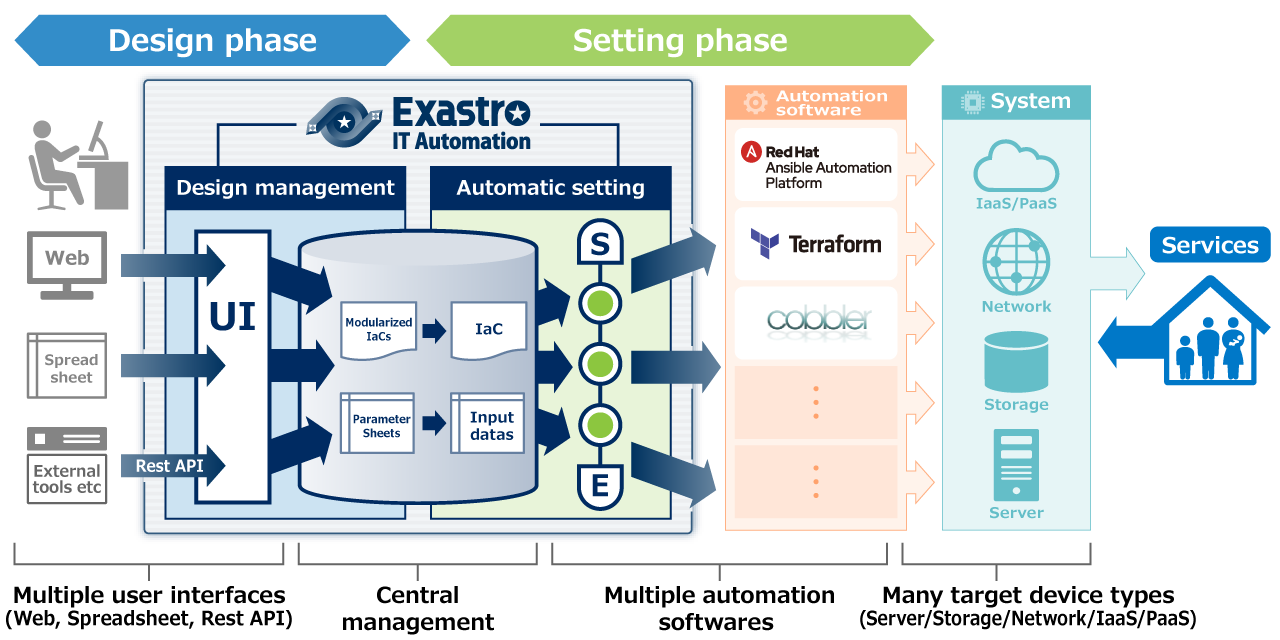
The Terraform-CLI-driver installs the Terraform CLI package provided by HashiCorp to the same server where the user’s ITA is located and uses it.

For more information about Terraform, please refer to the Terraform product manual.

## Terraform-CLI-driver

The Terraform-CLI-driver functions as an option for ITA systems, allowing Terraform installed to ITA systems to execute operations (Plan/PolicyCheck/Apply) and acquire operation logs.

Users can manage module files used to execute operations (Plan/Apply) as parts in ITA, so that they can be re-used.



**Figure 1.2-1 ITA System Overview**

Additionaly, the Terraform-CLI-driver can set variables in Modules from the screen.For details, please refer to Chapter "2. Handling variables with Terraform-CLI-driver" in this document.

# Handling variables with Terraform-CLI-driver

## Variable type

In the Terraform-CLI-driver, users can set specific variable values in Modules from the ITA configuration screen.

**※For more information on the configuring method, please refer to chapter "5.2.8Substitution value list“**

There is one module variable type that can be handled as an ITA variable.

|  |  |
| --- | --- |
| Type | Content |
| Normal variable | A variable that allows you to define one specific value for the variable name.  The variables in the Module should be written in the following format according to the variable rules of the HCL (HashiCorp Configuration Language). In this case, "xxx" is extracted from the Module file as a variable.  Users can also set type and default values. In the example below “○○” and “△△”are extracted as “type” and “default”  type and default configuration is not required.   |  | | --- | | variable “xxx” {  type = ○○  default = △△  ~Abbr~  } | |

## Extraction of variables and registration of specific values

User can register specific values by extrating variables out from module files uploaded to ITA.

The specific values of the extracted variables are registered in “5.2.8 Substitution value list”.

The registered variables and specific values are written in the terraform.tfvars file generated when the operation is run. In the file, “Variable name” is written as ”Key”, and “Specific value” as “Value”.

## Variable types

The Type of a variable can be configured within the variable.

When describing variables within Modules, make sure to follow the HCL (HashiCorp Configuration Language) variable rules.The variables handled in ITA are as follows:

For examples on how to describe them, please see” 8.1 Module file input example/ register example”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **type** | **Detailed description** | **Input order**  **Target※1** | **Member variable  Target※2** | **Type description** | **Default description** |
| string | Character string。 | × | × | string | ABC |
| number | Numeric value | × | × | number | 2022 |
| bool | True or false | × | × | bool | true |
| list | Array type | 〇 | × | list(string) | [“A”, “B”, “C”] |
| set | Array type. A unique value configuration is required.The specific value will not be checked if it is unique or not by ITA. | 〇 | × | set(number) | [1, 2, 3] |
| tuple | Array type. The user must decide which type is which number in advance.  The number of input values is already determined, so they can be selected as member variables from a pulldown menu on ITA. | × | 〇 | tuple([string, number]) | [“ABC”, 2022] |
| map | Key-value type.  If there a type that contains more than one map type configured on ITA, the user will not be able to specify KEY value from the type information.  It is therefore important that you make sure that the HCL SETTINGS are set to ON if the user plans to configure substitute values.  For more information regarding HCL settings, please see Chapter “5.2.7 Substitution value automatic registration” or “5.2.8 Substitution value list” | × | × | map(string) | { “key” = “value” } |
| object | key-value type。  ITA handles keys as Member variables. Do not include japanese characters in the key name. | × | 〇 | object({  key = number  }) | {  “key” = 2022  } |
| any | Type that fits all.  Handled the same as string type on ITA. | × | × | any | ABC |
| No description | If no "type" is described, it will be handled the same as a string type. | × | × |  | ABC |

※１…Substitute order  
The substitute order is the order of which specific values are set to variables (starting from top).  
If the variable type (or the type for the lowest variable in a hierarchy configuration) is "list" or "set", they can be configured in the Substitute value auto registration settings menu/Substitute value list menu.

Example： For “list” type variables

・tf file and registration values

|  |
| --- |
| variable "VAR\_hoge" {  type = list(string)  } |

1. Substitute value example（Substitute value auto registration settings/Substitute value list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | No input required | 1 | ABC |
| 2 | VAR\_hoge | No input required | 2 | DEF |

２．Value sent to Terraform

|  |
| --- |
| [“ABC”, “DEF”] |

Example: If the type of the variable at the lowest level of the variable hierarchy is "set"

・tf file and registration value

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  key = set(number)  })  } |

1. Substitute value example（Substitute value auto registration settings/Substitute value list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | key | 1 | 1 |
| 2 | VAR\_hoge | key | 2 | 2 |

２．Value sent to Terraform

|  |
| --- |
| {  key = [1, 2]  } |

※２…Member variable

Member variable is the key name if the variable type is "key-value". If the variable type is "object", the Member variable is "<KEY> = <TYPE>の<KEY>"

If the variable type is tuple, the Member variable is the numbered variables defined in the tuple (Numbered [0], [1], [2]...).

If the variable type is a registration target in the Nested variable list menu, the variable is numbered [0],[1],[2]... based on the maximum number of repetitions and is designated as Member variable.

For more information regarding Variable nests, please see Chapter “5.2.6　Nested variable list".

Example：If the variable type is “object”

・tf file and registration value

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  NAME = string,  IP = string  })  default = {  “NAME” = “machine\_01”,  “IP” = “127.0.0.1”  }  } |

1. Substitute value example（Substitute value auto registration settings/Substitute value list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | NAME | No input required | my\_machine |
| 2 | VAR\_hoge | IP | No input required | 192.168.0.1 |

２．Value sent to Terraform

|  |
| --- |
| {  NAME = “my\_machine”,  IP = “192.168.0.1”  } |

Example：If the variable type is target for Nested variable list.

・tf file and registration value

|  |
| --- |
| variable "VAR\_hoge" {  type = list(set(string))  default = [  [“aaa”, “bbb”],  [“ccc”, “ddd”]  ]  } |

1. Susbtitute value example （Substitute value auto registration settings/Substitute value list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | [0] | 1 | AAA |
| 2 | VAR\_hoge | [0] | 2 | BBB |
| 3 | VAR\_hoge | [1] | 1 | CCC |
| 4 | VAR\_hoge | [1] | 2 | DDD |

２．Value sent to Terraform

|  |
| --- |
| [  [“AAA”, “BBB”],  [ “CCC”, “DDD”]  ] |

Example：If the variable type is “tuple”.

・tf file and registration value

|  |
| --- |
| variable "VAR\_hoge" {  type = tuple([string, number])  default = [“aaa”, 2022]  } |

1. Susbtitute value example（Substitute value auto registration settings/Substitute value list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | [0] | No input required | bbb |
| 2 | VAR\_hoge | [1] | No input required | 2023 |

２．Value sent to Terraform

|  |
| --- |
| [“bbb”, 2023: |

# Terraform driver console menu structure

This chapter explains the ITA Console’s menu configuration.

## Menu / screen list

1. ITA Basic console menu

The list ITA Basic console menus used by the Terraform-CLI-driver is as shown below.

Table 3.1‑1 Basic console menu/screen list

|  |  |  |  |
| --- | --- | --- | --- |
| No | Menu group | Menu / Page | Description |
| 1 | ITA Basic console | Input operation list | Allows the user to maintain (view/register/update/abolish) the Operation list. |

1. **Terraform-CLI-driver console menu**

The menus in the Terraform-CLI-driver menu are as shown below.

Table 3.1‑2Terraform-CLI-driver console menu list.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Menu group | Menu/ Page | Hidden menu※１ | Description |
| 1 | Terraform | Interface information |  | Manages executed operation information. |
| 2 | Workspaces management |  | Manages Information of Workspaces used in Terraform. |
| 3 | Movement list |  | Manages a list of Movement that can be registered to the Symphony menu. |
| 4 | Module file collection |  | Manages Module files. |
| 5 | Movement-Module link |  | Manages links between Movement and Module files. |
| 6 | Nested variable list |  | Manages the maximum number of iterations of the member variables if the Variable type defined in the tf file registered in the Module file collection is "list" or "set" and "list", "set","tuple" or "object" is defined within said variables. |
| 7 | Substitution value automatic registration |  | Manages movements and variables that link items and values per operation registered in the parameter sheet menu. |
| 8 | Substitution value list |  | Manages the substitute values of the variables. |
| 9 | Operation execution |  | Allows the user to select and execute Movement and Operations. |
| 10 | Operation status |  | Displays the status of executed operations. |
| 11 | Operation list |  | Manages executed operation history. |
| 12 | Module variable link list | 〇 | Manages links between Module variables and Module files. |
| 13 | Member variable list | 〇 | Manages member variables. |
| 14 | Movement variable link list | 〇 | Manages links between Movements and Variable names. |

※1 Hidden menus are used to register and update data using the backyard function.

They are set to be hidden when the Terraform-CLI-driver is installed.

If you want to display the hidden menus, you can do so from the “Management console > Role/Menu link list” menu. For more information, please see the Management console user manual.

Note that the backyard function might not function normally if the data in the hidden menus have been changed. We recommend not changing any of the data.

# Terraform-CLI-driver workflow

This section explains the workflow for each of the Terraform consoles.

## Terraform workflow

The standard work flow for each Terraform console is as follows.

Details of each operations are described in the next section.

For information on how to use the ITA Basic Console, please refer to "User Instruction Manual\_Basic Console".

The flow to operation in Terraform are as follows.

1. **Register Input operation name**

**④ Register Operation pattern (Movement)パターン(Movement)の登録**

**⑤ Register Module file(s)**

**Required**

**Optional**

**【Legend】**

**⑥　Specify Module file(s) to Movement**

**③ Register and link Workspace**

**⑦ Set Maximum number of repetitions**

**⑧ Set Variable value**

**② Configure Interface information information**

**⑨ Execute operation**

**⑩ Check Operation status**

**⑪ Check Operation history**

* **Work flow details and references**

1. **Register Input Operation name**

Register an input operation name from the ITA Console's Operation list screen.

For details, refer to “5.1.1 Operation list”.

1. **Register interface information**

Register the interface information for the Terraform that links with ITA.

For details, refer to “5.2.1 Interface information”.

1. **Register and link Workspace.**

Register Workspace information and link with Terraform.

For details, refer to “5.2.2 Workspaces list”.

1. **Register operation pattern (Movement)**Register a movement.  
   For details, refer to “5.2.3 Movement list”.
2. **Register module files**

Register a Module file to be executed in the operation.

For details, refer to “5.2.4 Module files”.

1. **Set module files in Movement**

Specify the Module files in the registered Movement.

For details, refer to "5.2.5 Movement module link".

1. **Set Maximum number of repetitions**Set the maximum number of repetitions for the member variables.  
   For details, refer to "5.2.6　Nested variable list".
2. **Set variable value (Execute if needed)**Set the value of the variable(s) defined in the Module files registered in the corresponding Movements. The variable values does not need to be registered if variables are not being used.  
   For details, refer to "5.2.7 Substitution value automatic registration".
3. **Execution**Select and set the execution date and time, and the Operation to indicate the execution of the operation.

For details, refer to "5.2.8 Substitution value list".

