



# ITA\_利用手順マニュアル

Ansible-driver

—第1.5版—

## 免責事項

本書の内容はすべて日本電気株式会社が所有する著作権に保護されています。

本書の内容の一部または全部を無断で転載および複製することは禁止されています。

本書の内容は将来予告なしに変更することがあります。

日本電気株式会社は、本書の技術的もしくは編集上の間違い、欠落について、一切責任を負いません。

日本電気株式会社は、本書の内容に関し、その正確性、有用性、確実性その他いかなる保証もいたしません。

## 商標

- ・ LinuxはLinus Torvalds氏の米国およびその他の国における登録商標または商標です。
- ・ Red Hatは、Red Hat, Inc.の米国およびその他の国における登録商標または商標です。
- ・ Apache、Apache Tomcat、Tomcatは、Apache Software Foundationの登録商標または商標です。
- ・ Ansibleは、Red Hat, Inc.の登録商標または商標です。
- ・ AnsibleTowerは、Red Hat, Inc.の登録商標または商標です。

その他、本書に記載のシステム名、会社名、製品名は、各社の登録商標もしくは商標です。

なお、® マーク、TM マークは本書に明記しておりません。

※本書では「Exastro IT Automation」を「ITA」として記載します。

## 目次

目次	2
はじめに	4
1 Ansible driver 概要	5
1.1 Ansible について	5
1.2 AnsibleTower について	5
1.3 Ansible driver について	6
2 Ansible driver での変数取り扱い	7
2.1 変数の種類	7
2.2 変数の抽出および具体値登録	9
2.3 代入値登録による変数の扱い	10
3 Ansible driver コンソールメニュー構成	11
3.1 メニュー/画面一覧	11
4 Ansible driver 利用手順	13
4.1 作業フロー	13
4.1.1 Ansible-Legacy 作業フロー	13
4.1.2 Ansible-Legacy Role 作業フロー	16
4.1.3 Ansible-Pioneer 作業フロー	19
5 Ansible driver 機能・操作方法説明	22
5.1 基本コンソール	22
5.1.1 機器一覧	22
5.1.2 投入オペレーション一覧	26
5.2 Ansible 共通コンソール	27
5.2.1 インターフェース情報	27
5.2.2 Ansible Tower ホスト一覧	31
5.2.3 グローバル変数管理	33
5.2.4 テンプレート管理	35
5.2.5 ファイル管理	39
5.3 Ansible-Legacy/Role/Pioneer コンソール	42
5.3.1 OS 種別マスタ	42
5.3.2 Movement 一覧	44
5.3.3 プレイブック素材集 (Ansible-Legacy のみ)	47
5.3.4 ロールパッケージ管理 (Ansible-Legacy Role のみ)	49
5.3.5 対話種別リスト (Ansible-Pioneer のみ)	51
5.3.6 対話ファイル素材集 (Ansible-Pioneer のみ)	52
5.3.7 Movement 詳細	54
5.3.8 多段変数最大繰返数管理 (Ansible-Legacy Role のみ)	56
5.3.9 代入値自動登録設定	58
5.3.10 作業対象ホスト	61
5.3.11 代入値管理	62
5.3.12 作業状態確認	67
5.3.13 作業管理	70
5.3.14 作業実行	71
6 構築コード記述方法	72
6.1 Playbook (Ansible-Legacy) の記述	72

6.2	対話ファイル (Ansible-Pioneer) の記述 .....	73
6.3	ロールパッケージ (Ansible-Legacy Role) の記述 .....	85
6.4	ITAdreadme (Ansible-Legacy Role のみ) の記述 .....	88
6.5	読替表 (Ansible-Legacy Role のみ) の記述 .....	90
6.6	BackYard コンテンツ .....	92
6.7	Ansible 利用ガイドライン ITA 追加ルール .....	94
7	運用操作 .....	95
7.1	メンテナンス .....	95
7.2	メンテナンス方法について .....	96
8	付録 .....	97
8.1	Ansible 実行時に使用される投入データと ITA メニューの紐づけ .....	97
8.1.1	Ansible-Legacy 投入データ .....	98
8.1.2	Ansible-Pioneer 投入データ .....	100
8.1.3	Ansible-LegacyRole 投入データ .....	102
8.2	Ansible 実行時に作成される結果データ .....	104
8.2.1	Legacy/LegacyRole 結果データに保存されるファイル一覧 .....	104
8.2.2	Pioneer 結果データに保存されるファイル一覧 .....	105

## はじめに

---

本書では、ITA の機能および操作方法について説明します。

# 1 Ansible driver 概要

本章では Ansible、AnsibleTower および Ansible driver について説明します。

## 1.1 Ansible について

Ansible とは、多数の構築管理対象に対して、アプリケーション/システムのデプロイ作業を容易にする PF 構築自動化ツールです。

Ansible は、Playbook という YAML 形式のテキストファイルに定型処理をタスクとして記述し、それを Ansible に実行させることにより、さまざまな処理を実現できます。  
タスクはモジュールと呼ばれる処理プログラムと紐付いており、さまざまな機器に対する制御を行うことができます。

Ansible の詳細情報については、Ansible の製品マニュアルを参照してください。  
Ansible のバージョンは 2.2.1.0 となります。最新の Ansible に対応した記法は使えないことがありますので、注意してください。

## 1.2 AnsibleTower について

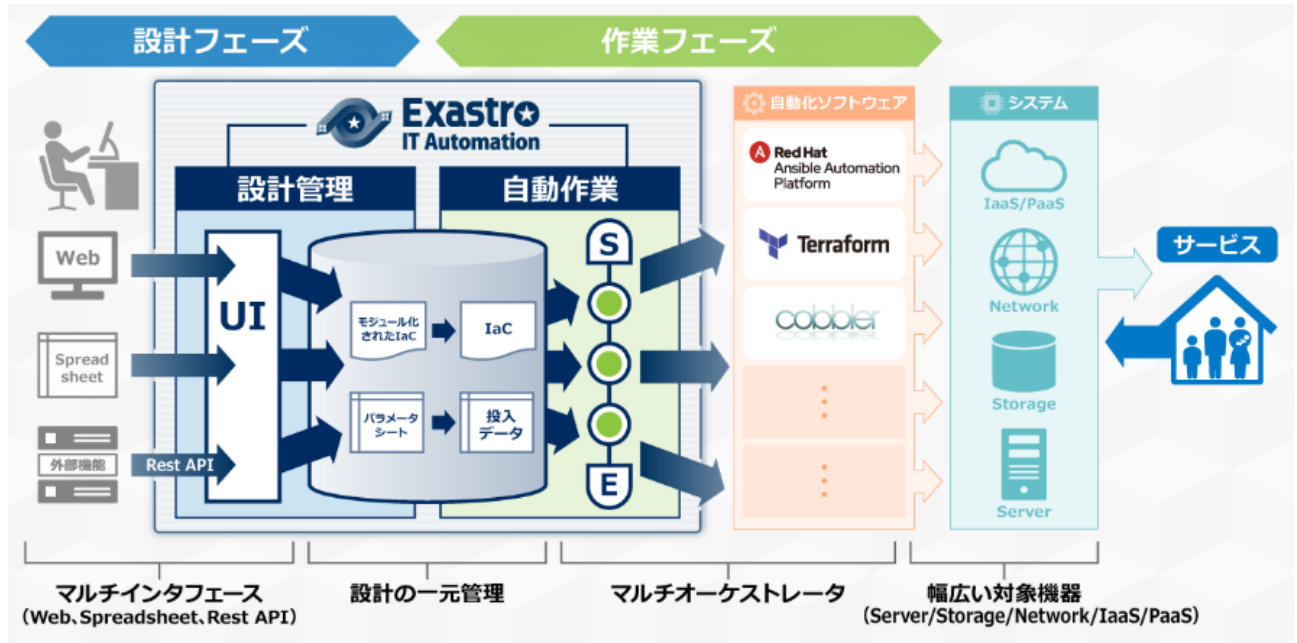
AnsibleTower とは、PF 構築自動化ツールである Ansible にアクセスコントロール、ジョブスケジューリング、タスクの可視化などの機能を拡張した管理プラットフォームです。

“プロジェクト”、“インベントリ”、“認証情報”の組合せで“ジョブテンプレート”を作成し Ansible を実行できます。  
複数の“ジョブテンプレート”を組み合わせで“ワークフロージョブテンプレート”を作成することによって、より多彩な作業フローを表現することができます。

AnsibleTower の詳細情報については、AnsibleTower の製品マニュアルを参照してください。  
本製品 ITA で対応可能な AnsibleTower のバージョンは、「システム構成／環境構築ガイド\_Ansible-driver 編」を参照してください。  
最新のバージョンに対応した記法は使えないことがありますので、ご注意ください。

## 1.3 Ansible driver について

Ansible driver は、ITA システムのオプションとして機能し、ITA システムで登録した構築対象のサーバ・ストレージ・ネットワーク各機器に対し、Ansible、AnsibleTower のどちらを経由するかを選択し、実際の運用設定を自動的に行います。



Ansible driver には用途に応じて以下 3 つのモードを用意しています。

- ① **Legacy モード**  
Ansible 標準の機能を用いて各種ホストへ設定を投入します。  
構築コードを単体 YAML ファイルとして登録し、作業パターンをその組み合わせで構成します。  
OS, NW の環境設定などの作業用に使われることを想定します。
- ② **Legacy Role モード**  
Legacy モードと同じく、Ansible 標準の機能を用いて各種ホストへ設定を投入します。  
構築コードをパッケージとして登録し、作業パターンを Role の組み合わせで構成します。  
製品部門などが提供する Role パッケージを用いて、製品のインストール、環境構築などを行う際に使われることを想定します。
- ③ **Pioneer モード**  
Ansible に独自モジュールを追加し、対話形式による設定投入を可能とします。  
サーバ、ストレージ、ネットワークを問わず、Telnet, SSH でログイン可能なあらゆる機器に対応しています。対象機器と直接やり取りが必要となるため、相応の IT スキルが必要となります。

また、Ansible driver は、Playbook 中の変数を画面から設定することができます。詳細は本書「[2Ansible driver での変数取り扱い](#)」をご参照ください。

## 2 Ansible driver での変数取り扱い

### 2.1 変数の種類

Ansible driver では、Playbook 中の変数の具体値を ITA の設定画面から設定することができます。

※設定方法の詳細は、本書「[5.3.11 代入値管理](#)」を参照してください。

Playbook 中の変数で、ITA の変数として扱える変数は以下の 6 種類があります。

表 2.1 変数の種類

種類	内容	Legacy	Pioneer	Legacy Role
通常変数	変数名に対して具体値を 1 つ定義できる変数です。 Playbook 内の変数は <code>{{△VAR_xxx△}}</code> で記述してください △:半角スペース xxx: 半角英数字とアンダースコア( _ ) e.g.) VAR_users: root	○	○	○
複数具体値変数	変数名に対して具体値を複数定義できる変数です。 Playbook 内の変数は <code>{{△VAR_xxx△}}</code> で記述してください。 △:半角スペース xxx: 半角英数字とアンダースコア( _ ) e.g.) VAR_users: - root - mysql	○	○	○
多段変数	階層化された変数です。 Playbook 内の変数は <code>{{△VAR_xxx△}}</code> で記述してください。 △:半角スペース xxx: 半角英数字とアンダースコア( _ ) e.g.) VAR_users: - user-name: alice authorized: password }   メンバー変数  メンバー変数名は、下記の 7 文字を除く ascii 文字 (0x20~0x7e)が使用出来ます。 " . [ ] ' ¥ : 尚、コーテーションで囲まないと変数名の先頭に使用出来ない文字がいくつかあります。詳しくは、Ansible ドキュメント <a href="#">Yaml syntax</a> を参照下さい。	×	×	○
グローバル変数	「グローバル変数」メニューから登録された変数です。	○	○	○



種類	内容	Legacy y	Pioneer	Legacy Role										
ITA 独自変数	ITA 独自で定義された変数です。 基本コンソールの機器一覧の下記項目を変数として扱えます。	○	○	○										
	<table><tr><th>項目名</th><th>変数名</th></tr><tr><td>ホスト名</td><td>__loginhostname__</td></tr><tr><td>プロトコル</td><td>__loginprotocol__</td></tr><tr><td>ログインユーザ ID</td><td>__loginuser__</td></tr><tr><td>ログインパスワード</td><td>__loginpassword__</td></tr></table>				項目名	変数名	ホスト名	__loginhostname__	プロトコル	__loginprotocol__	ログインユーザ ID	__loginuser__	ログインパスワード	__loginpassword__
	項目名				変数名									
	ホスト名				__loginhostname__									
	プロトコル				__loginprotocol__									
	ログインユーザ ID				__loginuser__									
	ログインパスワード				__loginpassword__									
	変数名の前後の「__」は半角アンダーバー2文字です。 機器一覧については、「利用手順マニュアル_基本コンソール」を参照してください。 作業実行時のディレクトリパスを下記の変数として扱えます。													
	<table><tr><th>項目名</th><th>変数名</th></tr><tr><td>作業ディレクトリパス</td><td>__workflowdir__</td></tr></table>				項目名	変数名	作業ディレクトリパス	__workflowdir__						
	項目名				変数名									
作業ディレクトリパス	__workflowdir__													
Playbook 内で作業ディレクトリパス配下にファイルを作成することで、「作業実行」の結果データでファイルをダウンロードすることができます。 Symphony 実行時の各 Movement で共有するディレクトリパスを下記の変数として扱えます。														
<table><tr><th>項目名</th><th>変数名</th></tr><tr><td>Symphony 作業ディレクトリパス</td><td>__symphony_workflowdir__</td></tr></table>	項目名	変数名	Symphony 作業ディレクトリパス	__symphony_workflowdir__										
項目名	変数名													
Symphony 作業ディレクトリパス	__symphony_workflowdir__													
Playbook 内で Symphony 作業ディレクトリパス配下にファイルを作成することで、各 Movement 間でファイルを共有することが出来ます。また、ansible driver の作業実行から実行した場合は __workflowdir__ と同じパスが設定されます。 Conductor 実行時の各 Movement で共有するディレクトリパスを下記の変数として扱えます。														
<table><tr><th>項目名</th><th>変数名</th></tr><tr><td>Conductor 作業ディレクトリパス</td><td>__conductor_workflowdir__</td></tr></table>	項目名	変数名	Conductor 作業ディレクトリパス	__conductor_workflowdir__										
項目名	変数名													
Conductor 作業ディレクトリパス	__conductor_workflowdir__													
Playbook 内で Conductor 作業ディレクトリパス配下にファイルを作成することで、各 Movement 間でファイルを共有することが出来ます。また、ansible driver の作業実行から実行した場合は __workflowdir__ と同じパスが設定されます。														
読替変数	Defaults 変数定義ファイルまたは ITA readme に定義されている「VAR_×××」形式以外の変数を ITA で扱う場合の変数「LCA_×××」です。 詳細は「6.7 読替表の記述」を参照して下さい。	×	×	○										

## 2.2 変数の抽出および具体値登録

各モードとも、ITA にアップロードした Playbook 等の資材から変数を抜き出します。

変数の抽出方法は以下のとおりです。

### (1) Ansible-Legacy

「プレイブック素材集 (本書: [5.3.3 プレイブック素材集 \(Ansible-Legacy のみ\)](#))」でアップロードした Playbook より、以下の書式の変数定義を抽出します。

```
{{△VAR_xxx△}}
```

※ △:半角スペース xxx: 半角英数字とアンダースコア( \_ )

Ansible-Legacy では具体値の登録の仕方で通常変数か複数具体値変数かを決定します。詳しくは「[5.3.11 代入値管理](#)」を参照してください。

### (2) Ansible-Pioneer

「対話ファイル素材 (本書: [5.3.6 対話ファイル素材集 \(Ansible-Pioneer のみ\)](#))」でアップロードした対話ファイルより、Ansible-Legacy と同様の変数定義の抽出を行います。

### (3) Ansible-Legacy Role

「ロールパッケージ管理 (本書: [5.3.4 ロールパッケージ管理 \(Ansible-Legacy Role のみ\)](#))」でアップロードしたロールパッケージ内の Playbook より、Ansible-Legacy と同様の抽出を行います。

詳しくは「ロールパッケージの記述(本書: [5.3.4 ロールパッケージ管理 \(Ansible-Legacy Role のみ\)](#))」を参照してください。

また、読替表を作成することで defaults 変数定義ファイルまたは ITA readme に定義されている「VAR\_xxx」以外の変数を ITA で扱うことが出来ます。詳しくは「[6.7 読替表の記述](#)」を参照して下さい。

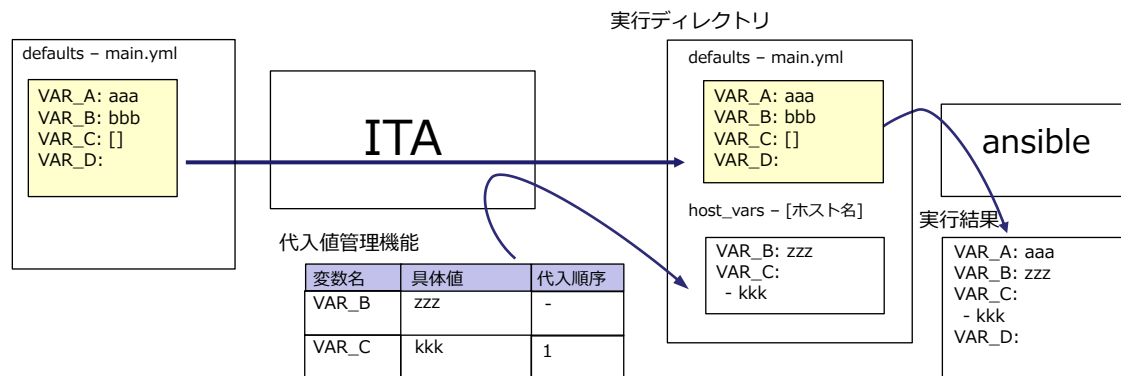
ITA にアップロードされた Playbook などの資材から変数を抽出して具体値を登録できます。

抽出した変数の具体値は、「[5.3.9 代入値自動登録設定](#)」や「[5.3.11 代入値管理](#)」で具体値を登録します。

登録された変数と具体値は、作業実行時にホスト変数ファイルに出力されます。

## 2.3 代入値登録による変数の扱い

Playbook で定義した変数の値は代入値登録機能により上書きすることができます。  
Playbook 中の変数と、代入値管理機能で登録した変数の値の関係を、以下の図に示します。



代入値管理機能で登録した変数の値は、各ホスト用に変数定義ファイル(host\_vars)に出力され、Ansible で元の Playbook と変数定義用ファイルを入力として各ホストに実行されます。

この結果、変数の値の優先順位は以下のようになります。

- ① 代入値管理機能で登録した値
  - ② Playbook 中の変数に指定した値
- 詳細は「5.3.11 [代入値管理](#)」を参照してください。

### 3 Ansible driver コンソールメニュー構成

本章では、ITA コンソールのメニュー構成について説明します  
なお、Web コンソールへのログイン方法、およびメニュー画面の構成要素/基本的な操作については、「[ファーストステップガイド](#)」を参照してください。

#### 3.1 メニュー/画面一覧

##### ① ITA 基本コンソールのメニュー

Ansible driver で利用する ITA 基本コンソールのメニュー一覧を以下に記述します。

表 3.1-1 基本コンソール メニュー/画面一覧

No	メニューグループ	メニュー・画面	説明
1	ITA 基本コンソール	機器一覧	作業対象システム一覧をメンテナンス(閲覧/登録/更新/廃止)します
2		紐付対象メニュー	代入値自動登録設定と連携する CMDB を管理します
3		投入オペレーション一覧	投入オペレーション一覧をメンテナンス(閲覧/登録/更新/廃止)できます

##### ② Ansible 共通コンソールのメニュー

Ansible 共通コンソールのメニュー一覧を以下に記述します。

表 3.1-2 共通コンソール メニュー/画面一覧

No	メニューグループ	メニュー・画面	説明
1	Ansible 共通コンソール	インターフェース情報	Ansible、AnsibleTower サーバのどちらを実行エンジンとし構築作業をするか選択をします。 ITA システム・Ansible driver サーバと実行エンジンのサーバが共有するディレクトリのパスおよび、実行エンジンのサーバへの接続インターフェース情報を管理します。
2		Ansible Tower ホスト一覧	AnsibleTower の RestAPI 実行に必要な情報、および構築資材を AnsibleTower にファイル転送するために必要な情報を管理します。
3		グローバル変数管理	Playbook や対話ファイルなどで共通利用する変数(以降、グローバル変数と称す)と具体値を管理します
4		テンプレート管理	Playbook 内の template モジュールなどで使用するテンプレートファイルと埋め込み変数を管理します
5		ファイル管理	Playbook 内の各モジュールで使用する素材ファイルと埋め込み変数を管理します

### ③ Ansible コンソールのメニュー

各 Ansible コンソールに対応するメニュー一覧を以下に記述します。

表 3.1-3 Ansible driver コンソール メニュー/画面一覧

No	メニューグループ			メニュー・画面	説明
	Ansible コンソール				
	Legacy	Legacy Role	Pioneer		
1			○	OS 種別	Pioneer より操作対象となる機器の OS 種別を管理します。
2	○	○	○	Movement 一覧	Symphony に登録する Movement の一覧を管理します
3	○			プレイブック素材集	Playbook ファイルを管理します
4		○		ロールパッケージ管理	ロールパッケージを管理します
5			○	対話種別リスト	同一目的の対話ファイルに対話種別としてまとめる種別を管理します
6			○	対話ファイル素材集	対話種別に紐づける OS 種別と ITA システム独自フォーマットの作業 手順ファイル(以降、対話ファイルと称す。)を管理します
7	○	○	○	Movement 詳細	Movement とプレイブック素材の関連付けを管理します
8		○		多段変数最大繰返数 管理	多段変数が繰返配列で構成されている場合の最大繰返配列数を管 理します
9	○	○	○	代入値自動登録設定	CMDB のメニューに登録されているオペレーションとホスト毎の項目 値を紐付ける Movement と変数を管理します
10	○	○	○	作業対象ホスト	Movement で使用するホストを管理します
11	○	○	○	代入値管理	変数の代入値を管理します
12	○	○	○	作業実行	作業実行する Movement とオペレーションを選択し実行を指示します
13	○	○	○	作業状態確認	作業実行状態を表示します
14	○	○	○	作業管理	作業実行履歴を管理します

※ Ansible RestAPI は、設定等ユーザー操作を行わないため、ここでの説明は省略します。

## 4 Ansible driver 利用手順

各 Ansible コンソールの利用手順について説明します

### 4.1 作業フロー

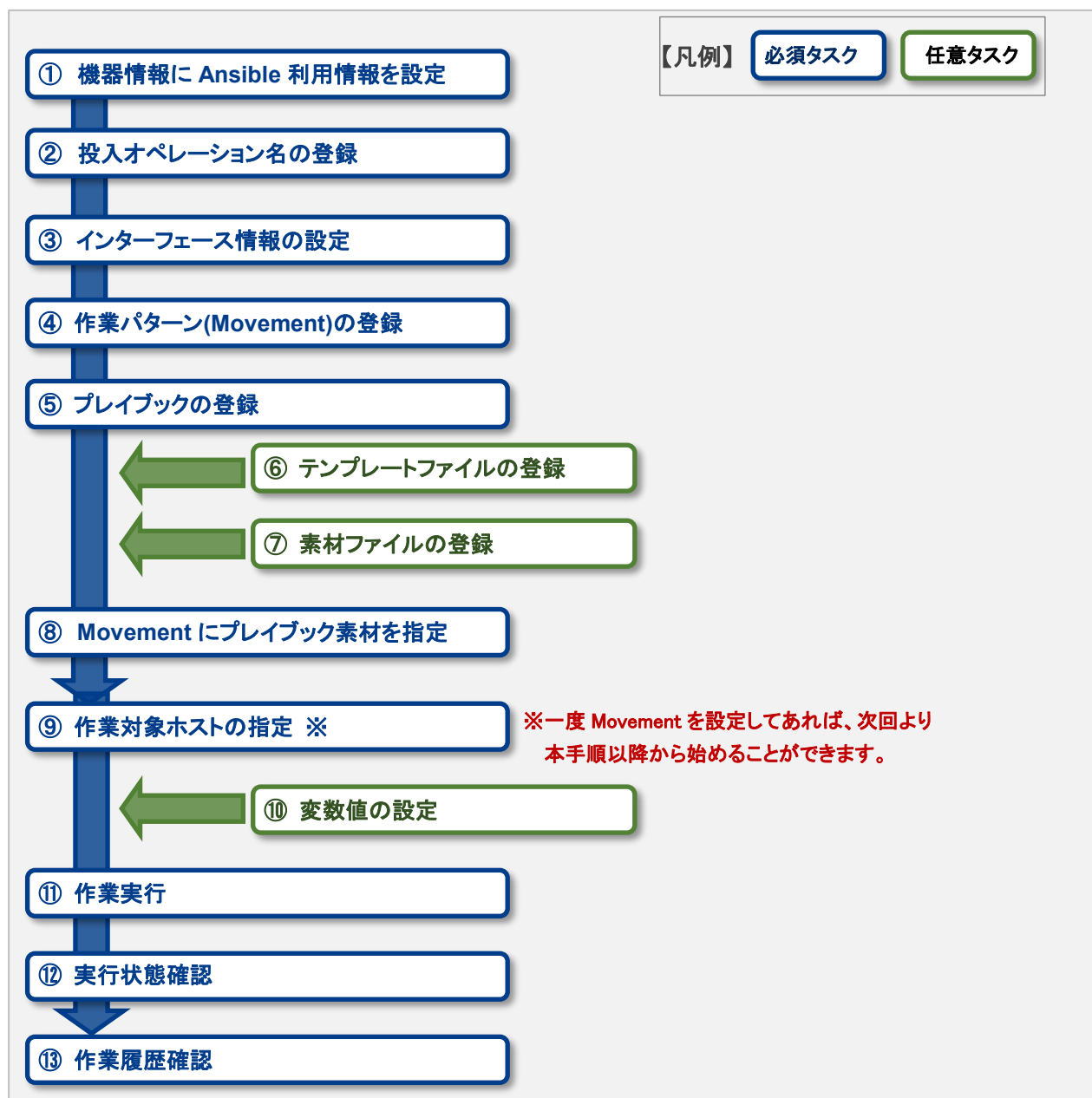
各 Ansible コンソールにおける標準的な作業フローは以下のとおりです。

各作業の詳細は次項に記載しています。

ITA 基本コンソールの利用方法は、「利用手順マニュアル\_基本コンソール」を参照してください。

#### 4.1.1 Ansible-Legacy 作業フロー

以下は、Ansible-Legacy で作業を実行するまでの流れです。



- **作業フロー詳細と参照先**

- ① **機器情報に Ansible 利用情報を設定**

- ITA 基本コンソールの機器一覧の画面から、各機器に対して Ansible 利用情報を設定します。  
詳細は [5.1.1 機器一覧](#) を参照してください。

- ② **投入オペレーション名の登録**

- ITA 基本コンソールの投入オペレーション一覧の画面から、作業用の投入オペレーション名を登録します。  
詳細は [5.1.2 投入オペレーション一覧](#) を参照してください。

- ③ **インターフェース情報の登録**

- Ansible 共通コンソールのインターフェース情報の画面から、Ansible、AnsibleTower サーバのどちらを実行エンジンにするかを選択し、実行エンジンのサーバへの接続情報の登録します。  
詳細は [5.2.1 インターフェース情報](#) を参照してください。

- ④ **作業パターン(Movement)の登録**

- Ansible-Legacy コンソールの Movement 一覧の画面から、作業用の Movement を登録します。  
詳細は [5.3.2 Movement 一覧](#) を参照してください。

- ⑤ **Playbook の登録**

- Ansible-Legacy コンソールのプレイブック素材集の画面から、作業で使用する Playbook を登録します。  
詳細は [5.3.3 プレイブック素材集 \(Ansible-Legacy のみ\)](#) を参照してください。

- ⑥ **テンプレートファイルの登録(必要に応じて実施)**

- Ansible 共通コンソールのテンプレート管理の画面から、Playbook 内の template モジュールなどで使用している template ファイル(src)と template 埋め込み変数の登録／更新／廃止を行います。  
詳細は [5.2.4 テンプレート管理](#) を参照してください。

- ⑦ **素材ファイルの登録 (必要に応じて実施)**

- Ansible 共通コンソールのファイル管理の画面から、作業対象サーバに配置するファイルを登録します。  
詳細は [5.2.5 ファイル管理](#) を参照してください。

- ⑧ **Movement にプレイブック素材を指定**

- Ansible-Legacy コンソールの Movement 詳細の画面から、登録した Movement にプレイブック素材を指定します。  
詳細は [5.3.7 Movement 詳細](#) を参照してください。

- ⑨ **作業対象ホストの指定**

- Ansible-Legacy コンソールの作業対象ホストの画面から、作業対象ホストを指定します。  
詳細は [5.3.10 作業対象ホスト](#) を参照してください。

- ⑩ **変数値の設定(必要に応じて実施)**

- Ansible-Legacy コンソールの代入値管理の画面から、Movement に登録した Playbook 内で定義した変数の値を設定します。変数を利用していない場合、設定は不要です。  
詳細は [5.3.11 代入値管理](#) を参照してください。

⑪ **作業実行**

Ansible-Legacy コンソールの作業実行の画面から、実行日時、投入オペレーションを選択して設定して処理の実行を指示します。

詳細は [5.3.14 作業実行](#) を参照してください。

⑫ **作業状態確認**

Ansible-Legacy コンソールの作業状態確認の画面では、実行した作業の状態がリアルタイムで表示されます。また、作業の緊急停止や、実行ログ、エラーログを監視することができます。

詳細は [5.3.12 作業状態確認](#) を参照してください。

⑬ **作業履歴確認**

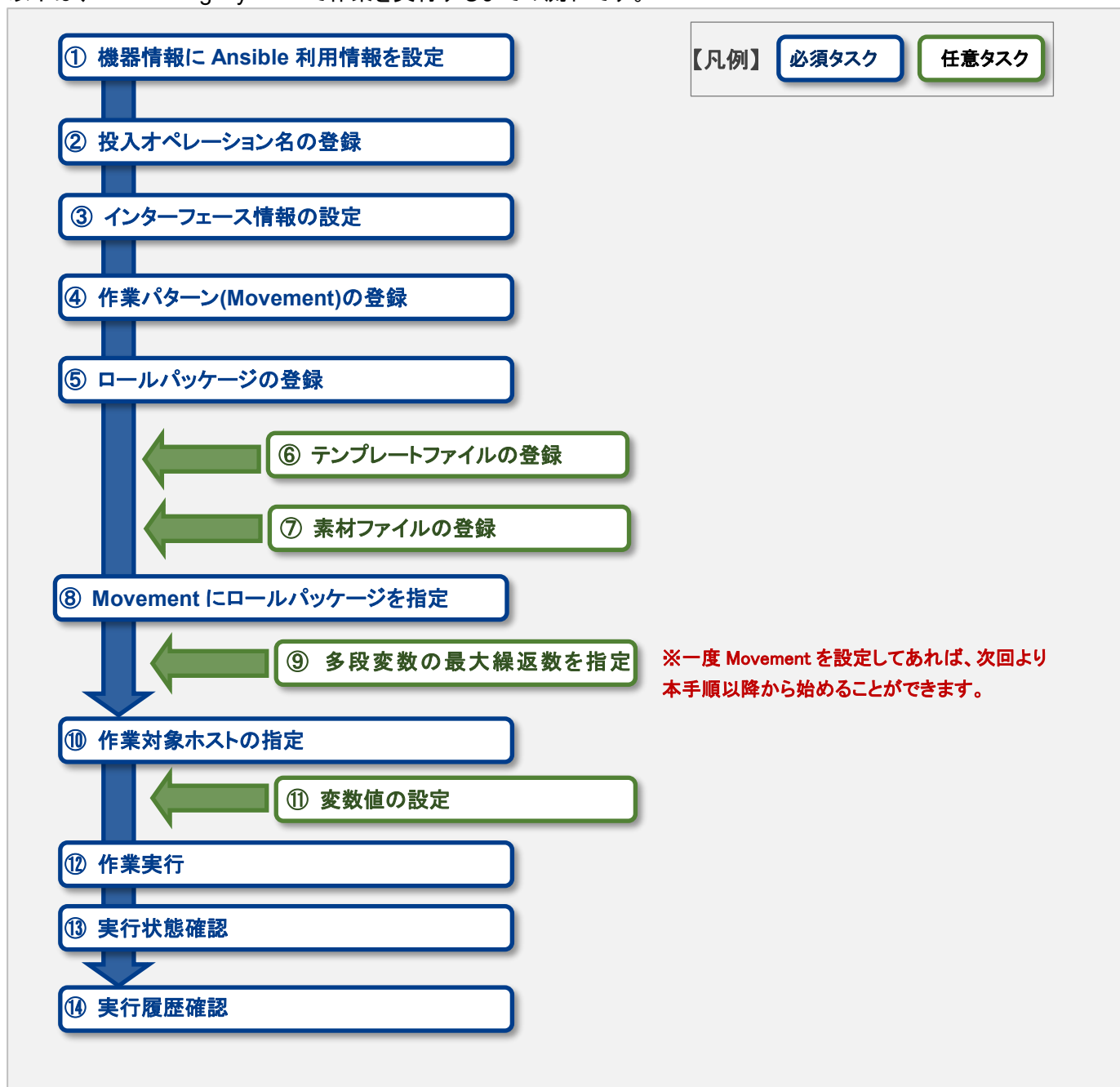
Ansible-Legacy コンソールの作業管理の画面では、実行した作業の一覧が表示され履歴が確認できます。