1. **Check operation status**

The status of the work executed is displayed in real time.

User can also monitor work emergency stops, execution logs and error logs.  
For details, refer to "5.2.9 Execution".

1. **Check operation history**

A list of the work executed is displayed and the history can be checked.

For details, refer to "5.2.10 Check operation status".

# Terraform driver function and operation method explanation

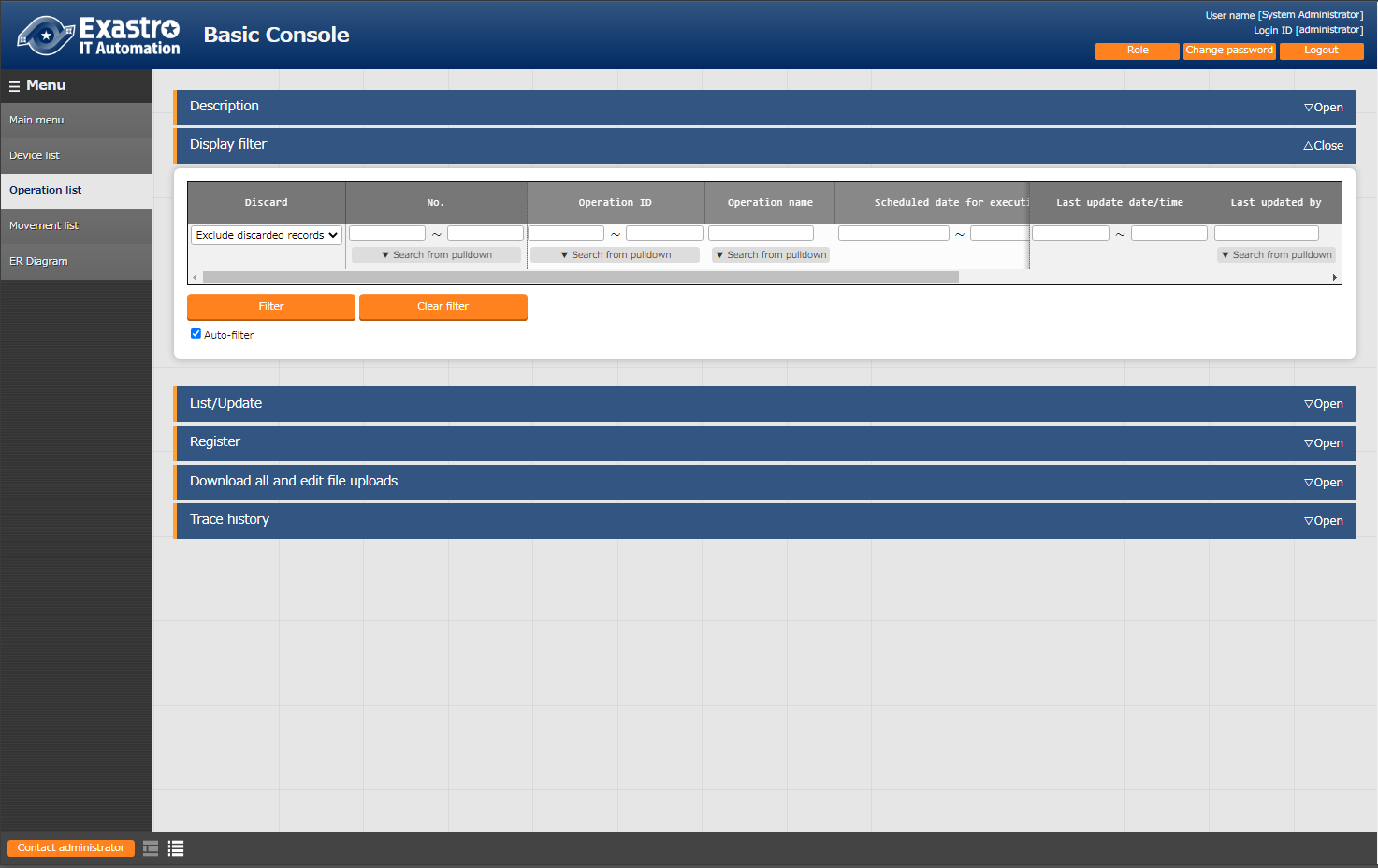
This document explains each console function used in Terraform driver.

## Basic console

This section explains a part of the ITA Basic Console.

Please refer to the ITA basic console Manual while operating the ITA Basic console.

### Operation list

1. The "Operation list" screen manages the operations on the target host that the Orchestrator executes. Select the work from the menu in the ITA basic console.

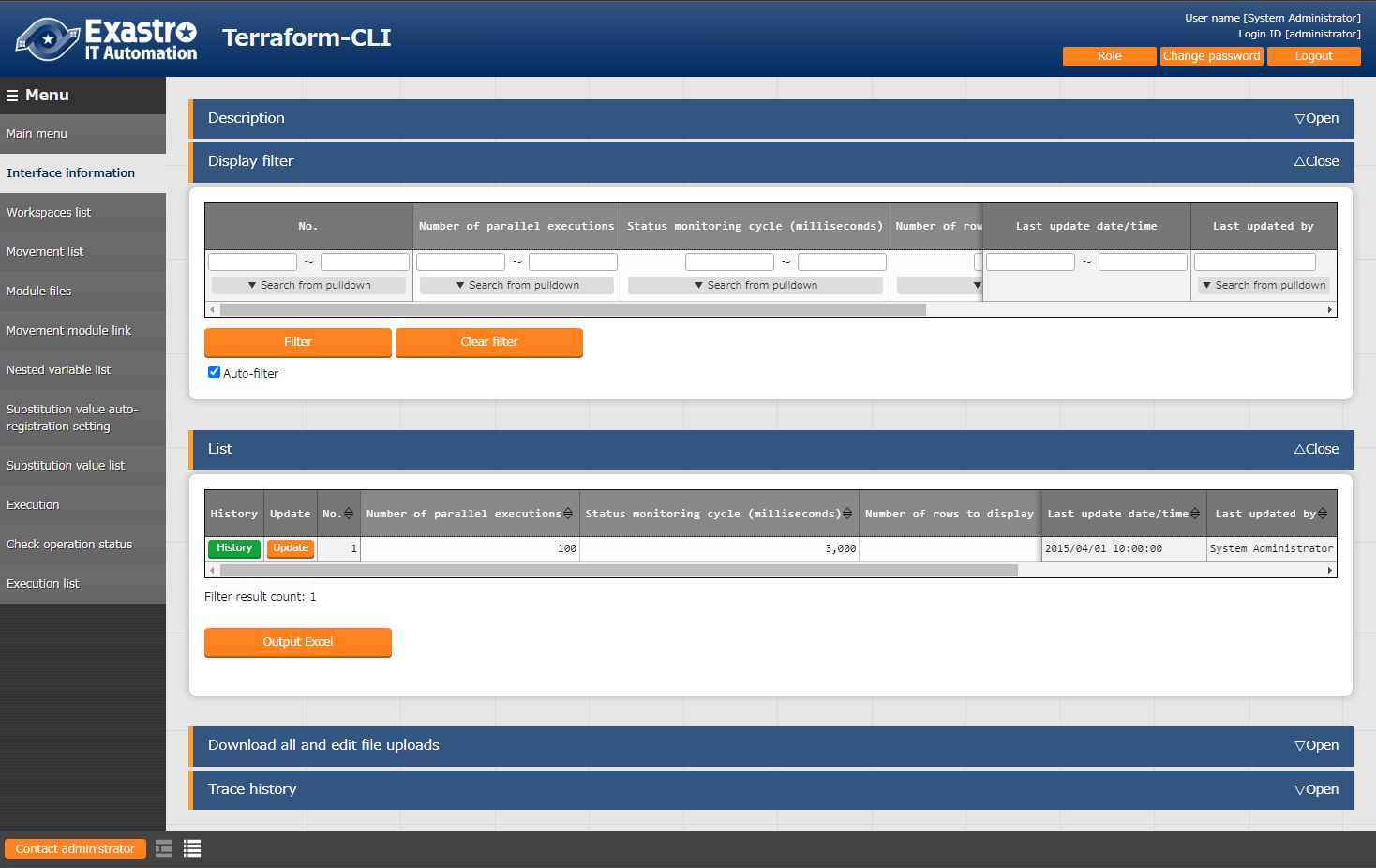
**Figure 5.1.1-1 Submenu screen (Operation list)**

For more information on how to register operations, please refer to "User Instruction Manual\_Basic Console"

## Terraform-CLI-driver console

This section describes the operation on the Terraform console.

### Interface information

1. In the “Interface Information” menu, users can maintain (view/update) information of the Terraform linked with the ITA system.

**Figure 5.2.1‒1 Submenu screen (Interface Information)**

1. Click the "List" button and then "Update" button to register Interface information.

If an operation is executed without registered interface information or with multiple records registered to it, the operation will meet an unexpected error when executed.

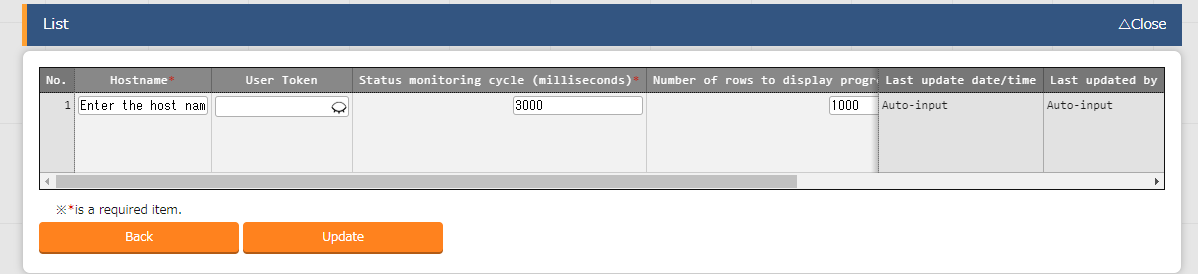


Figure 5.2.1-2 Registration screen (Interface Information)

1. The list of items on the interface information screen is as follows.

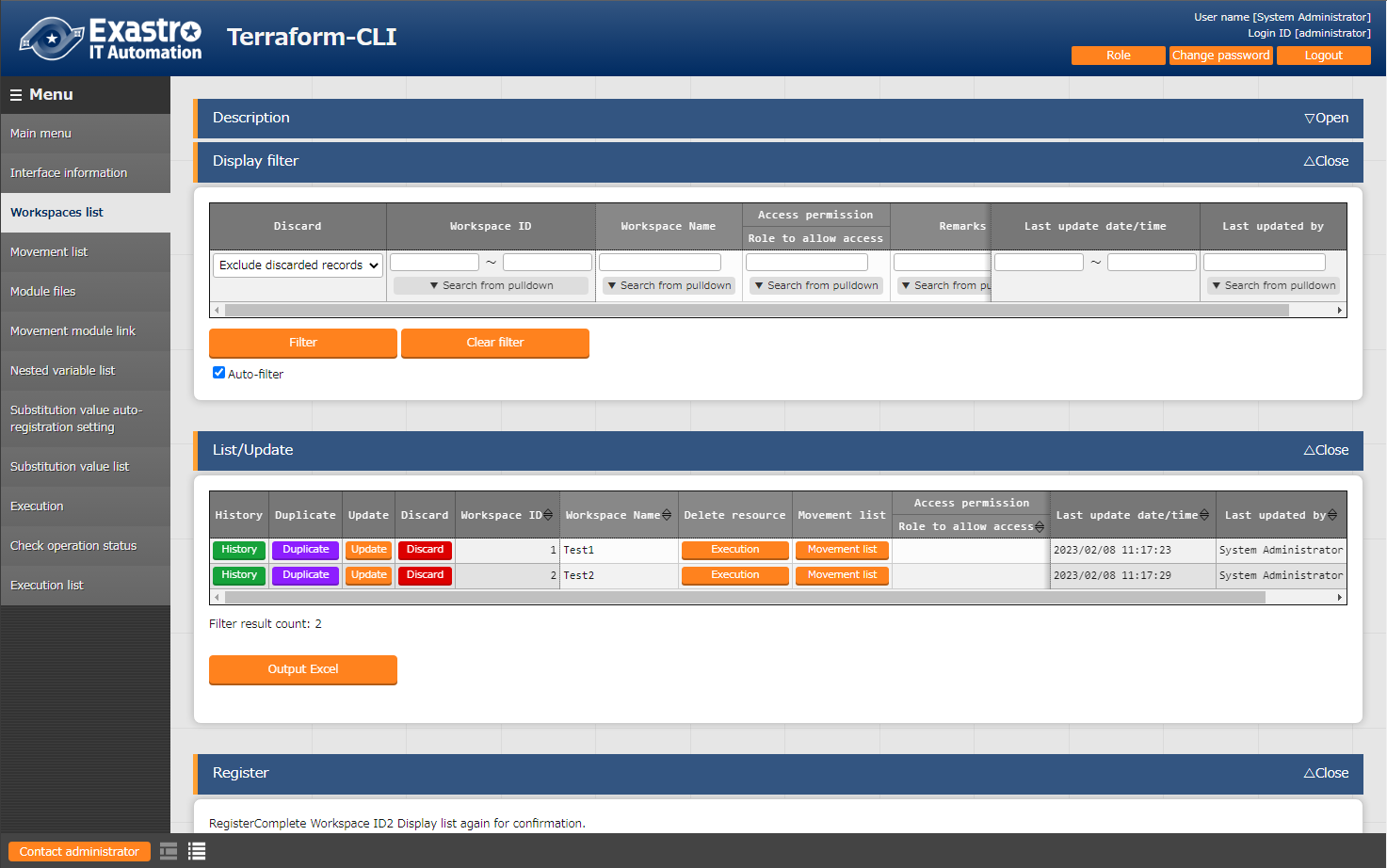
**Table 5.2.1-1 Item list（interface information）**

| **Item** | **Description** | **Input**  **required** | **Input type** | **Restriction** |
| --- | --- | --- | --- | --- |
| Number of parallel executions | Input the maximum amount of how many Movements (Terraform-CLI) that can be executed at the same time. | ○ | Manual input |  |
| Condition observation period (Unit milli second) | Enter the refresh space for the log displayed in "5.2.10 Check operation status". Usally, about 3000 milliseconds is the recommended value. | ○ | Manual input | Minimum value 1000 ms |
| Number of lines progress status displayed | Enter the maximum number of lines displayed in the progress log and error log in "5.2.10 Check operation status".Recommended value is 1000 lines. | ○ | Manual input | - |
| NULL link | If the specific value of the parameter sheet in the " Substitution value auto-registration settings" menu is NULL(blank), users can set registrations to the list to have the value NULL(blank) or not.  This value is applied when "NULL Link" (In the Substitution value auto-registration setting menu) is blank.  ・　If "Enable", any value in the parameter sheet is registered in the substitution value list.  ・　If "Disable", the value is registered in the value list only if the parameter sheet contains a value. | ○ | List selection |  |
| Remarks | Free description field. | - | Manual input | Maximum length 4000 bytes |

### Workspaces list

1. In the “Workspaces list”, users can maintain (view/register/update/abolish/delete resources) Workspaces used by Terraform.