詳細は [5.3.13 作業管理](#) を参照してください。



#### 4.1.2 Ansible-Legacy Role 作業フロー

以下は、Ansible-Legacy Role で作業を実行するまでの流れです。



## ● 作業フロー詳細と参照先

### ① 機器情報に Ansible 利用情報を設定

ITA 基本コンソールの機器一覧の画面から、各機器に対して Ansible 利用情報を設定します。  
詳細は [5.1.1 機器一覧](#) を参照してください。

### ② 投入オペレーション名の登録

ITA 基本コンソールの投入オペレーション一覧の画面から、作業用の投入オペレーション名を登録します。  
詳細は [5.1.2 投入オペレーション一覧](#) を参照してください。

### ③ インターフェース情報の登録

Ansible 共通コンソールのインターフェース情報の画面から、Ansible、AnsibleTower サーバのどちらを実行エンジンにするかを選択し、実行エンジンのサーバへの接続情報を登録します。  
詳細は [5.2.1 インターフェース情報](#) を参照してください。

### ④ 作業パターン(Movement)の登録

Ansible-Legacy Role コンソールの Movement 一覧の画面から、作業用の Movement を登録します。  
詳細は [5.3.2 Movement 一覧](#) を参照してください。

### ⑤ ロールパッケージの登録

Ansible-Legacy Role コンソールのロールパッケージ管理の画面から、作業で使用するロールパッケージを登録します。  
詳細は [5.3.4 ロールパッケージ管理 \(Ansible-Legacy Role のみ\)](#) を参照してください。

### ⑥ テンプレートファイルの登録(必要に応じて実施)

Ansible 共通コンソールのテンプレート管理の画面から、ロールパッケージの template モジュールなどで使用している template ファイル(src)と template 埋め込み変数の登録／更新／廃止を行います。  
詳細は [5.2.4 テンプレート管理](#) を参照してください。

### ⑦ 素材ファイルの登録 (必要に応じて実施)

Ansible 共通コンソールのファイル管理の画面から、作業対象サーバに配置するファイルを登録します。  
詳細は [5.2.5 ファイル管理](#) を参照してください。

### ⑧ Movement にロールパッケージを指定

Ansible-Legacy Role コンソールの Movement 詳細の画面から、登録した Movement にプレイブック素材を指定します。  
詳細は [5.3.7 Movement 詳細](#) を参照してください。

### ⑨ 多段変数の最大繰返数を指定

Ansible-Legacy Role コンソールの多段変数最大繰返管理の画面から、多段変数で配列定義されているメンバー変数の配列の最大繰返数を指定します。  
詳細は [5.3.8 多段変数最大繰返数管理 \(Ansible-Legacy Role のみ\)](#) を参照してください。

### ⑩ 作業対象ホストの指定

Ansible-Legacy Role コンソールの作業対象ホストの画面から、作業対象ホストを指定します。  
詳細は [5.3.10 作業対象ホスト](#) を参照してください。

⑪ **変数値の設定**

Ansible-Legacy Role コンソールの代入値管理の画面から、Movement に登録した Playbook 内で定義した変数の値を設定します。変数を利用していない場合、設定は不要です。

詳細は 5.3.11 [代入値管理](#) を参照してください。

⑫ **作業実行**

Ansible-Legacy Role コンソールの作業実行の画面から、実行日時、投入オペレーションを選択して設定して処理の実行を指示します。

詳細は 5.3.14 [作業実行](#) を参照してください。

⑬ **作業状態確認**

Ansible-Legacy Role コンソールの作業状態確認の画面から、実行した作業の状態がリアルタイムで表示されます。また、作業の緊急停止や、実行ログ、エラーログを監視することができます。

詳細は 5.3.12 [作業状態確認](#) を参照してください。

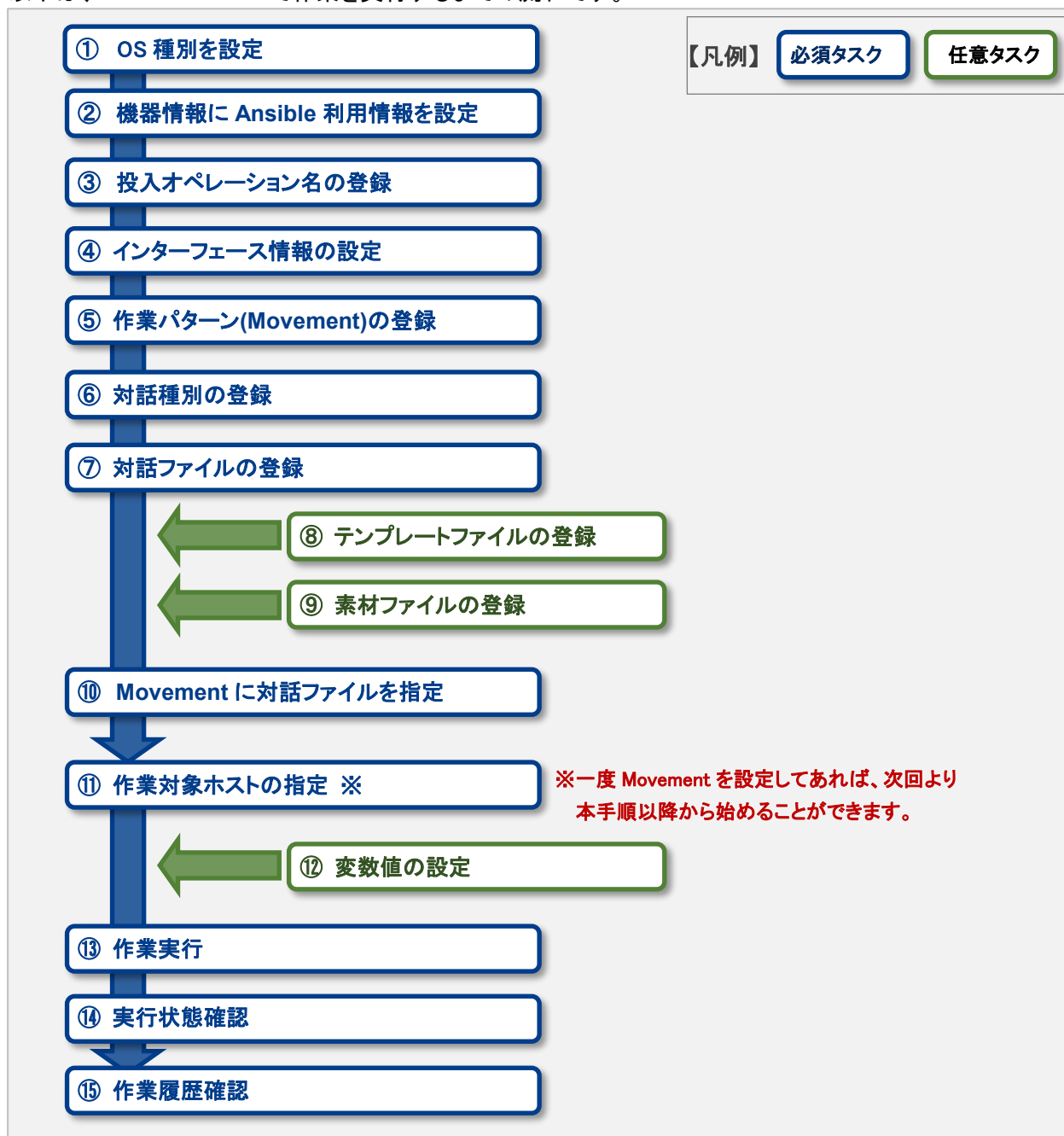
⑭ **作業履歴確認**

Ansible-Legacy Role コンソールの作業管理の画面から、実行した作業の一覧が表示され履歴が確認できます。

詳細は 5.3.13 [作業管理](#) を参照してください。

### 4.1.3 Ansible-Pioneer 作業フロー

以下は、Ansible-Pioneer で作業を実行するまでの流れです。



#### ● 作業フロー詳細と参照先

##### ① OS 種別の登録

Pioneer より操作対象となる機器の OS 種別を設定します。

##### ② 機器情報に Ansible 利用情報を設定

ITA 基本コンソールの機器一覧の画面から、各機器に対して Ansible 利用情報を設定します。  
詳細は [5.1.1 機器一覧](#) を参照してください。

③ 投入オペレーション名の登録

ITA 基本コンソールの投入オペレーション一覧の画面から、作業用の投入オペレーション名を登録します。

詳細は [5.1.2 投入オペレーション一覧](#) を参照してください。

④ インターフェース情報の登録

Ansible 共通コンソールのインターフェース情報の画面から、Ansible、AnsibleTower サーバのどちらを実行エンジンにするかを選択し、実行エンジンのサーバへの接続情報の登録します。

詳細は [5.2.1 インタフェース情報](#) を参照してください。

⑤ 作業パターン(Movement)の登録

Ansible-Pioneer コンソールの Movement 一覧の画面から、作業用の Movement を登録します。

詳細は [5.3.2 Movement 一覧](#) を参照してください。

⑥ 対話種別の登録

Ansible-Pioneer コンソールの対話種別リストの画面から、対話種別を登録します。

OS 種別ごとの差異を対話ファイルごとに定義し、同一目的の対話ファイルを対話種別として纏めて機器差分を吸収(抽象化)します。

詳細は、対話種別リスト(Ansible-Pioneer のみ) を参照してください。

⑦ 対話ファイルの登録

Ansible-Pioneer コンソールの対話ファイル素材集の画面から、対話種別と OS 種別の組み合わせに対して対話ファイルを登録します。

詳細は [5.3.6 対話ファイル素材集\(Ansible-Pioneer のみ\)](#) を参照してください。

⑧ テンプレートファイルの登録(必要に応じて実施)

Ansible 共通コンソールのテンプレート管理の画面から、対話ファイルで使用している template ファイルと template 埋め込み変数の登録／更新／廃止を行います。

詳細は [5.2.4 テンプレート管理](#) を参照してください。

⑨ 素材ファイルの登録(必要に応じて実施)

Ansible 共通コンソールのファイル管理の画面から、作業対象サーバに配置するファイルを登録します。

詳細は [5.2.5 ファイル管理](#) を参照してください。

⑩ Movement に対話ファイルを指定

Ansible-Pioneer コンソールの Movement 詳細の画面から、登録した Movement に対話ファイルに対応した対話種別リストを指定します。

詳細は [5.3.7 Movement 詳細](#) を参照してください。

⑪ 作業対象ホストの指定

Ansible-Pioneer コンソールの作業対象ホストの画面から、作業対象ホストを指定します。

詳細は [5.3.10 作業対象ホスト](#) を参照してください。

⑫ 変数値の設定

Ansible-Pioneer コンソールの代入値管理の画面から、Movement に登録した Playbook 内で定義した変数の値を設定します。変数を利用していない場合、設定は不要です。

詳細は [5.3.11 代入値管理](#) を参照してください。

### ⑬ 作業実行

Ansible-Pioneer コンソールの作業実行の画面から、実行日時、投入オペレーションを選択して設定して処理の実行を指示します。

詳細は [5.3.14 作業実行](#) を参照してください。

### ⑭ 作業状態確認

Ansible-Pioneer コンソールの作業状態確認の画面では、実行した作業の状態がリアルタイムで表示されます。また、作業の緊急停止や、実行ログ、エラーログを監視することができます。

詳細は [5.3.12 作業状態確認](#) を参照してください。

### ⑮ 作業履歴確認

Ansible-Pioneer コンソールの作業管理の画面では、実行した作業の一覧が表示され履歴が確認できます。

詳細は [5.3.14 作業管理](#) を参照してください。

## ■登録画面項目一覧凡例

次項に記載の登録画面項目一覧表の内容について説明します。

① 項目	② 説明	③ 入力 必須	④ 入力形式	⑤ 制約事項

#### ①項目

- ・サブメニュー内の項目名です

#### ②説明

- ・項目に対する説明です

#### ③入力必須

- ・○: 項目に対する内容の入力が必須の項目
- ・-: 項目に対する内容の入力が任意の項目

#### ④入力形式

- ・手動入力: 手動での入力が必要な項目
- ・自動入力: 自動で内容が入力される項目
- ・チェックボックス: チェックボックス形式の項目
- ・ボタン: ラジオボタン形式の項目
- ・リスト選択: リストボックス形式の項目

#### ⑤制約事項

- ・項目に対する制約事項(文字数制限など)です

## 5 Ansible driver 機能・操作方法説明

本章では、Ansible driver で利用する各コンソールの機能について説明します。

### 5.1 基本コンソール

本節では、ITA 基本コンソールでの操作について記載します。

本作業は ITA 基本コンソールマニュアルを参照して、ITA 基本コンソール画面内で作業を実施してください。

#### 5.1.1 機器一覧

(1) [機器一覧]では、作業対象ホストの情報を登録／更新／廃止を行います。

本書では、Ansible driver の動作に必要な機器一覧の項目(赤枠)について説明します。

他の項目についての説明は、「利用手順マニュアル\_基本コンソール」を参照してください。

The screenshot shows the 'Exastro IT Automation 基本コンソール' interface. On the left, the 'Menu' sidebar has '機器一覧' (Machine List) highlighted with a red box. The main content area is titled '説明' (Description) and '表示フィルタ' (Display Filter). It contains a table with the following columns: '廃止' (Cancel), '管理システム番号' (Management System Number), 'HW機器種別' (HW Device Type), 'ホスト名' (Host Name), 'IPアドレス' (IP Address), 'MACアドレス' (MAC Address), 'ネットワークデバイス名' (Network Device Name), and 'ログインユーザ' (Login User). Below the table are buttons for 'フィルタ' (Filter) and 'フィルタクリア' (Clear Filter), and a checkbox for 'オートフィルタ' (Auto Filter). The bottom of the page has a '管理者に連絡' (Contact Administrator) button.

図 5.1-1 サブメニュー画面(機器一覧)

(2) 「登録」-「登録開始」ボタンより、機器情報の登録を行います。

図 5.1-2 登録画面(機器一覧 - 共通項目)

図 5.1-3 登録画面(機器一覧 - Ansible 利用情報)

(3) 登録画面の共通項目一覧は以下のとおりです。  
Web 画面のカラム名の後ろに赤のアスタリスク(\*)が付いているカラムが必須入力になりますが、Ansible driver を利用する場合には、Ansible 利用情報を入力して下さい。  
未入力で作業実行した場合、想定外エラーとなる場合があります。

表 5.1-1 登録画面項目一覧(機器一覧)

項目		説明	入力必須	入力形式	制約事項
管理システム項番		登録情報を識別する一意のIDが自動入力されます	-	自動入力	-
ホスト名		ホスト名を記入します	○	手動入力	最大長 128 バイト
IP アドレス		IP アドレス(XXX.XXX.XXX.XXX 形式)を記入します	○	手動入力	最大長 15 バイト
EtherW ake OnLan	MAC アドレス	MAC アドレスを記入します	-	手動入力	最大長 17 バイト
	ネットワークデ バイ ス名	ネットワークデバイス名を記入します	-	手動入力	最大長 256 バイト
ログインユーザ ID		ログインユーザ ID を記入します	○	手動入力	最大長 30 バイト
ログイン パス ワ ード	管理	ITA でパスワードを管理する場合「●」を選択します	○	リスト選択	-
	ログイン パスワード	パスワードを指定します	○	手動入力	最大長 30 バイト
ssh 認証鍵ファイル		ssh 認証鍵ファイルを指定して鍵認証する場合のファイルを入力します。 認証方式が鍵方式の場合で ssh 認証鍵ファイルを指定する場合に必須入力となります。	-	ファイル 選択	最大サイズ 10K バ イト



項目			説明	入力必須	入力形式	制約事項
Ansible 利用情報	Legacy/Role 利用情報	認証方式	Ansible からホストへ接続する際の認証方式を選択します。 ・パスワード方式の場合、ログインパスワードの管理は、「●」である必要があります。 ・鍵方式の場合、また、公開鍵を交換しておく必要があります。 また、ログインユーザでの sudo 権限を NOPASSWD 付で/etc/sudoers に設定する必要があります。 Exp) Demo_user ALL=(ALL) NOPASSWD:ALL	○	リスト選択	説明欄記載のとおり
		WinRM 接続情報	ポート番号	－	手動入力	説明欄記載のとおり
			サーバ証明書	－	ファイル選択	最大サイズ 10K バイト
	Pioneer 利用情報	プロトコル	対象機器にログインする際のプロトコル(ssh/telnet)を選択します。	○	リスト選択	－
		OS 種別	対象機器の OS を選択します。 OS 種別マスタで登録されている OS 種別がリスト表示されます。	○	リスト選択	－
		接続オプション	(ssh 接続の場合) /etc/ansible.cfg/ssh_args に設定している ssh オプション以外のオプションを設定したい場合、設定したいオプションを入力します。 (telnet 接続の場合) telnet 接続時のオプションを設定したい場合、設定したいオプションを入力します。	－	手動入力	最大長 512 バイト
	Tower 利用情報	インベントリファイル追加オプション	ITA が設定していないインベントリファイルのオプションパラメータを yaml 形式で入力します。 Exp) ansible_connection: network_cli ansible_network_os: nxos	－	手動入力	最大長 512 バイト
		インスタンスグループ名※1	AnsibleTower がクラスター構成の場合、AnsibleTower のどのインスタンスグループに所属しているかを選択します。(※1) ここでの設定が Tower のインベントリへの指定となります。(※2) 未選択の場合は AnsibleTower のデフォルト("Tower") になります。 AnsibleTower がクラスター構成でない場合は、未選択で構いません。	－	リスト選択	－

項目		説明	入力必須	入力形式	制約事項
	接続タイプ	<p>AnsibleTower 認証情報の接続タイプを設定します。通常は machine を選択します。ansible_connection を local に設定する必要がある Network OS の場合に Network を選択します。</p> <p>Network を選択した場合、インベントリファイル追加オプションに Platform Options(ansible_connection 以外)を設定する必要があります。</p> <p>Exp)</p> <p>インベントリファイル追加オプションの設定例</p> <p>Network OS が ios の場合の設定値</p> <p>ansible_network_os: ios</p> <p>ansible_become: yes</p> <p>ansible_become_method: enable</p> <p>AnsibleTower の認証情報の接続タイプについては Tower ドキュメント <a href="#">認証情報タイプ</a> を参照して下さい。</p> <p>Network OS と ansible_connection の関連や Platform Options については、Ansible ドキュメント <a href="#">Platform Options</a> を参照ください。</p>	○	リスト選択	
備考		自由記述欄です。	-	手動入力	最大長 4000 バイト

※1 基本コンソールの入力項目ですが、Ansible driver のバックヤード機能”Tower サーバデータ連携”により取得したデータから選択します。

※2 インスタンスグループは、Tower のインベントリオブジェクトで指定しています。

## 5.1.2 投入オペレーション一覧

- (1) [投入オペレーション一覧]画面では、オーケストレータで実行する作業対象ホストに対するオペレーションを管理します。作業は ITA 基本コンソール内メニューより選択します。

The screenshot shows the 'Exastro IT Automation 基本コンソール' interface. The left sidebar menu has '投入オペレーション一覧' highlighted with a red box. The main content area features a table with columns: 停止 (Stop), No., オペレーションID (Operation ID), オペレーション名 (Operation Name), 実施予定日時 (Scheduled Time), and 最終実行日時 (Final Execution Time). Below the table are 'フィルタ' (Filter) and 'フィルタクリア' (Clear Filter) buttons. At the bottom, there is a table with four rows of action links: '一覧/更新' (List/Update), '登録' (Register), '全件ダウンロードとファイルアップロード編集' (Download all and edit file upload), and '変更履歴' (Change History).

図 5.1-4 サブメニュー画面(投入オペレーション一覧)

登録方法の詳細は、関連マニュアルの「利用手順マニュアル\_基本コンソール」をご参照下さい。

## 5.2 Ansible 共通コンソール

本節では、Ansible 共通コンソールでの操作について記載します。

### 5.2.1 インターフェース情報

- (2) [インターフェース情報]では、Ansible、AnsibleTower サーバのどちらを実行エンジンにするかを選択し、ITA システム・Ansible driver サーバと実行エンジンのサーバが共有するディレクトリのパスおよび実行エンジンのサーバへの接続インターフェース情報を登録／更新／廃止を行います。

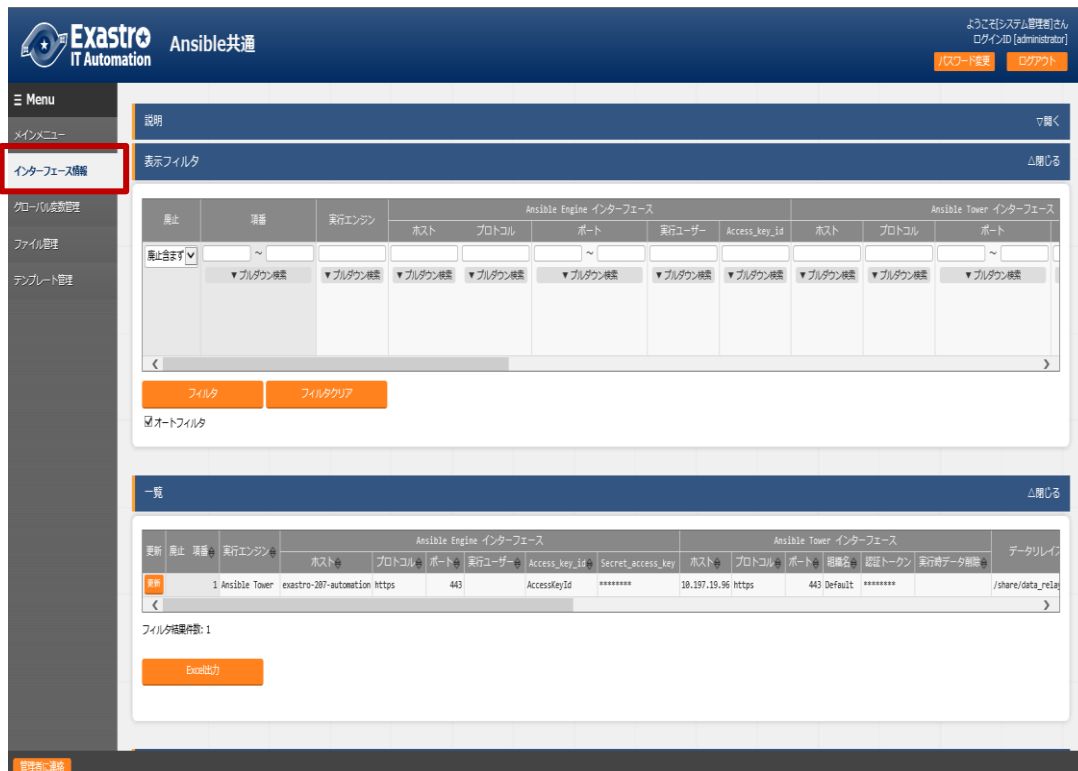


図 5.2-1 サブメニュー画面（インタフェース情報）

- (3) 「一覧」-「更新」ボタンより、インターフェース情報の登録を行います。

項目	実行エンジン	Ansible Engine インターフェース						Ansible Tower インターフェース			
		ホスト	プロトコル	ポート	実行ユーザー	Access_key_id	Secret_access_key	ホスト	プロトコル	ポート	組織名
1	Ansible Tower	exastro-207-autoi	https	443		AccessKeyId		10.197.19.96	https	443	Default

図 5.2-2 登録画面（インタフェース情報）

- (4) インタフェース情報画面の項目一覧は以下のとおりです。  
インタフェース情報が未登録または、複数レコード登録されている状態で作業実行した場合、**作業実行は想定外エラーとなります。**

表 5.2-1 登録画面項目一覧(インタフェース情報)

項目		説明	入力 必須	入力形式	制約事項
実行エンジン		実行するエンジンが Ansible か AnsibleTower を選択します。 AnsibleTower を選択した場合も ansible-vault コマンドを実行する Ansible Engine インタフェースが必要です。	○	リスト選択	
Ansible Engine インタフェース	ホスト	Ansible サーバのホスト名(または IP アドレス)を入力します。 HTTPS 通信の場合はホスト名が推奨です。	○	手動入力	最大長 128 バイト
	プロトコル	Ansible、AnsibleTower サーバとのプロトコルを http / https のどちらかを入力します。	○	手動入力	-
	ポート	Ansible、AnsibleTower サーバの接続ポート(80/443)を入力します。通常は HTTPS(443)です。	○	手動入力	-
	実行ユーザー	sudo コマンドで ansible-playbook/ansible-vault コマンドを実行するとユーザーを入力します。	○	手動入力	最大長 64 バイト
	ACCESS_KEY_ID	Ansible サーバ接続時の認証に使用するアクセスキーを入力します。	○	手動入力	最大長 64 バイト
	SECRET_ACCESS_KEY	Ansible サーバ接続時の認証に使用するシークレットアクセスキーを入力します。	○	手動入力	最大長 64 バイト
Ansible Tower インタフェース	代表ホスト	AnsibleTower ホスト一覧に登録されているホストの一覧より、ITA と通信する AnsibleTower を選択します。	-	リスト選択	実行エンジンが AnsibleTower の場合に入力必須
	プロトコル	Ansible、AnsibleTower サーバとのプロトコルを http / https のどちらかを入力します。	-	手動入力	実行エンジンが AnsibleTower の場合に入力必須
	ポート	Ansible、AnsibleTower サーバの接続ポート(80/443)を入力します。通常は HTTPS(443)です。	-	手動入力	実行エンジンが AnsibleTower の場合に入力必須
	組織名	Ansible Tower サーバに登録されている組織名を選択します。	-	リスト選択	実行エンジンが AnsibleTower の場合に入力必須
	認証トークン	ITA から Ansible Tower サーバに接続するユーザーの認証トークンを入力します。	-	手動入力	最大長 128 バイト 実行エンジンが AnsibleTower の場合に入力必須
	実行時データ削除	作業実行時に AnsibleTower 内で自動的に生成される実行時データを作業完了後に削除するかを選択します。 削除したい場合、リストより「削除する」を選択する	-	リスト選択	実行エンジンが AnsibleTower の場合に入力必須
データリストレージパス(ITA)※1		ITA システム・Ansible driver サーバから見たディレクトリを入力します。	○	手動入力	最大長 128 バイト
データリストレージパス (Ansible/Ansible)		Ansible RestAPI、AnsibleTower サーバから見たディレクトリを入力します。	○	手動入力	最大長 128 バイト

項目	説明	入力 必須	入力形式	制約事項
Tower)				
Symphony インスタンスデータリレイストレージパス (Ansible/Ansible Tower)	Symphony 実行時、各 Movement で共有するディレクトリを入力します。 ITA システムから見たパスは Symphony インターフェース情報より設定します。Symphony インターフェース情報については「利用手順マニュアル Symphony」を参照して下さい。	○	手動入力	最大長 128 バイト
Conductor インスタンスデータリレイストレージパス (Ansible/Ansible Tower)	Conductor 実行時、各 Movement で共有するディレクトリを入力します。 ITA システムから見たパスは Conductor インターフェース情報より設定します。Conductor インターフェース情報については「利用手順マニュアル Conductor」を参照して下さい。	○	手動入力	最大長 128 バイト
オプションパラメータ	<p>Movement 共通の ansible-playbook コマンドのオプションパラメータを入力します。</p> <p>Movement 固有のオプションパラメータは Movement 一覧で入力します。</p> <p>実行エンジンが Ansible の場合:</p> <p>ansible-playbook コマンドのオプションパラメータを入力します。</p> <p>-i オプションは ITA が設定します。</p> <p>実行エンジンが Ansible Tower の場合:</p> <p>以下のオプションパラメータが設定できます。</p> <p>-verbosity</p> <p>-f FORKS,--forks=FORKS</p> <p>-l SUBSET,--limit=SUBSET</p> <p>-e EXTRA_VARS,--extra-vars=EXTRA_VARS</p> <p>EXTRA_VARS: 変数名=具体値 変数名=具体値 .....</p> <p>-t TAGS,--tags=TAGS</p> <p>-b,--become</p> <p>-D,--diff</p> <p>--skip-tags=SKIP_TAGS</p> <p>--start-at-task=START_AT_TASK</p> <p>以下は Ansible Tower 独自のオプションパラメータです。</p> <p>-ufc,--use_fact_cache      ファクトのキャッシュの使用</p> <p>-as,--allow_simultaneous    同時実行ジョブの有効化</p> <p>-jsc,--job_slice_count=      ジョブスライス数</p> <p>Ansible Tower 独自のオプションパラメータについては、Ansible Tower ユーザーガイドのジョブテンプレートの説明を参照して下さい。</p>	-	手動入力	最大長 512 バイト Exastro1.2 では、ジョブスライス数「-jsc,--job_slice_count=」は対応していません。

項目	説明	入力 必須	入力形式	制約事項
並列実行数	同時に実行できる Movement(Legacy/Pioneer/ Legacy-Role)の最大数を入力します。	○	手動入力	
状態監視周期(単位ミ リ秒)	「5.3.12 作業状態確認」で表示されるログのリフレッ シュ間隔を入力します。通常は 3000 ミリ秒程度が推 奨値です。	○	手動入力	最小値 1000 ミリ秒
進行状態表示行数	「5.3.12 作業状態確認」での進行ログ・エラーログの 最大表示行数を入力します。通常は 1000 行程度が 推奨値です。	○	手動入力	-
NULL 連携	代入値自動登録設定でパラメータシート of 具体値が NULL(空白)の場合に、代入値管理への登録を NULL(空白)の値で行うか設定します。 代入値自動登録設定メニューの「NULL 連携」が空白 の場合この値が適用されます。 ・「有効」の場合、パラメータシート of 値がどのよう な値でも代入値管理への登録が行われます。 ・「無効」の場合、パラメータシートに値が入っている 場合のみ代入値管理への登録が行われます。	○	リスト選択	-
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

※1 データリレイストレージパスは、それぞれ異なるサーバで運用される場合、ディレクトリパス名が異なる可能性があるため、  
別々に管理します。詳細は「システム構成／環境構築ガイド\_Ansible-driver 編」を参照してください。

## 5.2.2 Ansible Tower ホスト一覧

[Ansible Tower ホスト一覧]では、AnsibleTower の RestAPI 実行に必要な情報、および構築資材を AnsibleTower にファイル転送するために必要な情報を登録／更新／廃止を行います。AnsibleTower をクラスター構成で構築されている場合、クラスター内の全てのホスト情報の登録が必要です。クラスター構成でない場合は、通信対象の AnsibleTower(1 台)のホスト情報を登録して下さい。

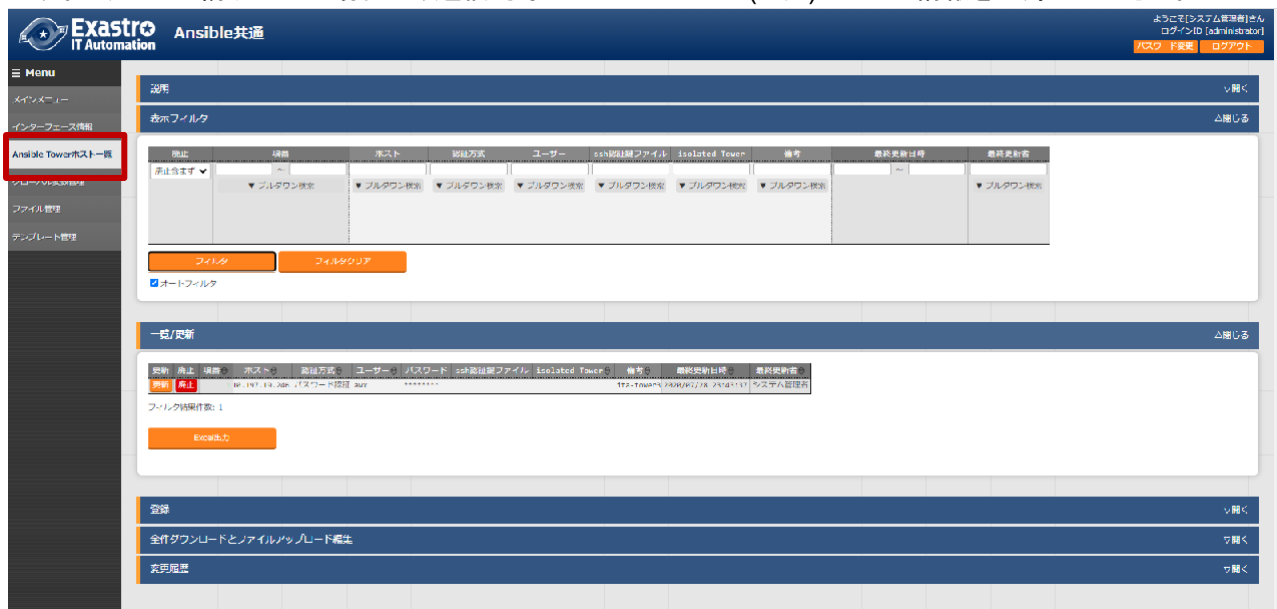


図 5.2-3 サブメニュー画面(Ansible Tower ホスト一覧)