Workspaces are used as directories for running Terraform commands.

When running operations on the same Workspace, the state files generated by Terraform are managed on a Workspace-by-Workspace basis to maintain power equality.

**Figure 5.2.22-1 Submenu screen（Workspaces list）**

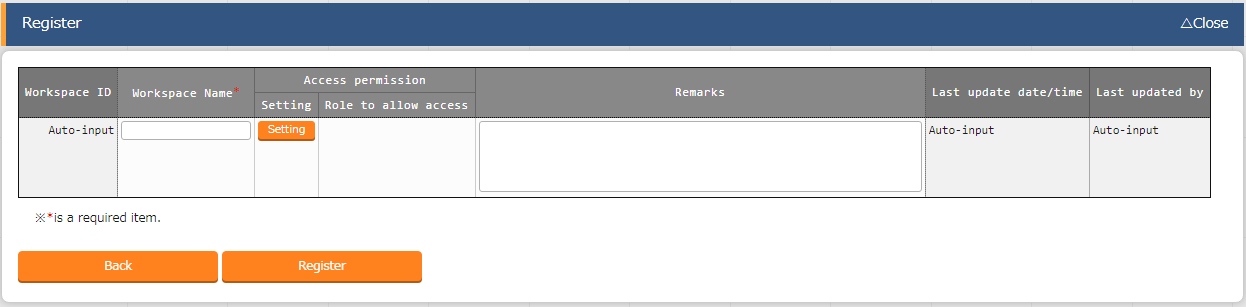
1. Click the "Register" → "Start registration" button to register Workspace infornation.

Figure 5.2.22-2 Registration screen（Workspaces list）

1. Clicking the “Delete resources” button deletes (Terraform Destroy) the target Workspace’s resources.  
   Clicking the “Movement list” button moves the user to the target Workspace’s “5.2.3 Movement list”

Figure 5.2.22-3 Terraform link (Workspaces List)

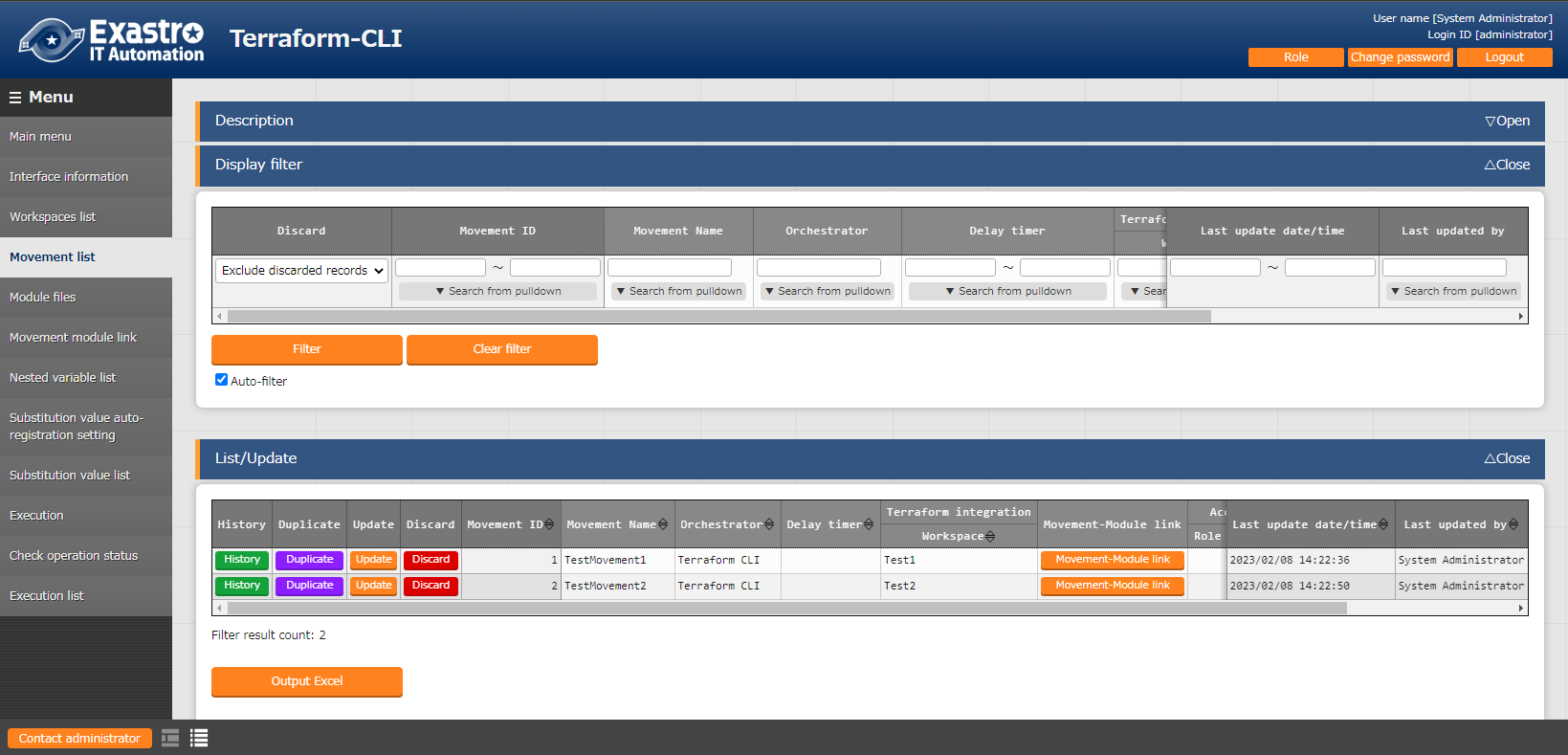
1. The items in the Workspaces list screen is as follows.

**Table 5.2.22-1 item list（Workspaces list）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Description** | **Input required** | **Input type** | **Restriction** |
| Workspace Name | Enter the name of the Workspace name.  Alphanumeric characters and symbols (\_ ,-) only (underbars and hyphens) are available. | ○ | Manual input | Maximum length 90 bytes |
| Delete Resources | This button deletes the target Workspace’s configurated/managed resources.Clicking the button will display a dialog box asking the user to confirm. Pressing “OK” will move the user to “5.2.10 Check operation status” where the Workspace’s configurated/managed resources are deleted. | - | - |  |
| Movement list | This button moves the user to “6.2.4 Movement list” | - | - |  |
| Remarks | Free description field. | - | Manual input | Maximum length 4000 bytes |

### Movement list

1. In “Movement list” users can maintain (browsing/registration/update/abolition) Movement names.

Since Movements needs to be linked to the Workspace as Terraform information, the target needs to be registered in “5.2.2 Workspaces list”.

**Figure 5.2.33.1 Submenu screen (Movement list)**

1. Click the "Register" → "Start registration" button to register Movement infornation.

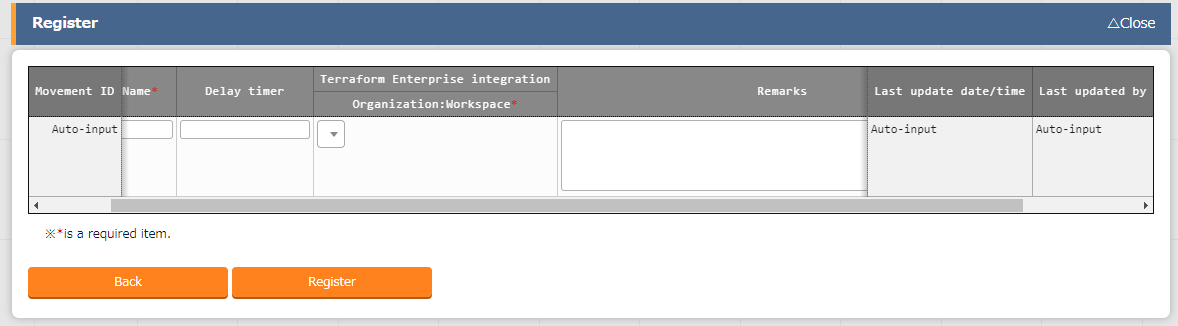
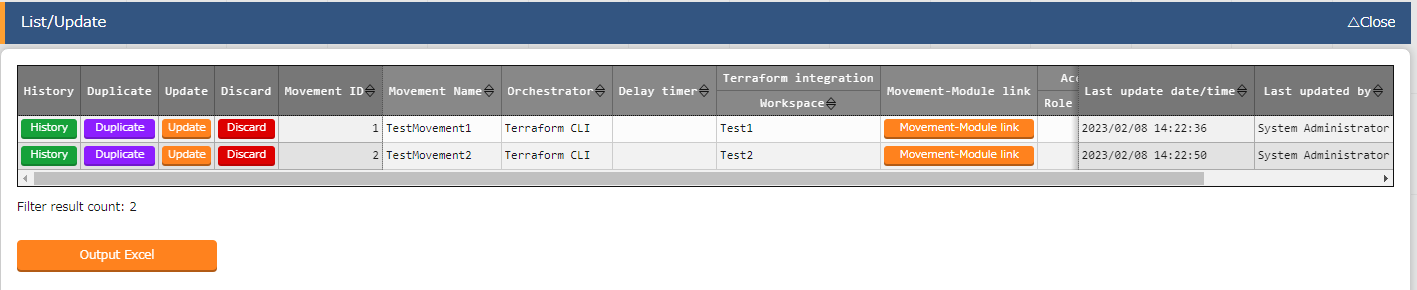


Figure 5.2.33-2 Registration screen (Movement list)

1. Clicking the Movement-Module link button will move the user to the target Movement's “5.2.5Movement module link”

**Figure 5.2.33-3 Submenu screen (Movement list)**

**(4)** The items in the Movement list screen are as follows.

**Table 5.2.3-1 Item list（Movement list）**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item** | | **Description** | **Input required** | **Input type** | **Restriction** |
| Movement Name | | Enter a name for the Movement. | ○ | Manual input | Maximum length 256 bytes |
| Orchestrator | | "Terraform" is automatically entered. | - | - | - |
| Delay timer | | Enter the specified period (1~) if you want the status to be displayed as a warning when the movement is delayed for the specified period. (Unit: minutes)  If it is not entered, no warning will be displayed. | - | Manual input | - |
| Terraform use information | Organization: Workspace | Select the Workspace registered (linked to Organization) in "5.2.2Workspaces list". | ○ | List selection |  |
| Movement-Module link | | Moves the user to “5.2.5Movement module link”. | - | - |  |
| Remarks | | Free description field. | - | Manual input | Maximum length 4000 bytes |