(1) 「一覧」-「更新」ボタンより、AnsibleTower のホスト情報の登録を行います。

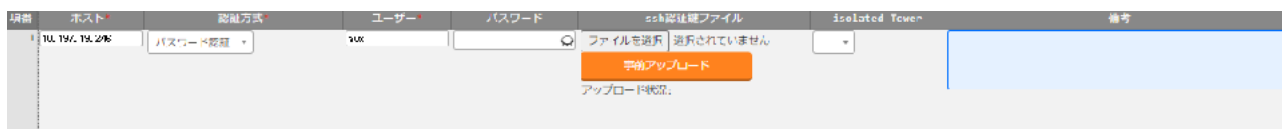


図 5.2-4 登録画面(Ansible Tower ホスト一覧)

(2) AnsibleTower ホスト一覧画面の項目一覧は以下のとおりです。

表 5.2-2 登録画面項目一覧(Ansible Tower ホスト一覧)

項目	説明	入力必須	入力形式	制約事項
ホスト	AnsibleTower サーバのホスト名(または IP アドレス)を入力します。 HTTPS 通信の場合はホスト名が推奨です。	○	手動入力	最大長 128 バイト
認証方式	AnsibleTower サーバへファイル転送(scp)で接続する際の認証方式を選択します。	○	手動入力	最大長 30 バイト
ログインユーザ	AnsibleTower サーバへファイル転送(scp)で接続する際のログインユーザを入力します。 ログインユーザ、AnsibleTower インストール時に生成される awx ユーザーにパスワードを設定し、使用す	○	手動入力	最大長 30 バイト



項目	説明	入力 必須	入力形式	制約事項
	ることを強く推奨します。			
パスワード	認証方式でパスワード認証を選択した場合に必須入力となります。 ログインユーザのパスワードを指定します。	－	手動入力	最大長 30 バイト
ssh 認証鍵ファイル	認証方式で鍵認証を選択した場合で、ssh 認証鍵ファイルを指定して鍵認証する場合のファイルを入力します。	－	ファイル 選択	最大サイズ 10K バイト
isolated Tower	クラスター構成で構築されている場合で、isolated Tower の場合に「●」を選択します。	－	選択方式	
備考	自由記述欄です。	－	手動入力	最大長 4000 バイト

### 5.2.3 グローバル変数管理

- (1) [グローバル変数管理]では、Playbook や対話ファイルなどで使用するグローバル変数名を登録／更新／廃止を行います。

Exastro IT Automation Ansible共通

ようこそ[システム管理者]さん  
ログインID [administrator]  
パスワード変更 ログアウト

Menu  
メインメニュー  
インターフェース情報  
**グローバル変数管理**  
ファイル管理  
テンプレート管理

説明  
表示フィルタ  
一覧/更新  
登録  
全件ダウンロードとファイルアップロード編集  
変更履歴

廃止	項目	グローバル変数名	具体値	変数名説明	備考	最終更新日時	最終更新者
廃止含まず	～					～	
	▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索	▼ブルダウン検索		▼ブル

フィルタ フィルタクリア

☒ オートフィルタ

管理者に連絡

図 5.2-5 サブメニュー画面(グローバル変数管理)

- (2) 「登録」-「登録開始」ボタンより、オペレーション情報の登録を行います。

登録 △開じる

項目	グローバル変数名*	具体値*	変数名説明	備考	最終更新日時	最終更新者
自動入力					自動入力	自動入力

※\*は必須項目です。

戻る 登録

図 5.2-6 登録画面(グローバル変数管理)

(3) グローバル変数管理画面の項目一覧は以下のとおりです。

表 5.2-3 登録画面項目一覧(グローバル変数管理)

項目	説明	入力 必須	入力形式	制約事項
グローバル変数名	変数名を入力します。 変数名は、「GBL_****」形式で入力します。 ****: 半角英数字とアンダースコア( _ )が利用可能です。(最小値:1 バイト、最大値:128 バイト)	○	手動入力	説明欄記載のとおり
具体値	具体値を入力します。複数行の具体値も入力できますが、Pioneer で使用しているグローバル変数で複数行の具体値を設定すると、作業実行時にエラーとなります。 具体値にファイル埋込変数「CPF_」とテンプレート埋込変数「TPF_」が入力出来ます。変数を記述する場合、Playbook に変数を記述する場合と同様、変数名を{{ }}で囲みます。 入力例) 具体値に TPF_sample を入力する場合 '{{△TPF_sample△}}'      △:半角スペース	○	手動入力	最大長 8192 バイト
変数名説明	変数に対する説明・コメントを入力します	-	手動入力	最大長 256 バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

## 5.2.4 テンプレート管理

- (1) [テンプレート管理]では、Playbook で定義している template モジュールや ios\_config モジュールなどのパラメータで使用する Jinja2 テンプレートファイルと template 埋め込み変数の登録／更新／廃止を行います。

テンプレート管理でテンプレートモジュールを登録しておくことで、Playbook 内で定義している template モジュールなどで使用する template ファイルを template 埋め込み変数で指定することが出来ます。

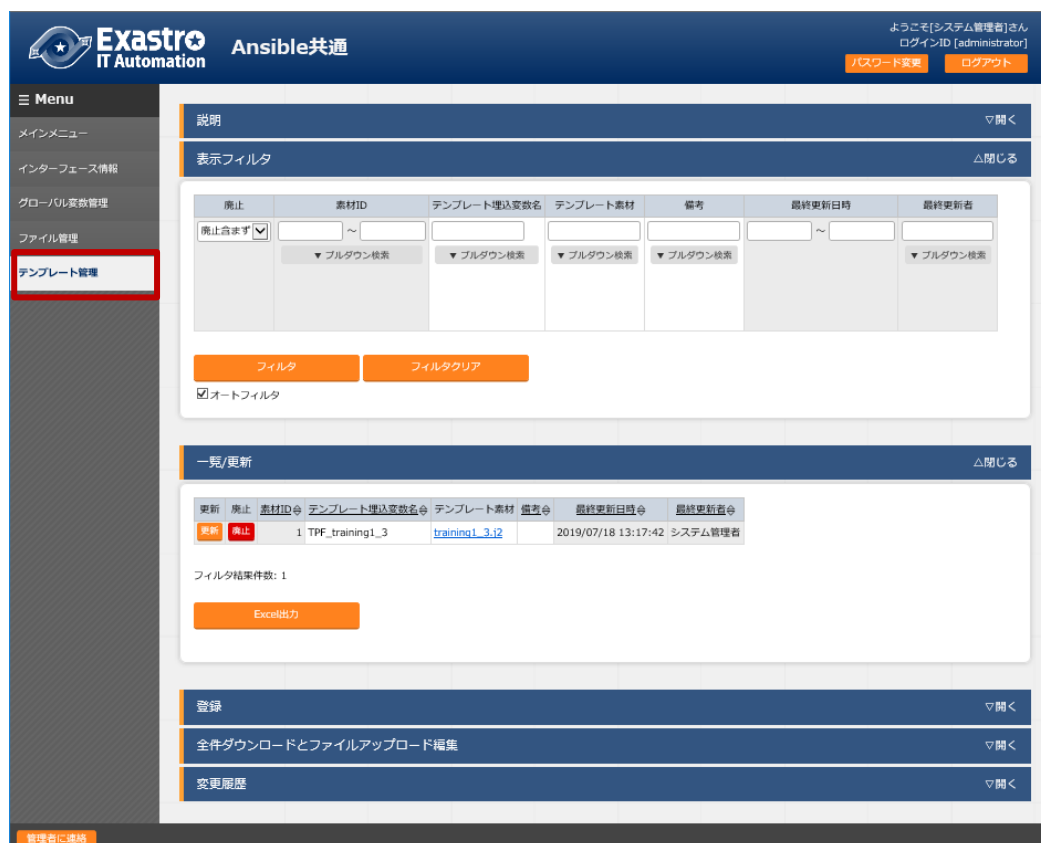


図 5.2-7 サブメニュー画面(テンプレート管理)

- (2) 「登録」-「登録開始」ボタンより、ファイル管理情報の登録を行います。

素材ID	テンプレート埋込変数名*	テンプレート素材*	備考	最終更新日時	最終更新者
自動入力		<div>事前アップロード</div> <div>アップロード状況:</div>		自動入力	自動入力

図 5.2-8 登録画面(テンプレート管理)

(3) 登録画面の項目は以下のとおりです。

表 5.2-4 登録画面項目一覧(テンプレート管理)

項目	説明	入力 必須	入力形式	制約事項
テンプレート 埋込変数名	template モジュールや ios_config モジュールなどのパラメータに埋め込む変数名を入力します。 変数名は、「TPF_****」形式で入力します。 ****: 半角英数字とアンダースコア( _ )が利用可能です。(最小値:1 バイト、最大値:128 バイト)	○	手動入力	説明欄記載のとおり
テンプレート素材	モジュールのパラメータで使用する Jinja2 テンプレートファイルをアップロードします。	○	ファイル 登録	テキスト形式
変数定義	テンプレート素材で使用している変数を定義します。 具体値は定義のみで使用することはありません。 Ansible-Role のみで使用するテンプレートで、default 変数定義ファイルなどに変数定義をしている場合、変数定義は省略できます。 同名の変数を複数のテンプレートで使用する場合、変数定義を合わせる必要があります。変数定義が一致していない場合は登録でエラーとなります。 変数定義は Ansible の仕様に準拠していますが、ITA 独自仕様があります。表 5.2-5-1 に変数定義の留意事項を記載します。また、変数名の命名規則は表 2.1 変数の種類と同様です。	-	手動入力	最大長 4000 バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

「登録」の前に、「テンプレート素材」を「事前アップロード(①)」してください。「アップロード状況(②)」に Playbook のファイル名が表示されたのを確認してから、「登録」ボタンを押してください。

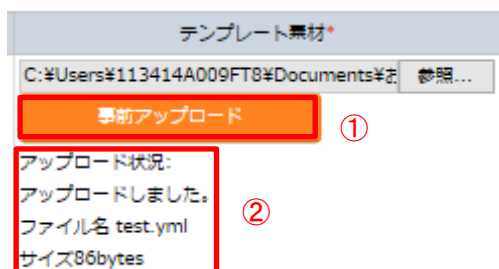


表 5.2-5 変数定義の留意事項

種類	留意事項
通常変数	具体値は省略可能です e.g.) VAR_sample_1: none VAR_sample_2:
複数具体値変数	具体値は省略可能です e.g.)

	<pre>VAR_sample_1: - none VAR_sample_2: []</pre>
多段変数	<p>階層化された変数構造が定義出来ます。</p> <p>e.g.)</p> <pre>VAR_sample_1: - item1: none   item2: VAR_sample_2: - array: - item1: none   item2:</pre> <p>多段変数を定義したテンプレートは Ansible-Role でのみ使用することが出来ます。</p> <p>Ansible-Role で使用する場合、同名の変数を default 変数定義ファイルなどに定義している場合は、変数定義を合わせる必要があります。変数定義が一致していない場合は登録でエラーとなります。</p>
グローバル変数	<p>具体値は省略可能です</p> <p>e.g.)</p> <pre>GBL_sample_1: none GBL_sample_2:</pre>
ITA 独自変数	変数の定義は不要です。
読替変数	<p>定義出来る変数の種類は以下の 3 種類です。</p> <ul style="list-style-type: none"> <li>・通常変数</li> <li>・複数具体値変数</li> <li>・多段変数</li> </ul> <p>各変数定義の留意事項は同様です。</p> <p>e.g.)</p> <pre>LCA_sample_1: LCA_sample_2: [] LCA_sample_3: - item1: none   item2:</pre> <p>読替変数を定義したテンプレートは Ansible-Role でのみ使用することが出来ます。</p>

詳しくは、別資料「利用手順マニュアル\_Ansible-driver\_別紙\_Ansible 利用ガイドライン\_追加ルール」を参照してください。

## ① Playbook の記述

テンプレート管理で登録したテンプレートを Playbook に記述する場合、該当のパラメータにテンプレート埋込変数名を記述します。テンプレート埋込変数名を使用しない場合、代入値管理で登録した変数や該当ファイルのパスを記述します。

Exp)

### Playbook の記述

- template: src='{{△ TPF\_hosts△}}' dest=/etc/hosts

△:半角スペース

登録内容

テンプレート埋込変数名	テンプレート素材
TPF_hosts	/etc/hosts

**\_dest はファイル名も記述してください。ファイル名の指定がない場合、登録したテンプレート素材のファイル名の前に ITA の管理番号が付与された名前のファイル名で処理されます。**

**たとえば、dest=/etc/ とした場合、ファイル名は/etc/10 桁の数値 \_hosts となります。**

## ② 対話ファイルの記述

対話ファイルに記述する場合、テンプレート埋込変数名を記述します。

Exp)

### 対話ファイルの記述

- expect: '{{△\_loginuser\_△}}@{{△\_loginhostname\_△}}'

exec: 'scp △ITA ユーザ@ITA ホスト名:{{△ TPF\_hosts△}}△転送先'

- expect: 'password:'

exec: ITA ユーザのパスワード'

△:半角スペース

登録内容

テンプレート埋込変数名	テンプレート素材
TPF_hosts	/etc/hosts

**転送先はファイル名も記述してください。ファイル名の指定がない場合、登録したファイル素材のファイル名の前に ITA の管理番号が付与された名前のファイル名で処理されます。**

**例えば、転送先=/etc/ とした場合、ファイル名は/etc/10 桁の数字\_hosts となります。**

**{{△ TPF\_hosts△}}は実行時に転送元の絶対パスに置換されます。**

内部の処理でテンプレートの変数定義を読み取り、「[5.3.9 代入値自動登録設定](#)」や「[5.3.11 代入値管理](#)」で具体値が登録可能になります。

読み取りのタイミングはリアルタイムではないので、「[5.3.9 代入値自動登録設定](#)」や「[5.3.11 代入値管理](#)」で変数が扱えるまでに時間がかかる<sup>※1</sup>場合があります。

**※1 読み取りのタイミングは「[7.2 メンテナンス方法について](#)」に記載していますので、そちらをご参照ください。**

## 5.2.5 ファイル管理

- (1) [ファイル管理]では、Playbook 内で定義している各モジュールで使用するファイルとファイル埋め込み変数の登録／更新／廃止を行います。

ファイル管理でファイル素材を登録しておくことで、Playbook 内で定義している各モジュールで使用するファイルをファイル埋め込み変数名で指定することが出来ます。

The screenshot shows the 'Exastro IT Automation Ansible共通' interface. The left sidebar has a menu with 'ファイル管理' (File Management) highlighted. The main content area is divided into sections: '説明' (Description), '表示フィルタ' (Display Filter), '一覧/更新' (List/Update), and '登録' (Registration). The '表示フィルタ' section contains a table with columns for '廃止' (Delete), '素材ID' (Material ID), 'ファイル埋込変数名' (File variable name), 'ファイル素材' (File material), '備考' (Remarks), '最終更新日時' (Last updated time), and '最終更新者' (Last updated by). The '一覧/更新' section contains a table with columns for '更新' (Update), '廃止' (Delete), '素材ID' (Material ID), 'ファイル埋込変数名' (File variable name), 'ファイル素材' (File material), '備考' (Remarks), '最終更新日時' (Last updated time), and '最終更新者' (Last updated by). The '登録' section contains links for '全件ダウンロードとファイルアップロード編集' (Download all and edit file upload) and '変更履歴' (Change history).

図 5.2-9 サブメニュー画面(ファイル管理)

- (2) 「登録」-「登録開始」ボタンより、ファイル管理情報の登録を行います。

The screenshot shows the registration form for file management. It includes fields for '素材ID' (Material ID), 'ファイル埋込変数名\*' (File variable name\*), 'ファイル素材\*' (File material\*), and '備考' (Remarks). There is a '事前アップロード' (Pre-upload) button and a 'アップロード状況' (Upload status) section. The form also includes '最終更新日時' (Last updated time) and '最終更新者' (Last updated by) fields.

図 5.2-10 登録画面(ファイル管理)

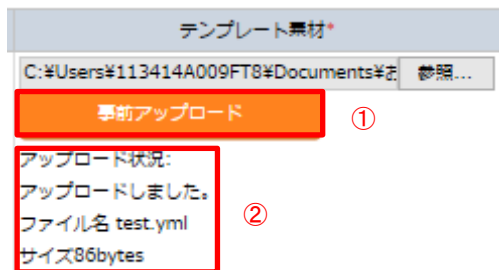


(3) 登録画面の項目は以下のとおりです。

表 5.2-6 登録画面項目一覧(ファイル管理)

項目	説明	入力 必須	入力形式	制約事項
ファイル埋込 変数名	各モジュールのパラメータに埋め込む変数名を入力します。 変数名は、「CPF_****」形式で入力します。 ****: 半角英数字とアンダースコア( _ )が利用可能です。 (最小値:1 バイト、最大値:128 バイト)	○	手動入力	説明欄記載のとおり
ファイル素材	各モジュールで使用するファイルをアップロードします。	○	ファイル 登録	最大サイズ 20M バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

「登録」の前に、「ファイル素材」を「事前アップロード(①)」してください。「アップロード状況(②)」に Playbook のファイル名が表示されたのを確認してから、「登録」ボタンを押してください。



## ① Playbook の記述

プレイブックの各モジュールに記述する場合、ファイル埋め込み変数を記述します。

e.g)

### Playbook の記述

登録内容

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

△:半角スペース

ファイル埋込変数名	ファイル素材
CPF_hosts	hosts

dest はファイル名も記述してください。ファイル名の指定がない場合、登録した**ファイル素材のファイル名の前に ITA の管理番号が付与された名前のファイル名**で処理されます。

たとえば、dest=/etc/ とした場合、ファイル名は/etc/10 桁の数字\_hosts となります。

```
-unarchive src={{△CPF_tool_tgz△}} dest=/usr/local/bin
```

△:半角スペース

ファイル埋込変数名	ファイル素材
CPF_tool_tgz	tool.tgz

## ② 対話ファイルの記述

対話ファイルに記述する場合、ファイル埋め込み変数を記述します。

e.g)

### 対話ファイルの記述

登録内容

```
- expect: '{{△__loginuser__△}}@{{△__loginhostname__△}}'
```

```
  exec: 'scp △ITA ユーザ@ITA ホスト名:{{△CPF_hosts△}}△転送先'
```

```
- expect: 'password:'
```

```
  exec: 'ITA ユーザのパスワード'
```

△:半角スペース

ファイル埋込変数名	ファイル素材
CPF_hosts	hosts

転送先はファイル名も記述してください。ファイル名の指定がない場合、登録した**ファイル素材のファイル名の前に ITA の管理番号が付与された名前のファイル名**で処理されます。

たとえば、転送先=/etc/ とした場合、ファイル名は/etc/10 桁の数字\_hosts となります。

{{△CPF\_hosts△}}は実行時に転送元の絶対パスに置換されます。

## 5.3 Ansibel-Legacy／Legacy Role／Pioneer コンソール

Ansibel-Legacy／Legacy Role／Pioneer コンソールの操作です。

### 5.3.1 OS 種別マスタ

- (1) [OS 種別マスタ] 画面では、ITA の Pioneer より操作対象となる機器の OS 種別を管理します。  
※本メニューは Ansible-Pioneer コンソールにのみ存在します。

The screenshot shows the Exastro IT Automation console interface. The sidebar menu on the left has 'OS種別マスタ' (OS Type Master) highlighted. The main content area displays a table with columns for OS Type ID, OS Type Name, Device Type (SV, NW, ST), and other details. Below the table, there are buttons for 'フィルタ' (Filter) and 'フィルタクリア' (Clear Filter). At the bottom, there are buttons for '登録' (Register), '全件ダウンロードとファイルアップロード編集' (Download all and edit file upload), and '変更履歴' (Change History).

図 5.3-1 サブメニュー画面(OS種別マスタ)

- (2) 「登録」-「登録開始」ボタンより、OS 情報の登録を行います。

The screenshot shows the registration form for OS types. It includes a table with columns for OS Type ID, OS Type Name, Device Type (SV, NW, ST), and a large text area for comments. There are also fields for '自動入力' (Auto Input) and '最終更新日時' (Last Update Time).

図 5.3-2 登録画面(OS種別マスタ)

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-2 登録画面項目一覧(OS種別マスタ)

項目		説明	入力 必須	入力形式	制約事項
OS 種別 ID		登録情報を識別する一意のIDが自動入力されます	○	自動入力	-
OS 種別名		任意の機器名称を入力します	○	手動入力	最大長 256 バイト
機種別	SV	機器種別がサーバの場合「●」を選択します	-	リスト選択	-
	NW	機器種別がネットワーク機器の場合「●」を選択します	-	リスト選択	-
	ST	機器種別がストレージ機器の場合「●」を選択します	-	リスト選択	-
備考		自由記述欄です	-	手動入力	-

## 5.3.2 Movement 一覧

(1) [Movement 一覧]では Movement 名の登録／更新／廃止を行います。

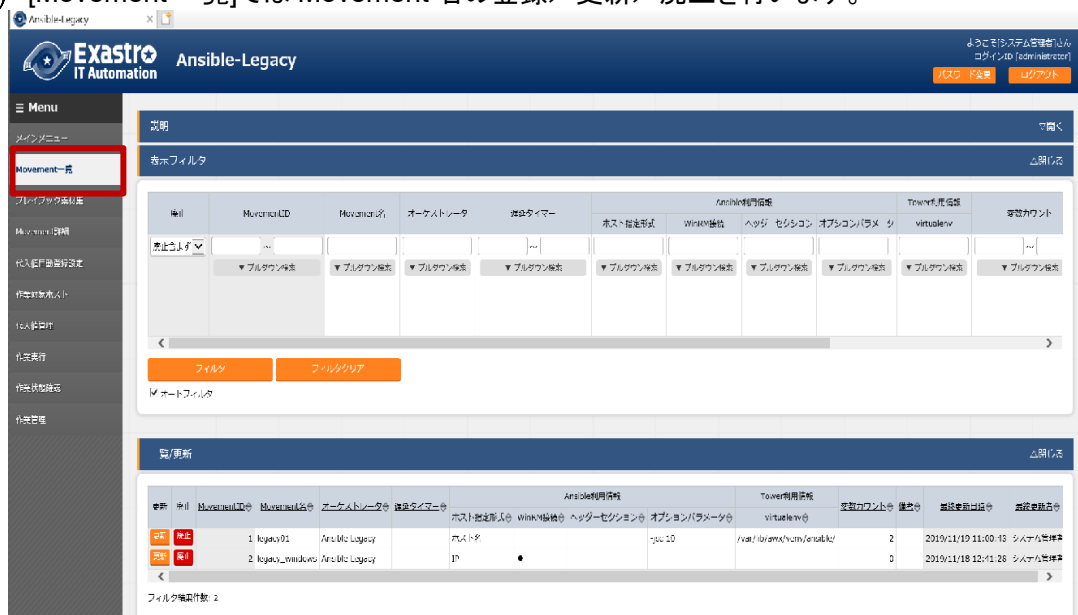


図 5.3-3 サブメニュー画面 (Movement 一覧)

(2) 「登録」・「登録開始」ボタンより、Movement 情報の登録を行います。

Movement 名*	遅延タイム*	Ansible利用情報				Tower利用情報
		ホスト指定形式*	WinRM接続	ヘッダーセクション	オプション/パラメータ	virtualenv
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

図 5.3-4 登録画面 (Movement 一覧)

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-2 登録画面項目一覧(Movement 一覧)

項目	説明	入力 必須	入力形式	制約事項
Movement 名	Movement の名称を入力します。	○	手動入力	最大長 256 バイト
遅延タイマー	Movement が指定期間遅延した場合にステータスを遅延として警告表示したい場合に指定期間(1～)を入力します。(単位:分) 未入力の場合は警告表示しません。	-	手動入力	-
ホスト指定形式	特別に IP アドレスで表現しないホストを指定したい場合に「ホスト名」を選択します。通常は IP が推奨です。	○	リスト選択	-
並列実行数 ※ Pioneer の Movement 一覧でのみ表示されます。	Ansible が同時実行可能な対象ホスト数を入力します。 この設定値以上の対象ホストがある場合、設定値分の対象ホストに対して実行の後、残りの対象ホストをランダムに実行していきます。 ■未入力時の振る舞いについて ・Ansible ドライバーの場合は、サーバの設定ファイル (/etc/ansible.conf) の内容がデフォルト値となります。 ・Ansible Tower ドライバーの場合は、Ansible Tower サーバのデフォルト値となります。	-	手動入力	NULL または整数
WinRM 接続 *1	対象ホストが Windows Server の場合に、「●」を選択します。	-	リスト選択	-
ヘッダーセクション ※ Pioneer の Movement 一覧では表示されません。	ITA が自動生成する親 Playbook の先頭から tasks または roles セクションまでのセクションを編集します。 未入力の場合は、以下を適用します。 Ansible: - hosts: all remote_user: ¥"{{ __loginuser__ }}¥" gather_facts: no become: yes  Ansible Tower: - hosts: all gather_facts: no become: yes ※winrm 接続の場合は become: yes は適用しません。	-	手動入力	最大長 512 バイト
オプションパラメータ	Movement 固有の ansible-playbook コマンドのオプションパラメータを入力します。 実行エンジンが Ansible の場合: ansible-playbook コマンドのオプションパラメータを入力します。 -i オプションは ITA が設定します。  実行エンジンが Ansible Tower の場合: 以下のオプションパラメータが設定できます。 -verbosity -f FORKS,--forks=FORKS	-	手動入力	最大長 512 バイト

	-I SUBSET,--limit=SUBSET -e EXTRA_VARS,--extra-vars=EXTRA_VARS EXTRA_VARS: 変数名=具体値 変数名=具体値 ..... -t TAGS,--tags=TAGS -b,--become -D,--diff --skip-tags=SKIP_TAGS --start-at-task=START_AT_TASK 以下は Ansible Tower 独自のオプションパラメータです。 -ufc,--use_fact_cache ファクトのキャッシュの使用 -as,--allow_simultaneous 同時実行ジョブの有効化 -jsc,--job_slice_count= ジョブスライス数 Ansible Tower 独自のオプションパラメータについては、Ansible Tower ユーザーガイドのジョブテンプレートの説明を参照して下さい。			
<b>Virtualenv</b> ※実行エンジンが AnsibleTower の場合に表示されます。	virtualenv で構築した ansible 実行環境を選択します。 未選択の場合は Tower インストール時にインストールされた ansible 実行環境を使用します。	-	リスト選択	
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

\*1 WindowsServer 以外の場合は、必ずリストから空白を選択してください。なお、Ansible-Pioneer は常に gather\_facts=false で動作します。

#### 【留意事項】

WinRM 接続で「●」を選択した場合は接続するホストをすべて WindowsServer とみなします。

### 5.3.3 プレイブック素材集(Ansible-Legacy のみ)

- (1) [プレイブック素材集]ではユーザーが作成した Playbook の登録／更新／廃止を行います。  
※本メニューは Ansible-Legacy コンソールにのみ存在します。  
Playbook の記述など関しては、「[6.1Playbook\(Ansible-Legacy\)の記述](#)」を参照してください。

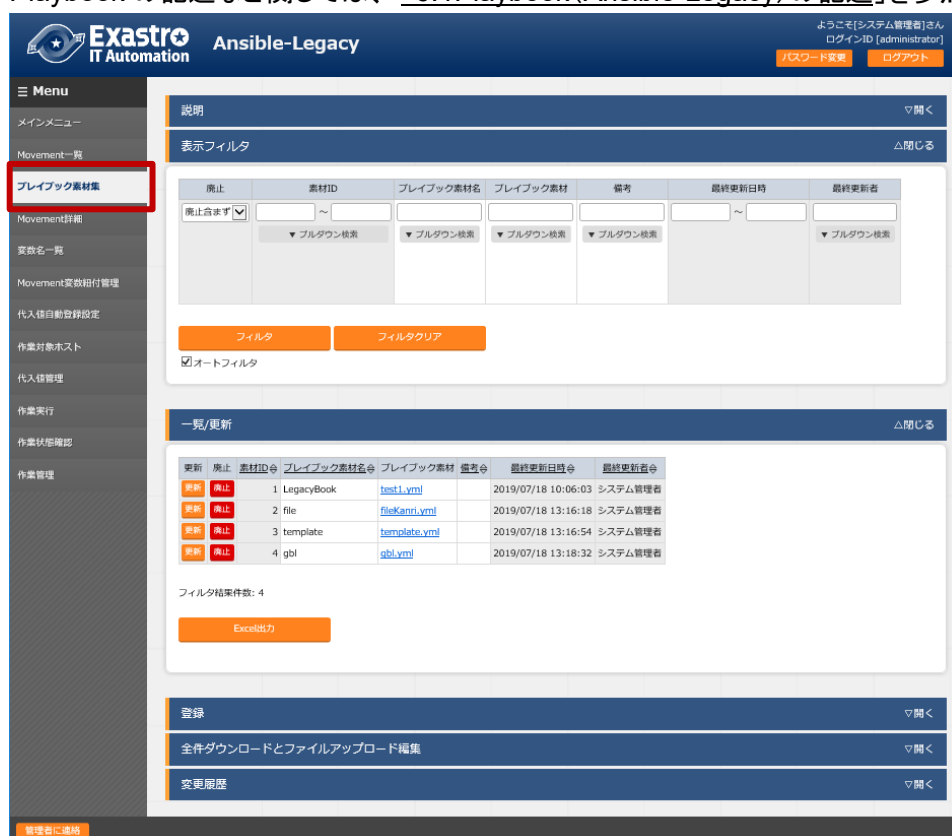


図 5.3-5 サブメニュー画面(プレイブック素材集)

- (2) 「登録」-「登録開始」ボタンより、Playbook の登録を行います。

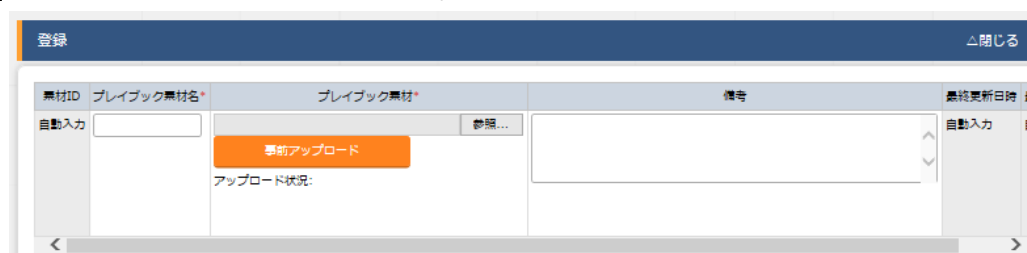


図 5.3-6 登録画面(プレイブック素材集)



(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-3 登録画面項目一覧(プレイブック素材集)

項目	説明	入力 必須	入力形式	制約事項
プレイブック素材名	ITA で管理するプレイブック素材名を入力します。	○	手動入力	最大長 256 バイト
プレイブック素材	作成した Playbook ファイルをアップロードします。 アップロードする Playbook ファイルは文字コードが UTF-8 の BOM で作成して下さい。 文字コードが UTF-8 の BOM なし以外の Playbook ファイルはアップロードでエラーになります。	○	ファイル選択	最大サイズ 20M バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

「登録」の前に、「プレイブック素材」を「事前アップロード(①)」してください。「アップロード状況(②)」に Playbook のファイル名が表示されたのを確認してから、「登録」ボタンを押してください。



内部の処理で Playbook ファイル内に定義している変数を抽出します。抽出した変数は、「5.3.11 [代入値管理](#)」や「5.3.9 代入値自動登録設定」で具体値の登録が可能になります。

抽出する タイミングはリアルタイムではありませんので、「5.3.12 [代入値管理](#)」や「5.3.9 代入値自動登録設定」で変数が扱えるまでに時間がかかる※1 場合があります。

※1 抽出のタイミングは「[7.2 メンテナンス方法について](#)」に記載していますので、そちらをご参照ください

### 5.3.4 ロールパッケージ管理 (Ansible-Legacy Role のみ)

(1) ユーザーが作成したロールパッケージファイルの登録／更新／廃止を行います。

※本メニューは Ansible-Legacy Role コンソールにのみ存在します。

ロールパッケージファイルは、「roles」のある階層のディレクトリを zip にて圧縮したものを登録してください。ロールパッケージディレクトリ構成などは「[6.3 ロールパッケージ \(Ansible-Legacy Role\) の記述](#)」を参照してください。