### Module files

1. In the “Module files” menu, users can maintenan (browsing/registration/update/abolition) user created module files.

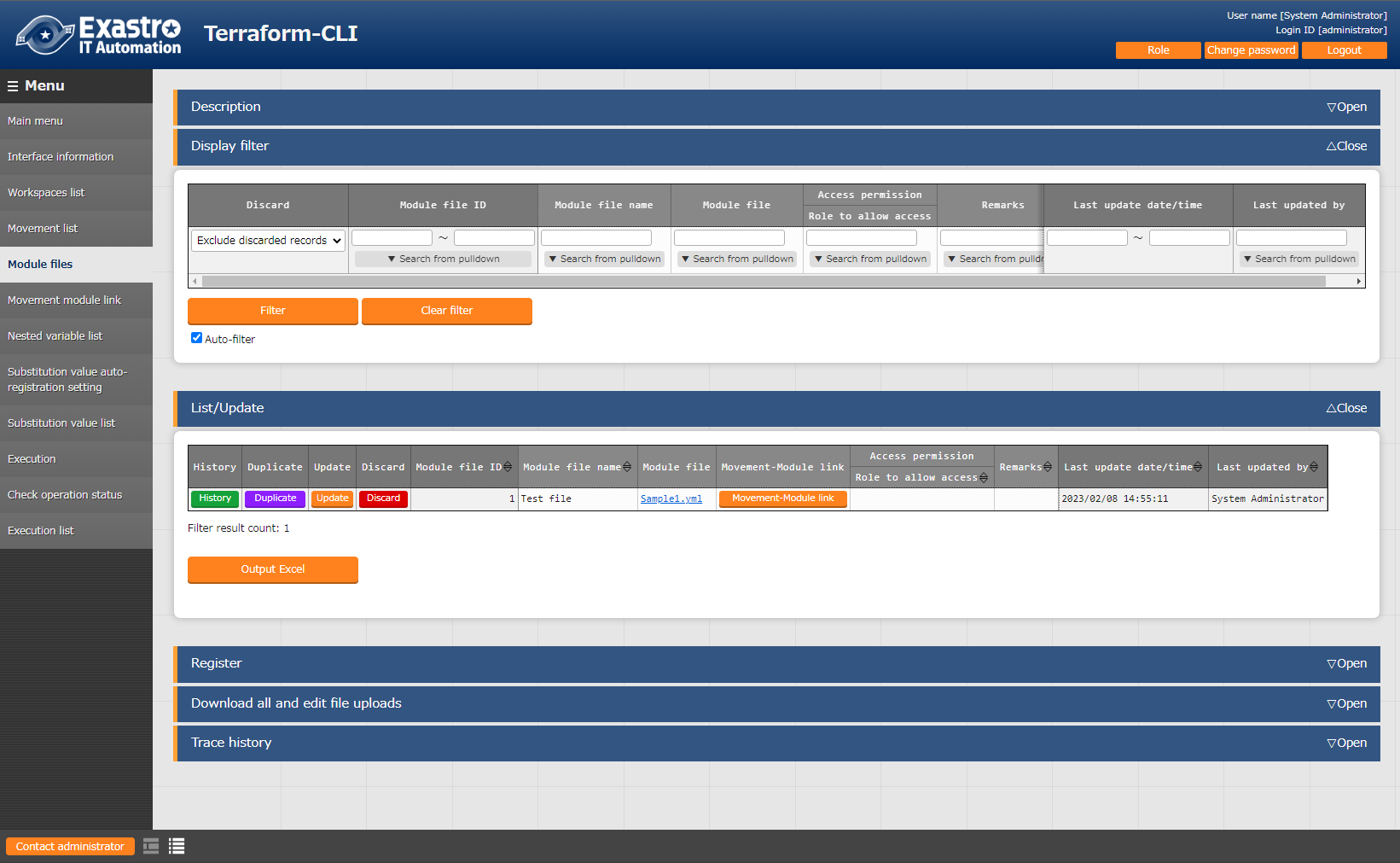
For more information regarding Module description, refer to " 6.1 Module description.

Figure 5.2.44-1 Submenu screen (Module files)

1. Click the "Register" → "Start registration" button to register Module infornation.

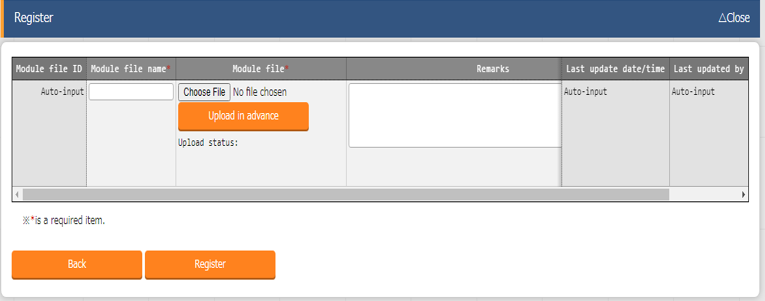
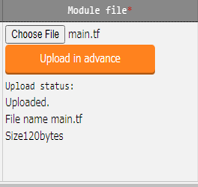
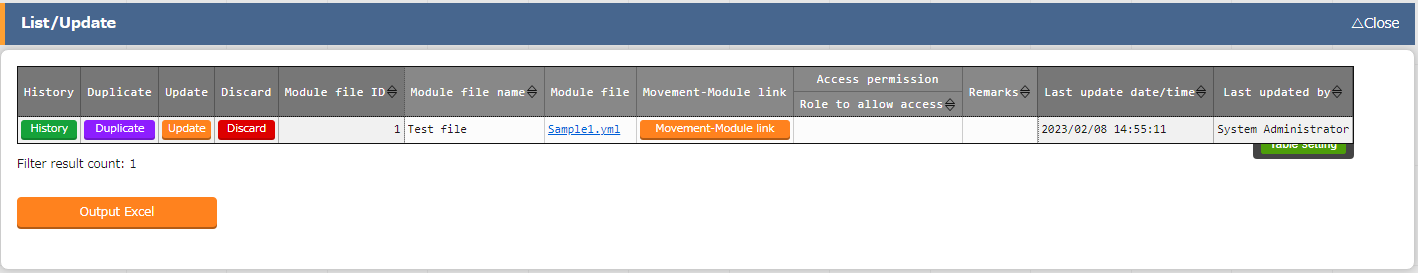


Figure 5.2.44-2 Registration screen (Module files)

Make sure to press the “Upload in advance” button (①) before registering the files to ITA.  
Confirm that the file name displayed (②) is correct and press the “Register” button. 

1. Clicking the Movement-Module link button will move the user to the target Movement’s “5.2.5 Movement module link”.

**Figure 5.2.44-2 Submenu screen (Module files)**



1. The items found in the Module files menu are as follows.

**Table 5.2.44-2 Item list (Module files)**

| **Item** | **Description** | **Input required** | **Input type** | **restriction** |
| --- | --- | --- | --- | --- |
| Module files name | Enter a name for the Module file. This is the name that will be used for the Module file in ITA. | ○ | Manual input | Maximum length 256 bytes |
| Module files | Upload the created Module files. | ○ | File selection | Maximum size 4G bytes |
| Movement-Module link | Moves the user to “6.2.10 Movement-Module link” |  |  |  |
| Remarks | Free description field. | - | Manual input | Maximum length 4000 bytes |

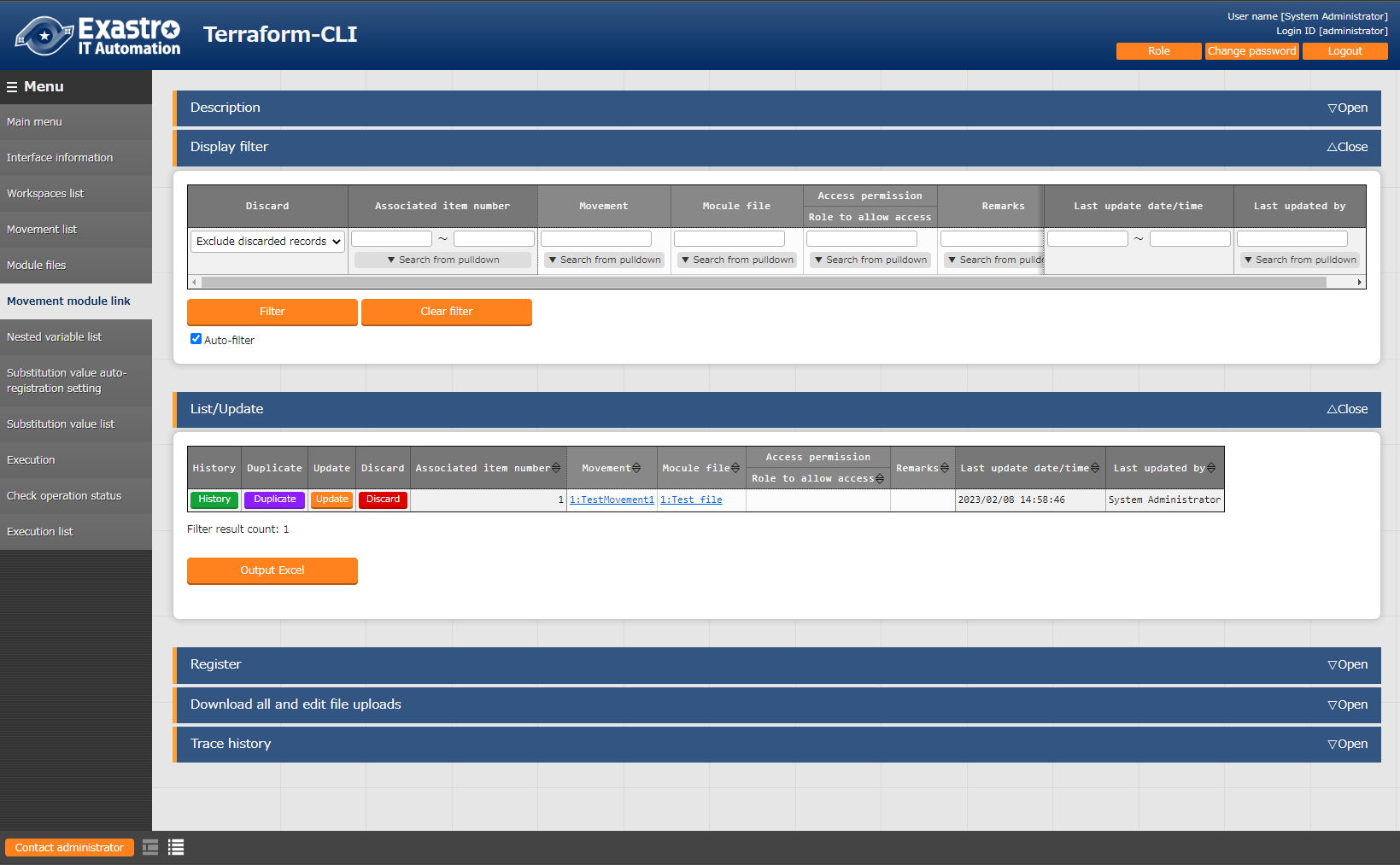
1. The variables defined in the Module files are extracted by a BackYard process.  
   The extracted variables can then have specific values registered to them in menus such as “5.2.7 Substitution value automatic registration” and “5.2.8 Substitution value list”.

Note that the extraction does not happen in real time, meaning it make take some time before they can be used as variables in menus such as

**※1 For more information regarding when the extraction happens, please see “③Change the startup period” under ” 7.2 Maintenance and Maintaining”**

### Movement module link

1. In　“Movement module link”, performs maintence (browsing/register/update/ abolition) of the module files executed in the Movement.

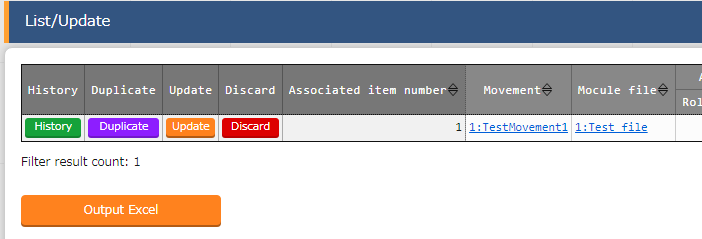
Multiple Module files can be linked to the Movement.

**Figure 5.2.55-1 Submenu screen（Movement module link）**

1. Click the "Register" → "Start registration" button to register Movement Module infornation.



Figure 5.2.55-2 Registration screen （Movement module link）

1. Clicking the Movement URL will move the user to “5.2.3 Movement list”.  
   Clicking the Module file URL will move the user to “5.2.4 Module files”.   
   **Figure 5.2.55-3 Sunmenu screen （Movement module link）**
2. The item list of Movement module link is as follows

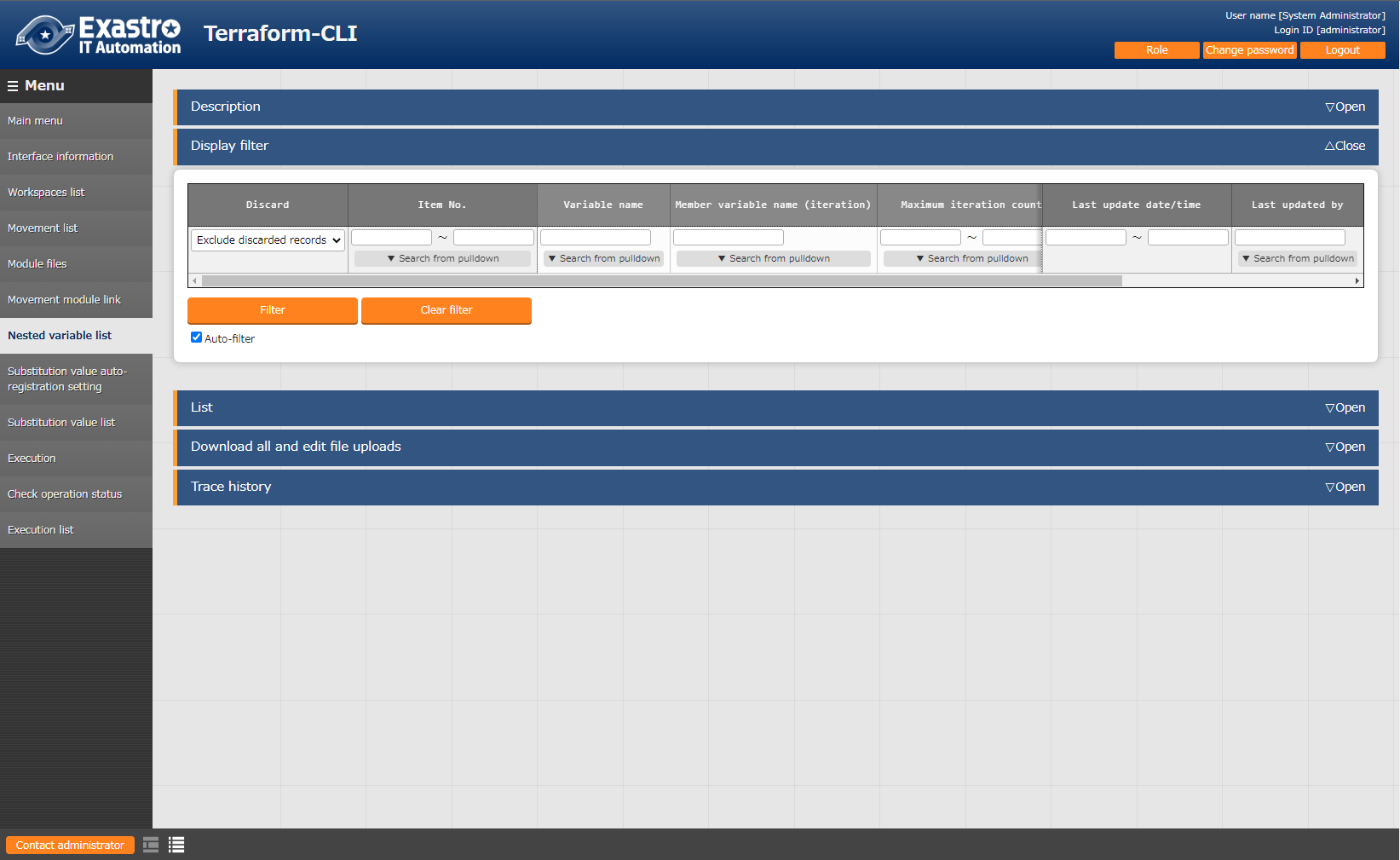
**Tab**l**e 5.2.55-1 Item list（Movement module link）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Description** | **Input required** | **Input type** | **Restriction** |
| Movement | Select a Movement registered in "5.2.3 Movement list". | ○ | List selection | - |
| Module files | Select a Module file registered in "5.2.4 Module files". | ○ | List selection | - |
| Remarks | Free description field. | - | Manual | Maximum length 4000 bytes |

### Nested variable list

1. In the "Nested variable list" menu, users can view and change the maximum number of repetitions of the member variables if the Variable type defined in the tf file registered in the Module file collection is "list" or "set" and "list", "set","tuple" or "object" is defined within said variables.

Items in this menu cannot be registered, discarded or restored as BackYard manages the records based on the Module file collection.

For more information, please see " 8.1 Module file input example/ register example”.

For more information regarding the examples of flows with the Nested variable list, please see 8.2 Nested variable list flow example”.

**Module file and Nested variable list registration**

・Module file（tf file） and registration value

|  |
| --- |
| variable "VAR\_hoge" {  type = list(  object({  IP = string,  NAME = string  })  )  default = [  { “IP” = “127.0.0.1”, “NAME” = “machine\_01” },  { “IP” = “127.0.0.2”, “NAME” = “machine\_02” }  ]  } |

・Nested variable list

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Maximum repetitions** |
| 1 | VAR\_hoge |  | 2 |

Figure 5.2.66‑1 Submenu screen（Nested variable list）

1. Press the “List”>”Update” button to edit the maximum amount of repetitions.

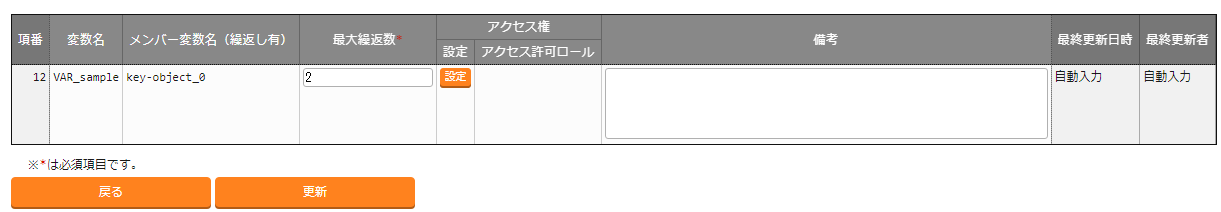


Figure 5.2.66‑2 Update screen（Nested variable list）

1. The item list is as following

Figure 5.2.66‑3 Item list （Nested variable list）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Description** | **Input required** | **Input method** | **Restrictions** |
| Variable name | Displays the variable(s) used in the file registered with the Movement-Module link. | - | No input required |  |
| Member Variable name（With repetitions） | If the Nest variable list target is a Member variable, the Member variable name will be displayed. The Member variables are displayed concatenating with variables with "." for each stage. | - | No input required |  |
| Maximum amount of repetitions | Input a number between 1 and 99,999,999.  The initial value is set to the number obtained from the defauly value in the tf file.  If there is not "default" in the tf file, the value "1" will be set.  If it is not last updated by "Terraform variable update procedure", it is not possible to change the value from updating the Module file. | ○ | Manual input | Input value:   1～99,999,999 |
| Remarks | Free description field | - | Manual input | Maximum length 4000 bytes |

※Initial registrations and repetition updates are not happening in real-time, so it might take a couple of minutes before the variables can be used in "5.2.7 Substitution value automatic registration“ and "5.2.8 Substitution value list"

The chapter " 7.2 Maintenance and Maintaining" contains more information regarding the timing of when they are updated.

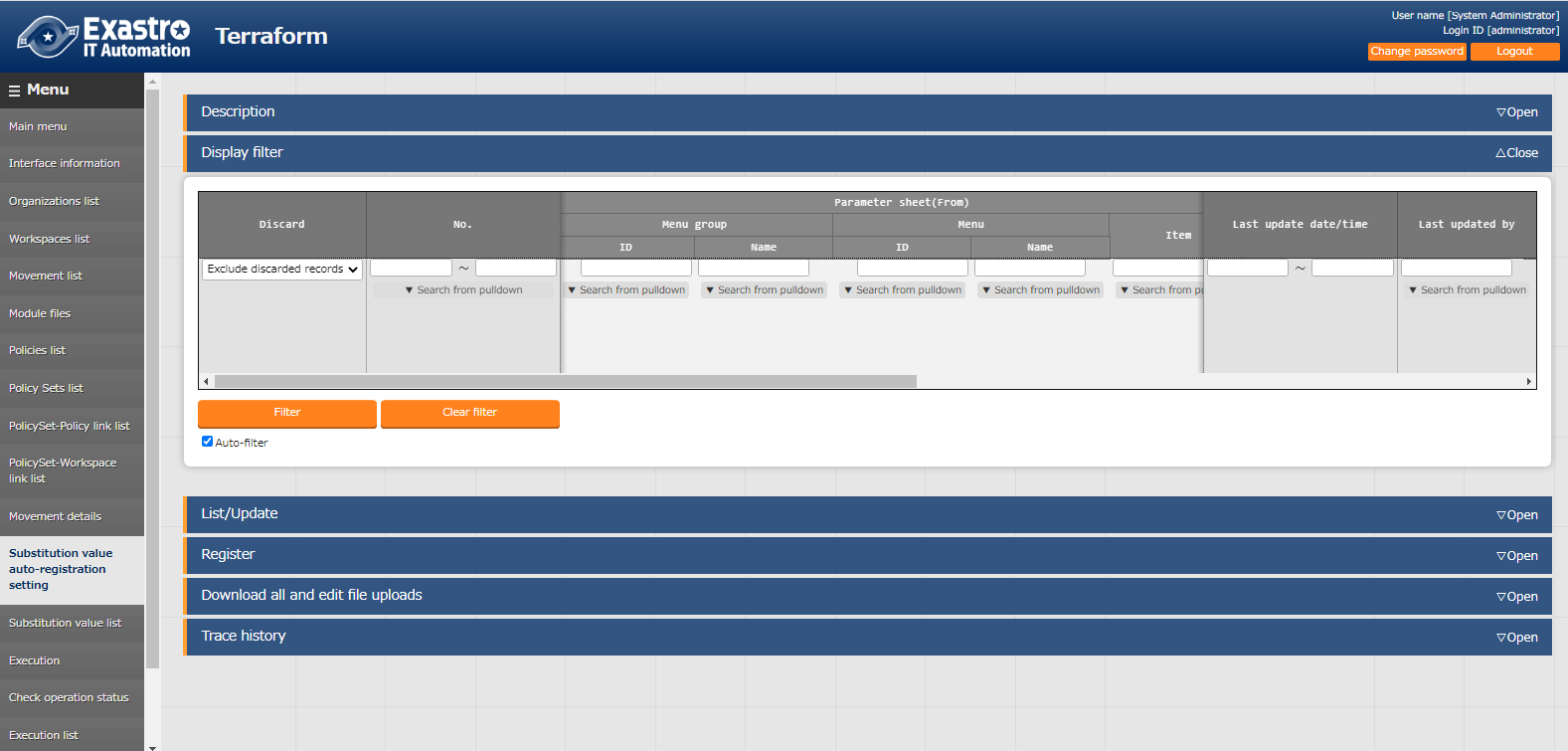
1. Access permission role

The permission roles set for variable nesting management are set to the permission roles of the Module material collection in which the relevant variable is defined.

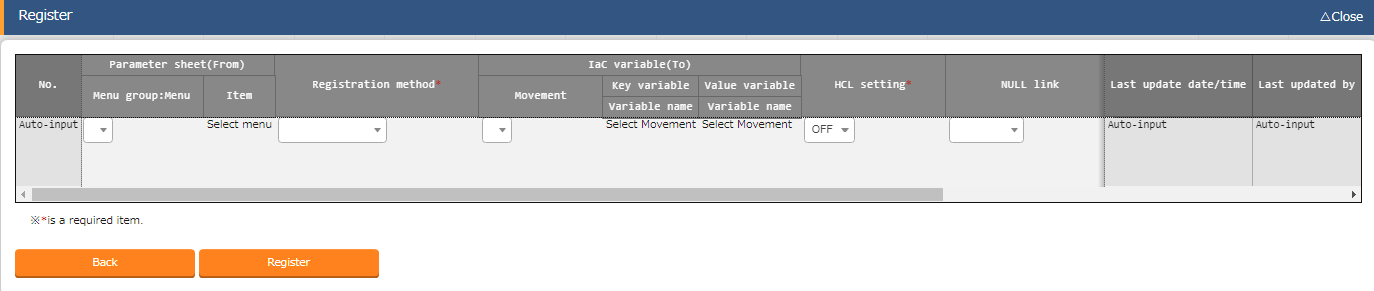
### Substitution value automatic registration

* 1. In "Substitution value automatic registration", users can link Parameter sheets created with the Menu creation tool (With Operation) and Movement variables.

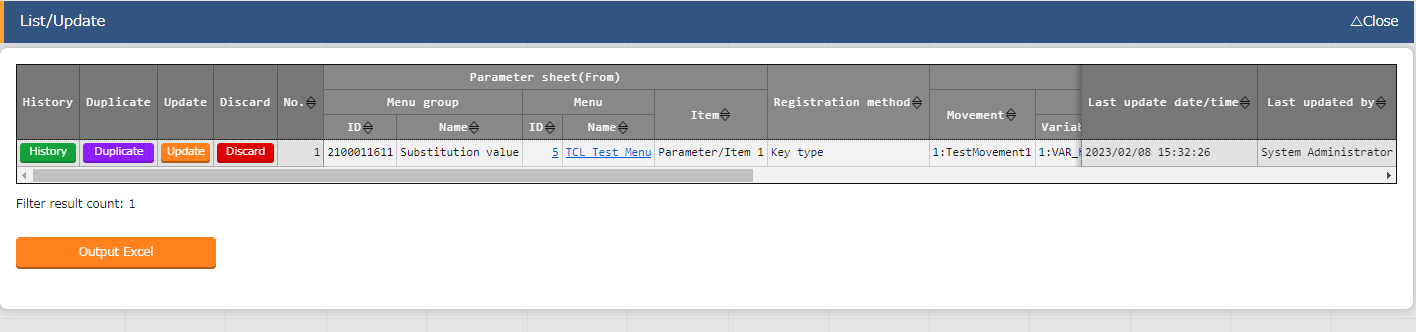
The registered information will be reflected to the Substitution value list by the BackYard Process.

The reflection rules is mentioned in” 6.2BackYard Content (2)Automatic Substitute Value Registration Settings”. 

**Figure 6.2.11-1 Sub-menu screen (Substitution value automatic registration)**

* 1. Click the "Register" → "Start registration" button to register Substitute value infornation.

**Figure 6.2.11-2 Registration Screen (Substitution value automatic registration)**

* 1. Clicking the Menu ID or the Menu name link will move the user to the target menu.  
     

**Figure 6.2.11-3 Submenu screen (Substitution value auto-registration settings)**

* 1. The list of items on the registration screen is as follows.