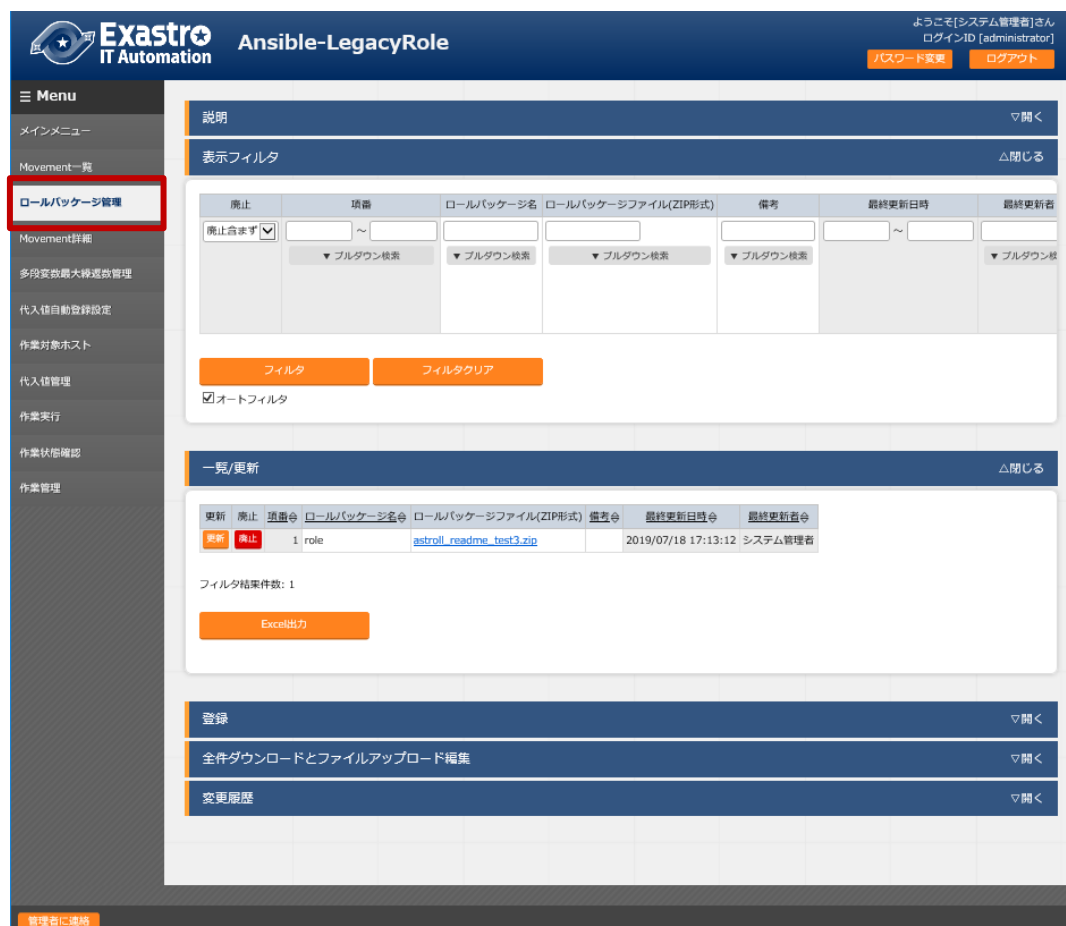


図 5.3-7 サブメニュー画面 (ロールパッケージ管理)

(2) 「登録」-「登録開始」ボタンより、ロールパッケージ情報の登録を行います。

項目	ロールパッケージ名*	ロールパッケージファイル (ZIP形式)*	備考	最終更新日時	最終更新者
自動入力	<input type="text"/>	<input type="button" value="選択アップロード"/> <input type="button" value="参照..."/>	<input type="text"/>	自動入力	自動入力
		アップロード状況:			

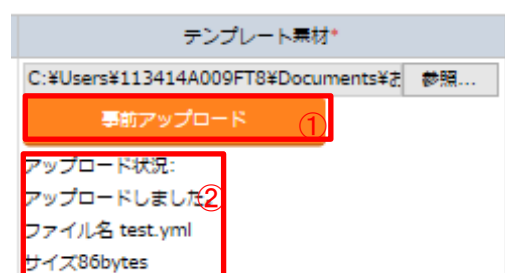
図 5.3-8 登録画面 (ロールパッケージ管理)

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-4 登録画面項目一覧(ロールパッケージ管理)

項目	説明	入力必須	入力形式	制約事項
ロールパッケージ名	ITA で管理するロールパッケージ名を入力します。	○	手動入力	最大長 256 バイト
ロールパッケージファイル	作成したロールパッケージファイル(zip 形式)をアップロードします。 アップロードするロールパッケージファイルに含まれる Playbook ファイルは文字コードが UTF-8 で BOM なしで作成して下さい。 UTF-8 で BOM なし以外の Playbook ファイルが含まれていると登録時にエラーとなります。 詳しくは、6.3 ロールパッケージ(Ansible-Legacy Role)の記述を参照下さい。	○	ファイル選択	最大サイズ 20M バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

「登録」の前に、「ロールパッケージファイル」を「事前アップロード(①)」してください。「アップロード状況(②)」にロールパッケージファイル名が表示されたのを確認してから、「登録」ボタンを押してください。



内部の処理でロールパッケージ内に定義している変数を取り出します。取り出した変数は、「5.3.9 代入値自動登録設定」や「5.3.11 代入値管理」で具体値の登録が可能になります。

抽出するタイミングはリアルタイムではありませんので、「5.3.11 代入値管理」や「5.3.9 代入値自動登録設定」で変数が扱えるまでに時間がかかる※1 場合があります。

※1 抽出のタイミングは「7.2 起動周期の変更」に記載していますので、そちらをご参照ください

### 5.3.5 対話種別リスト(Ansible-Pioneer のみ)

- (1) [対話種別リスト]では、対話種別の登録／更新／廃止を行います。  
 ※本メニューは Ansible-Pioneer コンソールにのみ存在します。  
 Ansible-Pioneer では、「OS 種別」ごとの差異を対話ファイルごとに定義し、同一目的の対話ファイルを「対話種別」として纏めて機器差分を吸収(抽象化)します。

The screenshot shows the Ansible-Pioneer web interface. The sidebar on the left has a menu with '対話種別リスト' (Dialog Type List) highlighted. The main content area has a header with '説明' (Description) and '表示フィルタ' (Display Filter). Below this is a search filter section with fields for '廃止' (Delete), '項目' (Item), '対話種別名' (Dialog Type Name), '備考' (Remarks), '最終更新日時' (Last Update Date/Time), and '最終更新者' (Last Updated By). There are buttons for 'フィルタ' (Filter) and 'フィルタクリア' (Clear Filter). Below the search section is a table with columns for '更新' (Update), '廃止' (Delete), '対話種別名' (Dialog Type Name), '備考' (Remarks), '最終更新日時' (Last Update Date/Time), and '最終更新者' (Last Updated By). The table shows one entry with the name 'list'. Below the table is a button for 'Excel出力' (Excel Output). At the bottom, there are buttons for '登録' (Registration), '全件ダウンロードとファイルアップロード編集' (Download all and edit file upload), and '変更履歴' (Change History).

図 5.3-9 サブメニュー画面(対話種別リスト)

- (2) 「登録」-「登録開始」ボタンより、オペレーション情報の登録を行います。

The screenshot shows the registration form for dialog types. It has a table with five columns: '項目' (Item), '対話種別名' (Dialog Type Name), '備考' (Remarks), '最終更新日時' (Last Update Date/Time), and '最終更新者' (Last Updated By). The '項目' column has a dropdown menu with '自動入力' (Automatic Input) selected. The '対話種別名' column has a text input field. The '備考' column has a large text area. The '最終更新日時' column has a date and time input field. The '最終更新者' column has a text input field.

図 5.3-10 登録画面(対話種別リスト)

- (3) 登録画面の項目一覧は以下のとおりです。

表 5.3-5 登録画面項目一覧(対話種別リスト)

項目	説明	入力必須	入力形式	制約事項
対話種別名	対話種別名を入力します	○	リスト選択	最大長 256 バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

### 5.3.6 対話ファイル素材集(Ansible-Pioneer のみ)

- (1) [対話ファイル素材集]では、ユーザーが作成した対話ファイルの登録／更新／廃止を行います。  
※本メニューは、Ansible-Pioneer コンソールにのみ存在します。
- (2) 対話ファイルの記述などに関しては、「[6.2 対話ファイル\(Ansible-Pioneer\)の記述](#)」を参照してください。

「対話種別」と「OS 種別」の組み合わせごとに対話ファイルを登録します。

1つの「対話種別」で複数の OS に対応させたい場合は、同じ「対話種別」で、「OS 種別」それぞれについて対話ファイルを登録してください。

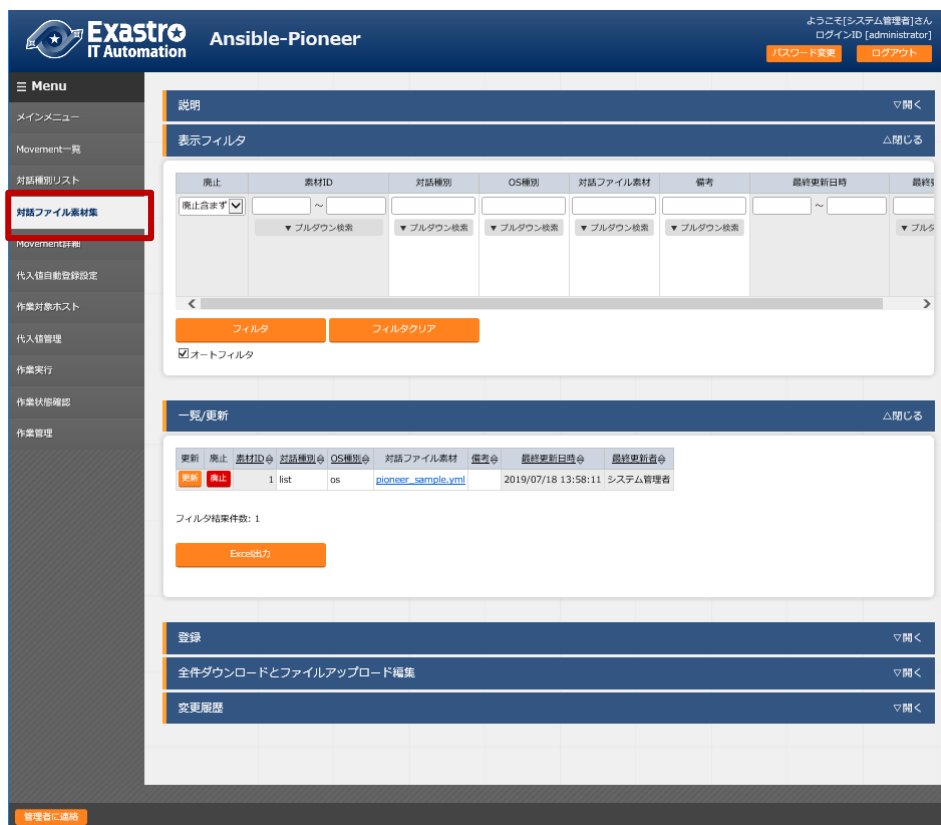


図 5.3-11 サブメニュー画面(対話ファイル素材集)

- (3) 「登録」-「登録開始」ボタンより、対話ファイル素材の登録を行います。

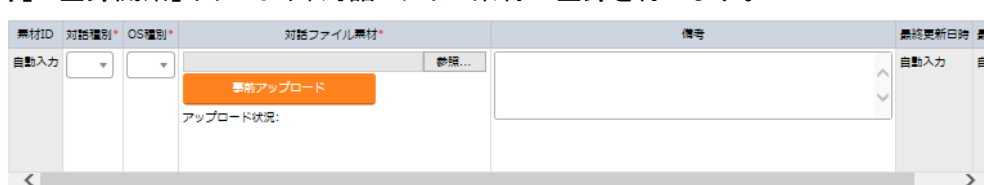


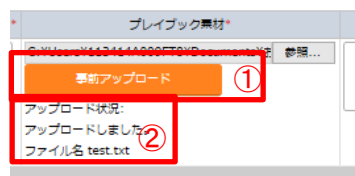
図 5.3-12 登録画面(対話ファイル素材集)

(4) 登録画面の項目一覧は以下のとおりです。

表 5.3-6 登録画面項目一覧(対話ファイル素材集)

項目	説明	入力 必須	入力形式	制約事項
対話種別	対話種別リストに登録されている対話種別が表示されます。登録する対話ファイルの対話種別を選択します。	○	リスト選択	-
OS 種別	OS 種別マスタに登録されている OS 種別が表示されます。登録する対話ファイルの OS 種別を選択します。	○	リスト選択	-
対話ファイル素材	対話種別と OS 種別に対応する対話ファイルをアップロードします。 アップロードする対話ファイルは文字コードが UTF-8 の BOM なしで作成して下さい。 文字コードが UTF-8 の BOM なし以外の対話ファイルはアップロードでエラーになります。	○	ファイル 登録	最大サイズ 20M バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

「登録」の前に、「対話ファイル」を「事前アップロード(①)」してください。「アップロード状況(②)」に対話ファイルのファイル名が表示されたのを確認してから、「登録」ボタンを押してください。



内部の処理で対話ファイル内に定義している変数を取り出します。取り出した変数は、「5.3.9 代入値自動登録設定」や「5.3.11 代入値管理」で具体値の登録が可能になります。

抜出するタイミングはリアルタイムではありませんので、「5.3.11 代入値管理」や「5.3.9 代入値自動登録設定」で変数が扱えるまでに時間がかかる※1場合があります。

※1 抜出のタイミングは「7.2 メンテナンス方法について」に記載していますので、そちらをご参照ください

### 5.3.7 Movement 詳細

(1) [Movement 詳細]では、Movement で実行する素材の登録／更新／廃止を行います。

ようこそ[システム管理者]さん  
ログインID [administrator]  
パスワード変更 ログアウト

Menu  
メインメニュー  
Movement一覧  
プレイブック素材集  
**Movement詳細**  
変数一覧  
Movement変数紐付管理  
代入値自動登録設定  
作業対象ホスト  
代入値管理  
作業実行  
作業状態確認  
作業管理

説明

表示フィルタ

廃止	紐付項目	Movement	プレイブック素材	インクルード順序	備考	最終更新日時
廃止含まず	▼ プルダウン検索	▼ プルダウン検索	▼ プルダウン検索	▼ プルダウン検索	▼ プルダウン検索	▼ プルダウン検索

フィルタ フィルタクリア

☒ オートフィルタ

一覧/更新

更新	廃止	紐付項目	Movement	プレイブック素材	インクルード順序	備考	最終更新日時	最終更新者
更新	廃止	1	3:Legacy_movement	LegacyBook	1		2019/07/18 10:06:13	システム管理者
更新	廃止	2	3:Legacy_movement	file	2		2019/07/18 13:19:26	システム管理者
更新	廃止	3	3:Legacy_movement	gbl	3		2019/07/18 13:19:36	システム管理者
更新	廃止	4	3:Legacy_movement	template	4		2019/07/18 13:19:47	システム管理者

フィルタ結果件数: 4

Excel出力

登録

全件ダウンロードとファイルアップロード編集

変更履歴

管理者に連絡

図 5.3-13 サブメニュー画面(Movement 詳細)

※画面は Ansible-Legacy のものです。

(2) 「登録」-「登録開始」ボタンより、Movement 詳細の登録を行います。

登録

紐付項目 Movement\* プレイブック素材\* インクルード順序\* 備考 最終更新日時 最終更新者

自動入力

1:LegacyMove

※\*は必須項目です。

戻る 登録

図 5.3-14 登録画面(Movement 詳細)

(3) 登録画面の項目一覧は以下のとおりです。

- Ansible-Legacy の場合

表 5.3-7 登録画面項目一覧(Movement 詳細 Ansible-Legacy の場合)

項目	説明	入力 必須	入力形式	制約事項
Movement	Movement 一覧で登録した Movement が表示されます。 Movement を選択します。	○	リスト選択	-
プレイブック素材	「 <a href="#">5.3.3 プレイブック素材集 (Ansible-Legacy のみ)</a> 」で登録したプレイブック素材が表示されます。プレイブック素材を選択します。	○	リスト選択	-
インクルード順序	プレイブック素材の実行順序(1～:一意値)を入力します。 入力されたインクルード順序(昇順)でプレイブック素材が実行されます。	○	手動入力	半角整数
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