**Figure 6.2.11-1 Item list (Substitution value automatic registration)**

| **Column** | | **Description** | **Input**  **required** | **Input type** | **Restriction** |
| --- | --- | --- | --- | --- | --- |
| Menu group ：menu | | The parameter sheet (with operation) created by the menu creation function is displayed.  Select the appropriate parameter sheet. | ○ | List selection |  |
| Item | | The items in the selected parameter sheet are displayed.Select the target item. | ○ | List selection |  |
| Registration method | | Value Type: Select when you want the set value of the item to be the specified value of the linked variable.  Key Type: Select when you want the name of the item to be the specified value of the linked variable.  If the set value of an item is blank, it will not be linkable.  Key-Value Type: Select when you want the name of the item (Key) and the set value (Value) to be used as the specified value of a linked variable. | ○ | List selection |  |
| Movement | | The movement registered in the Movement list is displayed.Select Movement. | ○ | List selection |  |
| Key  information | Variable name | The variables used in the materials registered in the Movement module link are displayed.  Select the variable you want to link to the specific value in the Key type. | ○ | List selection | Required if the registration method is Key or Key-Value type. |
| Member variable | If the selected variable name has Member variables, the Member variables will be displayed.  Select Member variable. | Depends on Variable name | List selection |  |
| Substitute order | Only required if multiple specific values can be set to a variable name. The substitute order of the specific values are input starting from (1~). The values are substituted rising from the input value. | Depends on Variable name | Manual input | Must be Blank or a positive integer |

| Value  information | Variable name | The variables used in the materials registered in the Movement module link are displayed.  Select the variable you want to link to the specific value in the Value type. | ○ | List selection | Required if the registration method is Key or Key-Value type. |
| --- | --- | --- | --- | --- | --- |
| HCL Settings | Select "OFF" or "ON".  BackYard processing takes over the selected value when it is reflected in the subsititution value list. |  | List selection | If the selected variable name is “map” type, the HCL settings must be set to “ON”. Make sure that the HCL settings matches the Operation, Movement and/or variable names are the same. |
| Member variable | If the selected variable name has Member variables, the Member variables will be displayed.  Select Member variable. | Depends on Variable name | List selection |  |
| Substitute order | Only required if multiple specific values can be set to a variable name. The substitute order of the specific values are input starting from (1~). The values are substituted rising from the input value. | Depends on Variable name | Manual input | Must be Blank or a positive integer |
| NULL link | | If the specific value of the parameter sheet in the "Substitution value auto-registration setting" is NULL(blank), users can set registrations to the list to have the value NULL(blank) or not.  This value is applied when "NULL Link"(In the Substitution value auto-registration setting menu) is blank.  ・If "Enable", any value in the parameter sheet is registered in the substitution value list.  ・If "Disable", the value is registered in the value list only if the parameter sheet contains a value.  ・If blank, the "NULL link" value of the interface information is applied. | - | List selection | - |
| Remarks | | Free description field. | - | Manual input | Maxumum length 4000 byte |

※If the selected registration method is set to “Key type”, the HCL settings are set to OFF when displayed in the Substitute value list.

※When configuring Member values, make sure to configure the variable’s other member variable’s specific values.  
Other values that did not have substitute value set to them will not use default values.

These values must  
be configured

E.g.

・tf file and register values

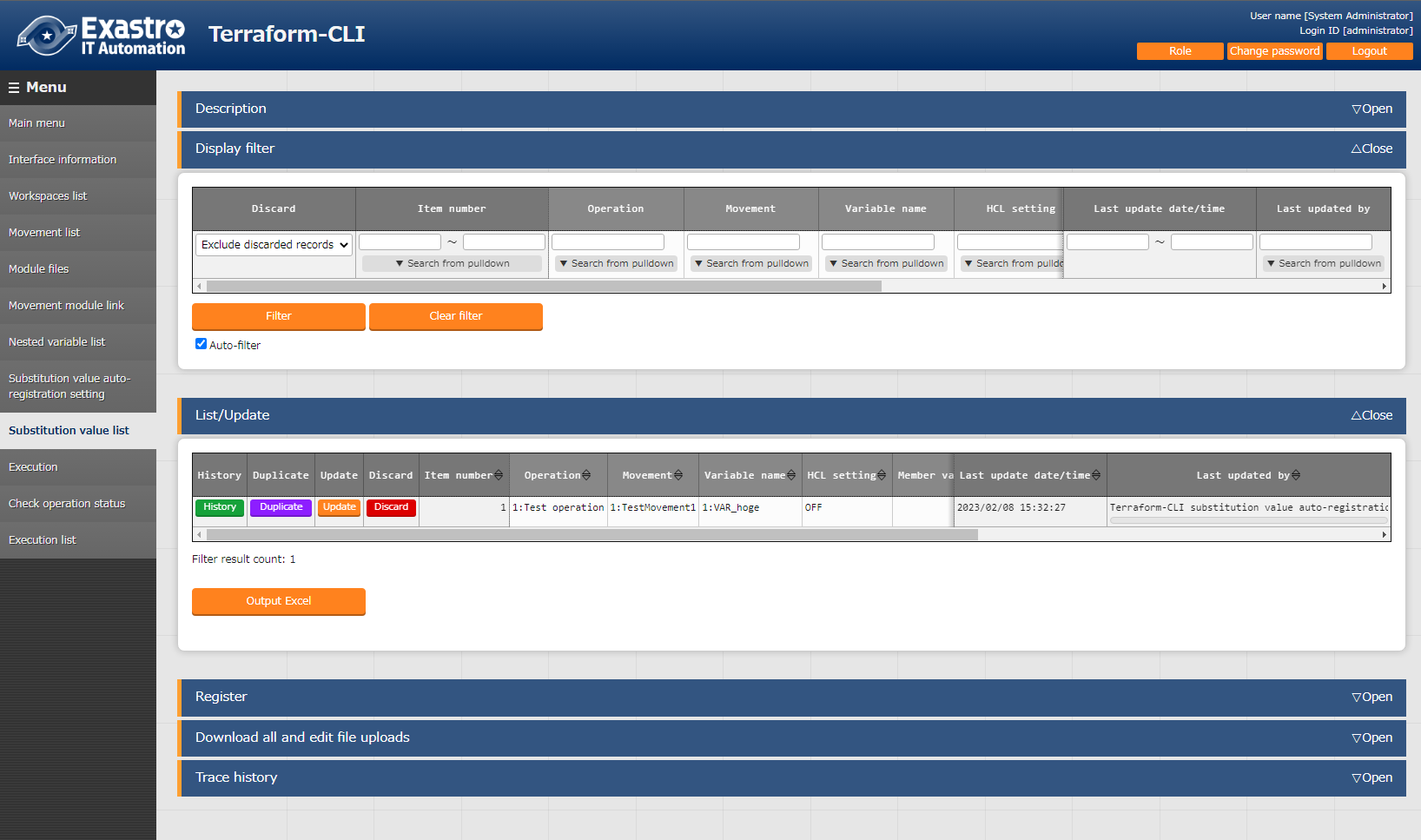
|  |
| --- |
| variable "VAR\_hoge" {  type = object({  IP = string  NAME = string  })  default = {  IP = “127.0.0.1”  NAME = “machine01”  }  } |

・Substitute value example(Substitute value auto registration settings/ Substitute value list)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | IP | No input required | 192.168.0.1 |
| 2 | VAR\_hoge | NAME | No input required |  |

### Substitution value list

1. In the “Substitution value list” menu, users can maintain (browse/register/update/abolish) the specific values that substitutes the variables within the Module used by Movements.



**Figure 5.2.88-1 Submenu screen （Substitution value list）**

1. Click the "Register" → "Start registration" button to register Substitute values.

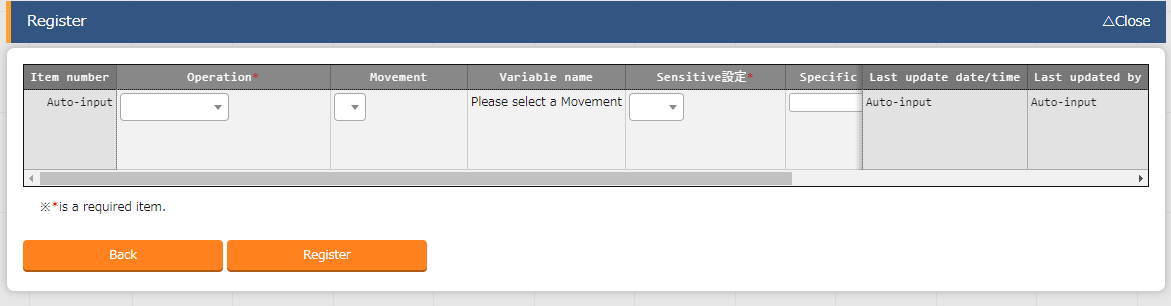


Figure 5.2.88-2 Registration screen（Substitution value list）

The variables in substitution value list are reflected from the file information registered in   
"5.2.4 Module files”

**※Please refer to the timing of reflection is described in "(3) Change of starting period" of " 7.2 Maintenance and Maintaining ".**

1. For the variables registered in the substitution value list, the "variable name" is registered as "Key" and "Specific value" is registered as "Value" for the Variables managed in the Workspace on the Terraform side when the operation is executed.

If "HCL settings" is set to "ON", it will be registered with "HCL" checked.

If "Sentive settings" is set to "ON", it will be registered with "Sentive" checked.

1. The list of items for assignment value list is as follows.

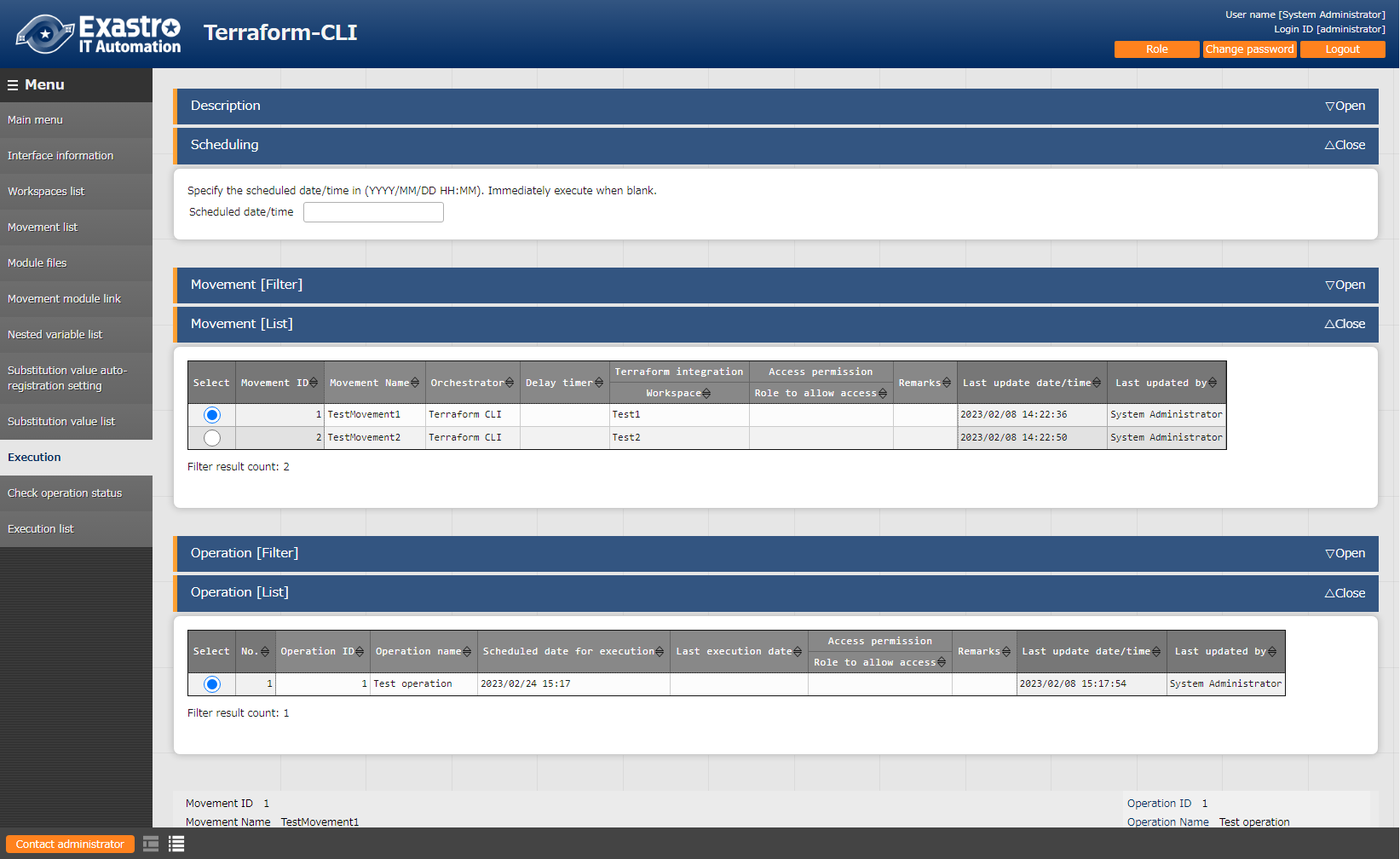
**Table 6.2.12-1　Item list（Substitution value list）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Description** | **Input required** | **Input type** | **Restriction** |
| Operation | Select the target Operation. | ○ | List selection | - |
| Movement | Select the target Movement. | ○ | List selection | - |
| Variable name | From the Module files registered in the Movement module link, the name of the variable attached to the selected Movement is displayed. Select a variable. | ○ | List selection | - |
| HCL settings | Select "OFF" or "ON".  If "ON" is selected, "HCL" will be enabled for Variables when they are registered in the Workspace on the Terraform side.  Use this setting when configuring a variable to a value that is not a character string. | ○ | List selection |  |
| Variable name | The variables used in the materials registered in the Movement module link are displayed.  Select the variable you want to link to the specific value in the Key type. |  |  |  |
| Substitute order | Only required if multiple specific values can be set to a variable name. The substitute order of the specific values are input starting from (1~). The values are substituted rising from the input value. | Depends on Variable name | Manual input | Must be Blank or a positive integer |
| Default value | Display the specific value linked to the variable within “default” |  |  |  |
| Sensitive settings | Select “OFF” or “ON”.  If "ON" is elected, the specific value will be encrypted won't be displayed on ITA.  Also keep in mind that when registering any variables to the workspace on the Terraform side, "Sensitive" for that variable will be enabled and specific values will not be displayed. | ○ | List selection |  |
| Specific value | Enter the specific value of the variable to use in Operations/Movements. | ○ | Manual input | Maximum length 8192 bytes |
| Remarks | Free description field. | - | Manual input | Maximum length 4000 bytes |

5.2.3

### Execution

1. The “Execution” menu allows users to execute operations. The users can use the radio buttons to select which Movement and Operation to execute. Pressing the “Execute” button on the bottom of the screen will move the user to “5.2.10 Check operation status” where the operation is run.



**Figure 6.2.14-1 Submenu screen（Execution）**

1. **Specify scheduled date/time**

Users can reserve Execution and Plan confirmations by entering the "Scheduled date/time".

Only date/time can be registered for "Schedule date/time"

1. **Specify Movement**

Select the Movement registered in the "5.2.3 Movement list".

1. **Specify Operation**

Select the Operation registered in the "5.1.1 Operation list".

1. **Execution**

Clicking the "Execute" button will move the user to ""5.2.10 Check operation status"" and execute the operation.

"Apply" will automatically be executed after the Plan and/or the PolicyCheck are completed.

1. **Plan confirmation**

Clicking the "Plan Confirmation" button will similarly to clicking the "Execute" button, start the execution. However, "Discard Run" will be applied to RUN after the Plan and PolicyCheck are completed, and "Apply" will not be executed.

If a module with an output block is run with Conductor,

the contents of the output block will be saved to the deta relay storage path(shared movement directory) as a json file.

This file allows users to use a value output by Terraform from a different Movement (in the same conductor).

File path:

[Data relay storage path]/[Conductor instance ID]/terraform\_output\_[Operation No].json

Example：/exastro/data\_relay\_storage/conductor/0000000001/terraform\_output\_0000000001.json

Data relay storage path -[Conductor]-[Conductor interface information ] - [Data relay storage path]

Conductor instance ID・・・[Conductor]-[Conductor list] - [Conductor instance ID]（The 10 numbers from the left.）

Operation No.・・・ [Terraform]-[Execution list] - [Operation No.]（The 10 numbers from the left.）

Description

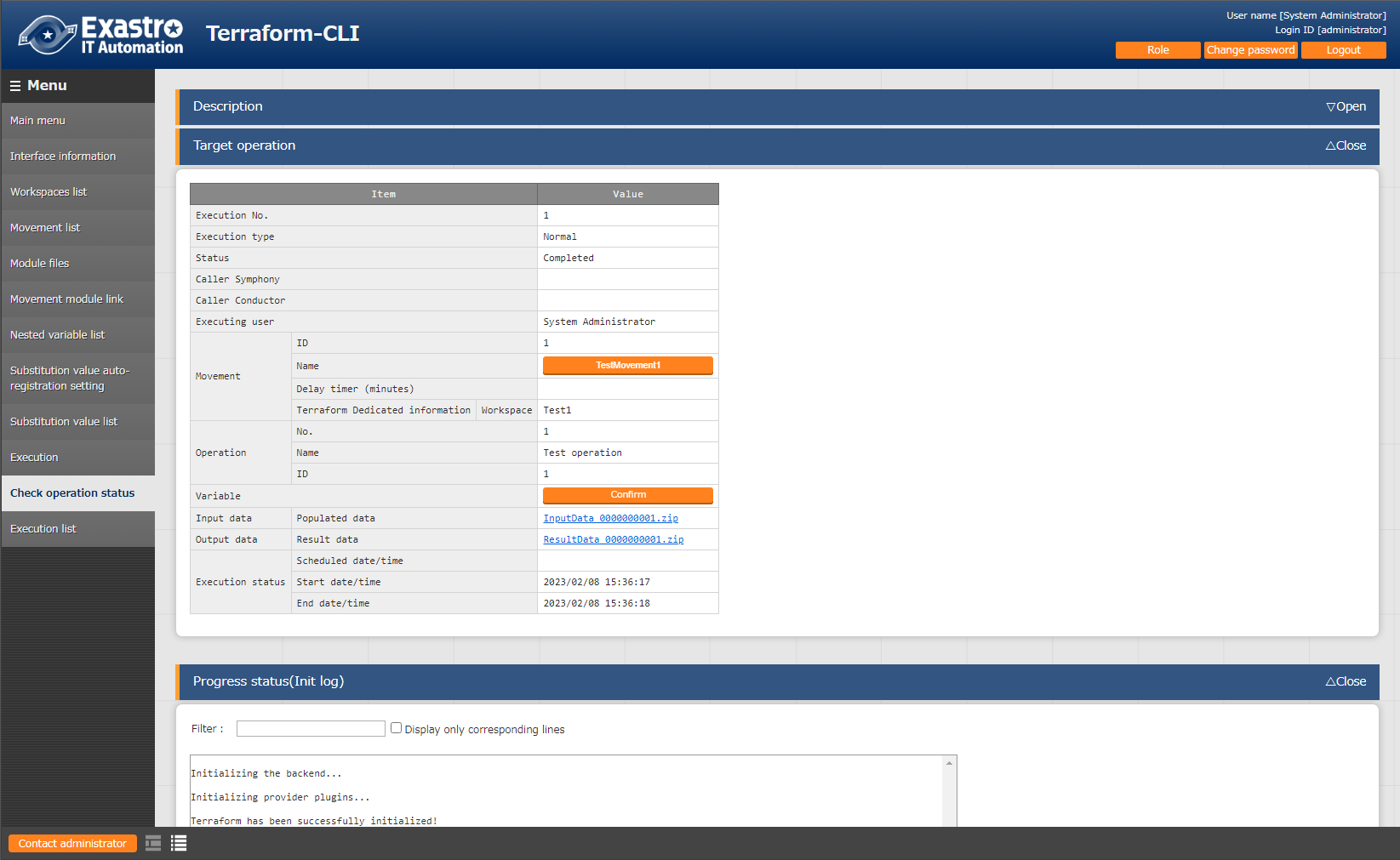
|  |
| --- |
| variable "VAR\_sample" {  type = string  default = "sample\_string"  }  output "output\_sample" {  value = "${var.VAR\_sample}"  } |

Output

|  |
| --- |
| {  "output\_sample": "sample\_string"  } |

### Check operation status

1. This page allows users to monitor the operation execution status.



**Figure 5.2.100-1 Submenu screen （Check operation status）**

1. **Display of execution status**

The Status displayed matches the Execution status of the operation.

The "Execution Type" will contain "Plan Confirmation" for plan confirmations, and "Normal" for other cases.

Execution log for Progress (Plan Log), Progress (PolicyCheck Log) and Progress (Apply Log) executed in Terraform are displayed in Plan/ PolicyCheck / Apply.

If the status displays “unexpected error”, the Progress status (Error log) will display a message if there might be an issue with the contents registered to “5.2.1 Interface information” or with registered web contents.

Other errors will not be displays in the Progress status (error log), but in the Process log.

※Please refer to "④ Log file name" in " 7.2 Maintenance and Maintaining" for the process log.

“Call Symphony”, displays which Symphony was executed. It will be blank if you execute it directly from the Terraform driver or from Conductor.

In “Call Conductor”, displays which Conductor was executed. It is blank if you execute it directly from the Terraform driver or from Symphony.

"Execution user" displays the user logged in when the "Execute" button was pressed from the execution menu.

※If the “Execution type” is set to “Delete resources”, the following items will not be configured.

・Called Symphony

・CalledConductor

・Movement（ID、name、Data delay（min））

・Operation（No.、name、ID）

・Substitute value

・Input data

1. **Substitution value confirmation**

By clicking the "confirmation" button, "5.2.8 Substitution value list" will display and the substitution value filtered by the operation and Movement of operation target will be displayed.

1. **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.

1. **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 exeuction 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.
2. **Input data**Users can download executed module files and a zip file containing a list of policy materials and the configured substitute values in Json format. The files are as following:

**Table 5.2.100-1 Input data files**

|  |  |
| --- | --- |
| **File name** | **Description** |
| (Input Module file name) | Contains all the input module files. Stored directly under the zip file. |
| Terraform.tfvars | Contains “Variable name(Key)” and “Specific value(Value)” for the set substitute values.  Does not contain data for items with Secure settings set to ON. |

1. **Result data**

User can download execution logs, error logs, and state files generated by Terraform.

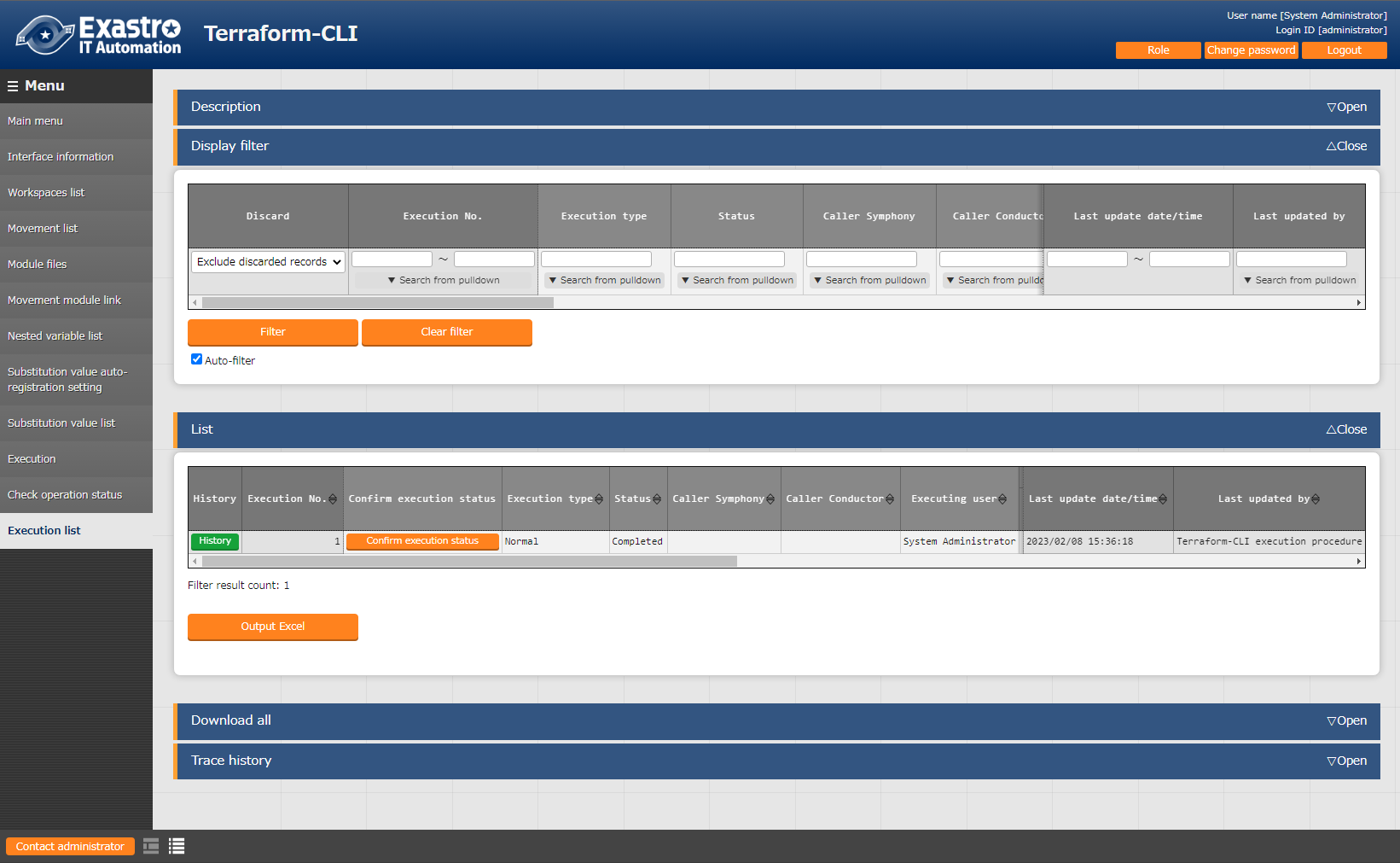
**Table 5.2.100-2 Result data files**

|  |  |
| --- | --- |
| **File name** | **Description** |
| Init.log | Log file that contains the contents output to the progress status(init log) |
| plan.log | Log file that contains the contents output to the progress (plan) log. |
| apply.log | Log files that contains the contents output to the progress (apply) log. |
| error.log | Log files that contains the contents output to the progress (error) log. |
| result.txt | File that describes the proress status used by BackYard when executing operations. |
| .terraform.lock.hcl | File generated by Terraform. Contains provider nad module information. |
| terraform.tfstate | State file generated by Terraform. Is saved as an encrypted file. |
| terraform.tfstate.backup | A backup of the state file generated by Terraform. Is saved as an encrypted file. |

### Execution list

1. This menu allows users to view operation history.

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

Clicking the "Check execution status" button, will move the user to “5.2.10 Check operation status" where they can view a detailed execution status.

**Figure 5.2.111-1 Submenu screen（Execution list）**

# How to write construction code

Describes the description of Module and Policy in Terraform driver.

## Module description

Module files are written in HashiCorp's own language called HCL (HashiCorp Configuration Language).

For more information on HCL, see the Terraform product documentation.

## BackYard Content

1. **Automatic variable registration**

This function extracts variables from the Module files registered in "5.2.4 Module files".

Please refer to "2.1 Variable types" for more information regarding "Variable extraction rules".

Additionally, the timing of the extraction depends on the startup cycle of the "Automatic Terraform Variables Registration" process.

1. **Automatic Substitute Value Registration Settings**

The Information from the movement and variables linked to the set values of Operation itens in the target parameter sheets is reflected in the Substitute Value list.

The timing of the extraction depends on the startup cycle of the "Terraform Auto Registration Settings" process.

The Substitute value list can be updated by multiple operators. If the last update was performed by another operator, it will not be reflected.

If you want to reflect the data of the Automatic Substitute value registration settings, please delete the corresponding record in the Substitute Value list.

The rules for reflecting the substitution value list are written below.

1. **When the information registered in the Substitution value automatic registration is reflected in the Substitution value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Substitution value list status** | **Without applicable record** | **With applicable record** | | | **Record abolishing** |
| **= Specific value** | **≠ Specific value** | |
| **Last update** | |
| **BackYard process** | **Other operators** |
| Reflected to substitution value list | Add new record | - | Update specific values for the applicable record | - | Abolition record revival |

※Applicable record: Operation + Movement + Variable name + HCL configuration + Records with the same access permission.

1. **Information not registered in substitution value automatic registration (registered only for the substitution value list) is reflected in the subsitution value list.**

|  |  |  |
| --- | --- | --- |
| **Substitution**  **value list status** | **With applicable record** | |
| **Last update** | |
| **BackYard process** | **Other operators** |
| Reflected to  substitution value list | Record abolishing | - |

1. **HCL settings**

The value of the "HCL setting" configured for substitution value automatic registration is set to the same value when reflected in the substitution value list.

1. **Sensitive settings**

If the item in the Link-target parameter sheet is set to "Password", the "Sensitive settings" will be set to "ON" when reflected in the Substitution value list.

1. **Access permission roles**

The access permission role of the operation set in the record of the link-target parameter sheet and the access permission role of the Movement set in the record of automatic assignment value registration are referred to, and all matching access permission roles are set when reflected in the Substitute value list.

If no permission role is set for either of them (blank), the blank space will also be set when reflected in the Substitute Value list.

Additionally, if there are no single matching permission roles, no record will be created in the Substitute Value list.

# 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.

## Maintenance

The files required to start/stop/restart Terraform driver processes are as follows.

|  |  |
| --- | --- |
| **Description** | **Target file name** |
| Terraform operation execution monitoring  Execute the unexecuted Operation. | ky\_terraform\_execute-workflow.service |
| Terraform operation execution monitoring  Check the status of executing work and acquire logs. | ky\_terraform\_checkcondition-workflow.service |
| Terraform variable automatic registration  Remove variables from uploaded Module files. | ky\_terraform\_varsautolistup-workflow.service |
| Terraform auto registration settings  The information configured for substitution value automatic registration is reflected in the substitution value list. | ky\_terraform\_valautosetup-workflow.service |

The target file is stored in “/usr/ lib/ystemd/system”.

The method of Starting/Stopping/Restarting a process is as follows. Please execute the command with root permission.

1. Start process

# systemctl start ky\_terraform\_execute-workflow.service

1. Stop process

# systemctl stop ky\_terraform\_execute-workflow.service

1. Restart Process

# systemctl restart ky\_terraform\_execute-workflow.service

Please replace each target file name with a start/stop/restart.

## Maintenance and Maintaining

1. Change to NORMAL level

Rewrite line 8 of the following file with "DEBUG" to NORMAL.

　Log level configuration file: <installation directory> /ita-root/confs/backyardconfs/ita\_env

1. Change to DEBUG level

　Rewrite line 8 of the following file with "DEBUG" to NORMAL.

　 Log level configuration file: <installation directory>/ita-root/confs/backyardconfs/ita\_env

1. Change the startup period

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

　With exceptions, use the default value for the startup period.

ExecStart=/bin/sh ${ITA\_ROOT\_DIR}/backyards/common/ky\_loopcall-php-procedure.sh /bin/php /bin/php ${ITA\_ROOT\_DIR}/backyards/terraform\_driver/ky\_terraform\_execute-workflow.php ${ITA\_ROOT\_DIR}/logs/backyardlogs **5** ${ITA\_LOG\_LEVEL} > /dev/null 2>&1

After rewriting the file, **it enables after the process is restarted.**

1. Log file name

|  |  |
| --- | --- |
| **Process name** | **Log file name** |
| ky\_terraform\_execute-workflow | ky\_terraform\_execute-workflow\_YYYYMMDD.log |
| ky\_terraform\_checkcondition-workflow | ky\_terraform\_checkcondition-workflow\_YYYYMMDD.log |
| ky\_terraform\_varsautolistup-workflow | ky\_terraform\_varsautolistup-workflow\_YYYYMMDD.log |
| ky\_terraform\_valautosetup-workflow | ky\_terraform\_valautosetup-workflow\_YYYYMMDD.log |

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

# Appendix

## Module file input example/ register example

The following section are examples of inputting and registering module files in relation to the flow number in "4.1 Terraform Workflow".

* + - 1. Simple pattern
  1. string type

|  |
| --- |
| variable "VAR\_hoge" {  type = string  default = "def-string"  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | No input required | No input required | sample\_str |

　　　　　　Value registered to Terraform’s Variables

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | sample\_str |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. number type

|  |
| --- |
| variable "VAR\_hoge" {  type = number  default = 2022  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | No input required | No input required | 2023 |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | 2023 |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. bool type

|  |
| --- |
| variable "VAR\_hoge" {  type = bool  default = true  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | No input required | No input required | false |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | false |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. list type

|  |
| --- |
| variable "VAR\_hoge" {  type = list(string)  default = [“aaa”, “bbb”, “ccc”]  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | No input required | 1 | AAA |
| 2 | VAR\_hoge | OFF | No input required | 2 | BBB |

**TerraformのVariables Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“AAA”, “BBB”] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. set type

|  |
| --- |
| variable "VAR\_hoge" {  type = set(string)  default = [“aaa”, “bbb”, “ccc”]  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | No input required | 1 | AAA |
| 2 | VAR\_hoge | OFF | No input required | 2 | BBB |

**TerraformのVariables Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“AAA”, “BBB”] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. tuple type

|  |
| --- |
| variable "VAR\_hoge" {  type = tuple([string, number])  default = [“aaa”, 2022]  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | [0] | No input required | AAA |
| 2 | VAR\_hoge | OFF | [1] | No input required | 2023 |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [“AAA”, 2023] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. map type

The map type needs the HCL SETTINGS to be ON.

|  |
| --- |
| variable "VAR\_hoge" {  type = map(string)  default = {  “key” = “value”  }  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | ON | No input required | No input required | { “aaa” = “bbb”} |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | { “aaa” = “bbb”} |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. object type

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  IP = string,  NAME = string  })  default = {  “IP” = “127.0.0.1”,  “NAME” = “machine01”  }  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | IP | No input required | 192.168.0.1 |
| 2 | VAR\_hoge | OFF | NAME | No input required | my\_machine |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | {  “IP” = “192.168.0.1”,  “NAME” = “my\_machine”  } |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. any type

|  |
| --- |
| variable "VAR\_hoge" {  type = any  default = “def-any”  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | key | No input required | aaa |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | aaa |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* 1. Description with no type

|  |
| --- |
| variable "VAR\_hoge" {  default = “def-string”  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | No input required | No input required | aaa |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | aaa |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

* + - 1. Complex pattern

1. list typearraylist

**⑫ Configure maximum number of repetitions**

|  |
| --- |
| variable "VAR\_hoge" {  type = list(list(string))  default = [  [“a”, “b”, “c”],  [“d”, “e”, “f”]  ]  } |

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetition）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 2 |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | [0] | 1 | AAA |
| 2 | VAR\_hoge | OFF | [0] | 2 | BBB |
| 3 | VAR\_hoge | OFF | [1] | 1 | CCC |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  [“AAA”, “BBB”],  [“CCC”]  ] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

1. list type objects

**⑫ Configure maximum number of repetitions**

|  |
| --- |
| variable "VAR\_hoge" {  type = list(  object({  NAME = string  AGE = number  })  )  default = [  { “NAME” = “Tanaka”, “AGE” = 30 },  { “NAME” = “Yamamoto”, ” AGE” = 26 }  ]  } |

**Nested variable list**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetition）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 2 |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | [0].NAME | No input required | HONDA |
| 2 | VAR\_hoge | OFF | [0].AGE | No input required | 20 |
| 3 | VAR\_hoge | OFF | [1].NAME | No input required | OGIKUBO |
| 4 | VAR\_hoge | OFF | [1].AGE | No input required | 50 |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  {“NAME” = “HONDA”, “AGE” = 20}  {“NAME” = “OGIKUBO”, “AGE” = 50}  ] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

1. object’s list objects

**⑫ Configure maximum number of repetitions**

|  |
| --- |
| variable "VAR\_hoge" {  type = object({  FRUIT = list(object{  NAME = string, PRICE = number  }),  VEGETABLE = l ist(object{  NAME = string, PRICE = number  })  })  default = {  FRUIT = [  { NAME = “Apple”, PRICE = 120 },  { NAME = “Orange”, PRICE = 80 }  ],  VEGETABLE = [  { NAME = “Eggplant”, PRICE = 100 },  { NAME = “Tomato”, PRICE = 200 }  ]  }  } |

**Nested variable list**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetition）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge | FRUIT | 2 |
| 2 | VAR\_hoge | VEGETABLE | 2 |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | FRUIT.[0].NAME | No input required | Kiwi |
| 2 | VAR\_hoge | OFF | FRUIT.[0].PRICE | No input required | 200 |
| 3 | VAR\_hoge | OFF | FRUIT.[1].NAME | No input required | Grape |
| 4 | VAR\_hoge | OFF | FRUIT.[1].PRICE | No input required | 1000 |
| 5 | VAR\_hoge | OFF | VEGETABLE.[0].NAME | No input required | Lettuce |
| 6 | VAR\_hoge | OFF | VEGETABLE.[0].PRICE | No input required | 100 |
| 7 | VAR\_hoge | OFF | VEGETABLE.[1].NAME | No input required | Cabbage |
| 8 | VAR\_hoge | OFF | VEGETABLE.[1].PRICE | No input required | 110 |

**⑬ Configure Variable values**

**⑥ Register Module file**

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | {  FRUIT = [  { NAME = “Kiwi”, PRICE = 200 },  { NAME = “Grape”, PRICE = 1000 }  ],  VEGETABLE = [  { NAME = “Lettuce”, PRICE = 100 },  { NAME = “Cabbage”, PRICE = 110 }  ]  } |

**⑭ Execute Operation**

* + - 1. Special types
  1. Map type under list type

The map type needs the HCL SETTINGS to be ON.

|  |
| --- |
| variable "VAR\_hoge" {  type = list(map(string))  default = [{  “key” = “value”  }]  } |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | ON | No input required | No input required | [{ “aaa” = “bbb”}] |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [{ “aaa” = “bbb”}] |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑭ Execute Operation**

## Nested variable list flow example

The following example is reference to the Terraform CLI driver workflow’s “5.2.6　Nested variable list”

* + - 1. Increase amount of maximum amount of repetitions.

Member variable added from updating the Nested variable list.

|  |
| --- |
| variable "VAR\_hoge" {  type = list(object({ IP = string, NAME = string }))  default = [  { “IP” = “127.0.0.1”, NAME = “machine01”},  { “IP” = “127.0.0.2”, NAME = “machine02”}  ],  } |

**Nested variable list（When registering）**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetition）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 2 |

**Nested variable list（When updating）**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetition）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 3 |

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | [0].IP | No input required | 192.168.1.1 |
| 2 | VAR\_hoge | OFF | [0].NAME | No input required | yamamoto |
| 3 | VAR\_hoge | OFF | [1].IP | No input required | 192.168.1.2 |
| 4 | VAR\_hoge | OFF | [1].NAME | No input required | suzuki |
| 5 | VAR\_hoge | OFF | [2].IP | No input required | 192.168.1.3 |
| 6 | VAR\_hoge | OFF | [2].NAME | No input required | tanaka |

**⑬ Configure Variable values**

**⑥ Register Module file**

**⑫ Configure maximum amount of repetitions**

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  { “IP” = “192.168.1.1”, “NAME” = “yamamoto” },  { “IP” = “192.168.1.2”, “NAME” = “suzuki” },  { “IP” = “192.168.1.3”, “NAME” = “tanaka” }  ] |

**⑭ Execute Operation**

* + - 1. Decreasing maximum amount of repetitions

|  |
| --- |
| variable "VAR\_hoge" {  type = list(object({ IP = string, NAME = string }))  default = [  { “IP” = “127.0.0.1”, NAME = “machine01”},  { “IP” = “127.0.0.2”, NAME = “machine02”},  { “IP” = “127.0.0.3”, NAME = “machine03”}  ],  } |

**Nested variable list（When registering）**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetitions）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 3 |

**Nested variable list（When updating）**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item No.** | **Variable name** | **Member Variable name（With repetitions）** | **Maximum amount of repetitions** |
| 1 | VAR\_hoge |  | 2 |

**⑥ Register Module file**

999

Updating the Nested variable list made it not possible to select Member variable [2].IP and [2].NAME.

**Substitute value auto registration settings/Substitute value list**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item No.** | **Variable name** | **HCL SETTINGS** | **Member variable** | **Substitute order** | **Specific value** |
| 1 | VAR\_hoge | OFF | [0].IP | No input required | 192.168.1.1 |
| 2 | VAR\_hoge | OFF | [0].NAME | No input required | yamamoto |
| 3 | VAR\_hoge | OFF | [1].IP | No input required | 192.168.1.2 |
| 4 | VAR\_hoge | OFF | [1].NAME | No input required | suzuki |
| 5 | VAR\_hoge | OFF | [2].IP | No input required |  |
| 6 | VAR\_hoge | OFF | [2].NAME | No input required |  |

**Value registered to Terraform’s Variables**

|  |  |
| --- | --- |
| **Key** | **Value** |
| VAR\_hoge | [  { “IP” = “192.168.1.1”, “NAME” = “yamamoto” },  { “IP” = “192.168.1.2”, “NAME” = “suzuki” },  ] |

**⑬ Configure Variable values**