- Ansible-Legacy Role の場合

表 5.3-8 登録画面項目一覧(Movement 詳細 Ansible-Legacy の場合)

項目	説明	入力 必須	入力形式	制約事項
Movement	Ansible-Legacy と同様。	○	リスト選択	-
ロールパッケージ名	ロールパッケージ管理で登録したロールパッケージが表示されます。実行するロールパッケージを選択します。	○	リスト選択	-
ロール名	ロールパッケージ名で選択したロールパッケージに含まれているロール名が表示されます。実行するロールパッケージ内のロールを選択します。	○		-
インクルード順序	Ansible-Legacy と同様。	○	手動入力	半角整数
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

**【注意】**

同一 Movement に複数のロールパッケージを登録しないでください。

複数のロールパッケージが登録されている Movement を作業実行した場合、想定外エラーとなります。

- Ansible-Pioneer の場合

表 5.3-9 登録画面項目一覧(Movement 詳細 Ansible-Pioneer の場合)

項目	説明	入力 必須	入力形式	制約事項
Movement	Ansible-Legacy と同様。	○	リスト選択	-
対話種別	「 <a href="#">対話種別リスト (Ansible-Pioneer のみ)</a> 」で登録した対話種別が表示されます。実行する対話種別を選択します。 ホスト毎に OS 種別と対話種別に関連付く対話ファイルが実行対象となります。	○	リスト選択	-
インクルード順序	Ansible-Legacy と同様。	○	手動入力	半角整数
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト



### 5.3.8 多段変数最大繰返数管理 (Ansible-Legacy Role のみ)

- (1) [多段変数最大繰返管理]では、「5.3.4 ロールパッケージ管理 (Ansible-Legacy Role のみ)」で登録したロールパッケージで定義されている多段変数内で繰返配列定義されているメンバー変数の配列の最大繰返数の更新が行えます。変更したいメンバー変数の更新ボタンをクリックし最大繰返数を更新します。

図 5.3-15 サブメニュー画面 (多段変数最大繰返数管理)

- (2) 「一覧」-「更新」ボタンより最大繰返数の更新を行います。(※登録ボタンではありません)

項番	変数名	メンバー変数名 (繰返し有)	最大繰返数 *	備考	最終更新日時	最終更新者
1	VAR_users	-	2		自動入力	自動入力

図 5.3-16 登録画面 (多段変数最大繰返数管理)

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-10 登録画面項目一覧(多段変数最大繰返数管理)

項目	説明	入力 必須	入力形式	制約事項
最大繰返数	配列の最大繰返数を 1～99,999,999 の範囲で入力します。	○	手動入力	入力値 1～99,999,999
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

メンバー変数名の表示は各階層の変数を「.」でスコープします。また、1 階層目が繰返配列の場合はメンバー変数名が「-」と表示されます。

e.g.)

変数定義	メンバー変数の表示	最大繰返数の初期値
VAR_users:		
- name: alice	-	1
authorized:		
- /tmp/alice/onekey.pub		
- craete_users:	craete_users	2
Name: root		
password: xxxxxxxxx		
- craete_users:		
Name: mysql		
password: xxxxxxxxx		

内部の処理でロールパッケージ内に定義している多段変数繰返配列で定義されているメンバー変数の繰返数を初期登録します。初期登録後、多段変数最大繰返数管理で繰返数を更新することが出来ます。

なお、初期登録および繰返数の更新はリアルタイムではないので、「[5.3.11 代入値自動登録設定](#)」や「[5.3.9 代入値管理](#)」で変数が扱えるまでに時間がかかる<sup>※1</sup>場合があります。

※1 初期登録および繰返数の更新タイミングは「[7.2 メンテナンス方法について](#)」に記載していますので、そちらをご参照ください

### 5.3.9 代入値自動登録設定

- (1) メニュー作成機能で作成したパラメータシートと、Movement の変数を紐付けます。登録した情報は内部の処理により代入値管理と作業対象ホストに反映されます。

[6.8 BackYard コンテンツ \(2\) 代入自動値登録設定](#)に反映ルールを記載しています。

The screenshot shows the '代入値自動登録設定' sub-menu in the Exastro IT Automation Ansible-Legacy interface. The main content area displays a table for parameter sheets with columns for '項目' (Item), '登録方式' (Registration Method), and '項目' (Item). The table is currently empty. Below the table, there are buttons for 'フィルタ' (Filter) and 'フィルタクリア' (Clear Filter). The bottom section contains buttons for '一覧/更新' (List/Update), '登録' (Register), '全件ダウンロードとファイルアップロード編集' (Download all and edit file upload), and '変更履歴' (Change history).

図 5.3-17 サブメニュー画面(代入値自動登録設定)

※画面は Ansible-Legacy Role のものです

- (2) 「登録」-「登録開始」ボタンより代入値自動登録設定を行います。

The screenshot shows the '登録画面' (Registration Screen) for '代入値自動登録設定'. The table has columns for '項目' (Item), '登録方式' (Registration Method), 'Movement', 'Key変数' (Key Variable), 'Value変数' (Value Variable), and 'NULL連携' (NULL Linkage). The table is currently empty.

図 5.3-18 登録画面(代入値自動登録設定)

表 5.3-11 対応カラム一覧(代入値自動登録設定)

カラム		Legacy	Legacy Role	Pioneer
メニューグループ:メニュー		○	○	○
項目		○	○	○
登録方式		○	○	○
Movement		○	○	○
Key 情報	変数名	○	○	○
	メンバー変数名	—	▲	—
	代入順序	△	△	△
Value 情報	変数名	○	○	○
	メンバー変数名	—	▲	—
	代入順序	△	△	△
NULL 連携		●	●	●

○:必須

●:任意

△:選択した変数が複数具体値設定可能な変数の場合のみ必須

▲:選択した変数が多段変数の場合のみ必須

—:非表示

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-12 登録画面項目一覧(代入値自動登録設定)

カラム	説明	入力 必須	入力形式	制約事項
メニューグループ: メニュー	メニュー作成機能で作成したパラメータシートが表示 されます。 該当のパラメータシートを選択します。	○	リスト選択	-
項目	選択したパラメータシートの項目が表示されます。 対象の項目を選択します。	○	リスト選択	-
登録方式	Value 型:項目の設定値を紐付けた変数の具体値と する場合に選択します。 Key 型:項目の名称を紐付けた変数の具体値とする 場合に選択します。 項目の設定値が空白の場合は紐付け対象外となり ます。 Key-Value 型:項目の名称(Key)と設定値(Value)を 紐付けた変数の具体値とする場合に選択します。	○	リスト選択	-
Movement	Movement 一覧で登録した Movement が表示され ます。Movement を選択します。	○	リスト選択	-

カラム		説明	入力 必須	入力形式	制約事項
Key 情報	変数名	Movement 詳細で登録した資材で使用している変数が表示されます。 Key 型で具体値に紐付けたい変数を選択します。	○ または ／	リスト選択	登録方式で Key 型 または Key-Value 型を 選択した場合は必須
	メンバー変 数名	変数名で多段変数を選択した場合に多段変数の メンバー変数が表示されます。 メンバー変数を選択します。	○ または ／	リスト選択	
	代入順序	複数具体値が設定できる変数の場合のみ必須入力 になります。 具体値の代入順序(1～)を入力します。入力値に従 い昇順で代入されます。具体値が複数ない場合でも 代入順序(1～)を入力します。	○ または ／	手動入力	ブランク または、 正の整数
Value 情報	変数名	Movement 詳細で登録した資材で使用している変数が表示されます。 Value 型で具体値に紐付けたい変数を選択します。	○ または ／	リスト選択	登録方式で Value 型 または Key-Value 型を 選択した場合は必須
	メンバー変 数名	変数名で多段変数を選択した場合に多段変数の メンバー変数が表示されます。 メンバー変数を選択します。	○ または ／	リス ト 選 択	-
	代入順序	複数具体値が設定できる変数の場合のみ必須入力 になります。 具体値の代入順序(1～)を入力します。入力値に従 い昇順で代入されます。具体値が複数ない場合でも 代入順序(1～)を入力します。	○ または ／	手動入力	ブランク または、 正の整数
NULL 連携		パラメータシート在具体値が NULL(空白)の場合 に、代入値管理への登録を NULL(空白)の値で行う か設定します。 ・「有効」の場合、パラメータシートの値がどのよう な値でも代入値管理への登録が行われます。 ・「無効」の場合、パラメータシートに値が入って いる場合のみ代入値管理への登録が行われます。 ・空白の場合、Ansible インターフェース情報の「NU LL 連携」の値が適用されます。	-	リスト選択	-
備考		自由記述欄です。	-	手動入力	最大長 4000 バイト

※ メンバー変数名の表記内容については「5.3.11 代入値管理」に記載していますので、そちらをご参照くだ  
さい。

### 5.3.10 作業対象ホスト

(1) [作業対象ホスト]では、オペレーションに関連付く Movement とホストの登録／更新／廃止を行います。

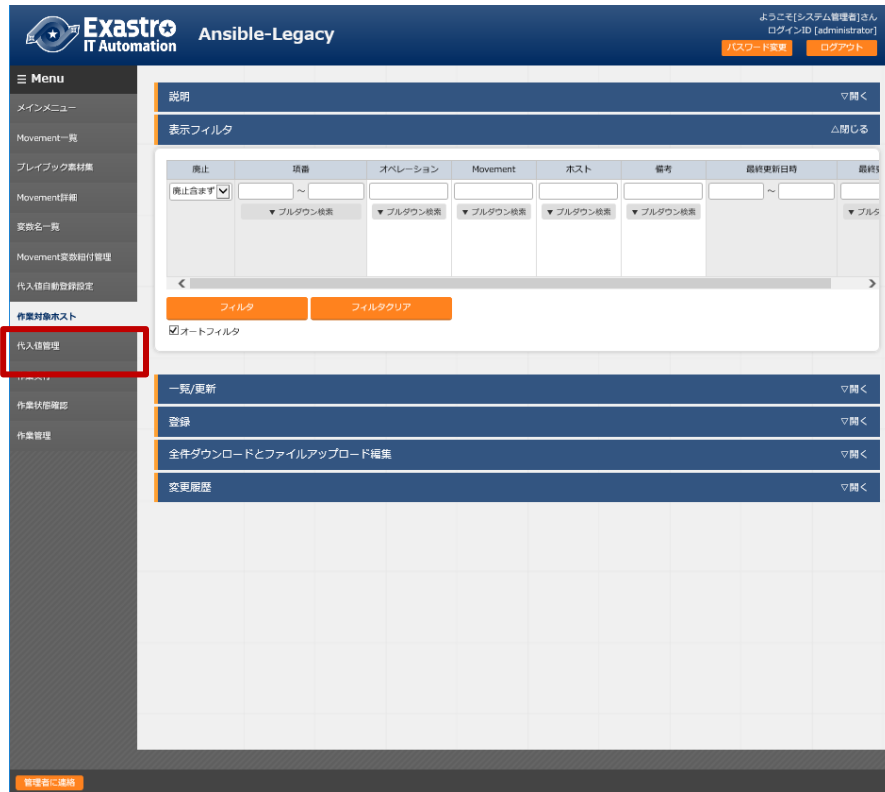


図 5.3-19 サブメニュー画面(作業対象ホスト)

(2) 「登録」-「登録開始」ボタンより、作業対象ホスト登録を行います。

図 5.3-20 登録画面(作業対象ホスト)

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-13 登録画面項目一覧(作業対象ホスト)

項目	説明	入力必須	入力形式	制約事項
オペレーション	投入オペレーション一覧に登録されているオペレーションが表示されます。オペレーションを選択します。	○	リスト選択	-
Movement	Movement 一覧に登録されている Movement が表示されます。オペレーションに紐付ける Movement を選択します。	○	リスト選択	-
ホスト	機器一覧に登録されているホスト名が表示されます。オペレーションに紐付けるホストを選択します。	○	リスト選択	-
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

### 5.3.11 代入値管理

- (1) 変数への代入値の登録／更新／廃止を行います。
- オペレーションごとに、対象の Movement で利用される Playbook やテンプレートファイル内の変数「VAR\_」に代入する具体値をメンテナンス(閲覧/登録/更新/廃止)できます。
- また、読替表の定義により「VAR\_」以外の変数「LCA\_」に対して代入する具体値をメンテナンスできます。詳しくは「[6.7 読替表の記述](#)」を参照してください。
- 登録した変数の情報は作業実行時にホスト変数ファイル(host\_vars/配下)に出力されます。



図 5.3-21 サブメニュー画面(代入値管理)

※画面は Ansible-Legacy Role のものです。

- (2) 「登録」-「登録開始」ボタンより代入値管理を行います。

項目	オペレーション*	Movement	ホスト	変数名	メンバー変数名	具体値	代入順序	デフォルト値
自動入力	<input type="text"/>	オペレーションを選択して下さい	オペレーションを選択して下さい	Movementを選択して下さい	変数名を選択して下さい	<input type="text"/>		

図 5.3-22 登録画面(代入値管理)

代入値管理の変数名は、アップロードされた Playbook や代入値自動登録設定で登録された情報から反映されます。

※ 反映のタイミングは「[7.2 メンテナンス方法について](#)」に記載していますので、そちらをご参照ください。

(3) 登録画面の項目一覧は以下のとおりです。

表 5.3-14 対応カラム一覧(代入値管理)

カラム	Legacy	Legacy Role	Pioneer
オペレーション	○	○	○
Movement	○	○	○
ホスト	○	○	○
変数名	○	○	○
メンバー変数名	—	▲	—
代入順序	△	△	△
デフォルト値(表示のみ)	—	○	—

○:必須

△:選択した変数が複数具体値設定可能な変数の場合のみ必須

▲:選択した変数が多段変数の場合のみ必須

—:非表示



**表 5.3-15 登録画面項目一覧(代入値管理)**

項目	説明	入力必須	入力形式	制約事項
オペレーション	作業対象ホストに登録されているオペレーションが表示されます。オペレーションを選択します。	○	リスト選択	-
Movement	作業対象ホストに登録されているデータの中から、選択されたオペレーションに紐づく Movement が表示されます。Movement を選択します。	○	リスト選択	-
ホスト	作業対象ホストに登録されているデータで選択されたオペレーションと Movement に紐づくホストが表示されます。ホストを選択します。	○	リスト選択	-
変数名	Movement 詳細にて登録されている資材の中から、選択された Movement にアタッチしている変数名が表示されます。変数を選択します。	○	リスト選択	-
メンバー変数名	変数名で多段変数を選択した場合に多段変数のメンバー変数が表示されます。メンバー変数を選択します。	○ または ／	リスト選択	-
具体値	オペレーション/Movement/ホストで使用する変数の具体値を入力します。複数行の具体値も入力できますが、Pioneerで複数行の具体値を設定すると、作業実行時にエラーとなります。 具体値にファイル埋込変数「CPF_」とテンプレート埋込変数「TPF_」が入力出来ます。変数を記述する場合、Playbookに変数を記述する場合と同様、変数名を{{ }}で囲みます。 入力例) 具体値に TPF_sample を入力する場合 '{{△TPF_sample△}}'      △:半角スペース △:推奨	○	手動入力	最大長 8192 バイト
代入順序	複数具体値が設定できる変数の場合のみ必須入力になります。 具体値の代入順序(1～)を入力します。入力値に従い昇順で代入されます。具体値が複数ない場合でも代入順序(1～)を入力します。	○ または ／	手動入力	ブランク または、 正の整数
デフォルト値	変数名およびメンバー変数名で選択されている変数のデフォルト変数定義ファイル(defaults->main.yml)に設定されている具体値を表示します。 詳しくは「 <a href="#">6.4 ITAreadme (Ansible-Legacy Role のみ) の記述</a> 」を参照してください。 具体値が 8 進数文字列(0755)や 16 進数文字列(0x1ED)の場合、10 進数(493)で表示されます。 具体値が「true」「Yes」「Y」「y」の場合は空白が表示されます。 具体値が「false」「No」「N」「n」の場合は False が表示されます。	-	表示のみ	-
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

### 【メンバー変数名の表示内容】

多段変数の場合にのみメンバー変数の選択が必要になります。メンバー変数に表示される変数は具体値を必要とする変数のみです。

メンバー変数名の表示は各階層の変数を「.」でスコープします。繰返配列の場合は「[ ]」で繰返位置(0～)をスコープします。繰返し配列の数は「[5.3.8 多段変数最大繰返数](#)」で設定を行います。

e.g.)

変数定義	メンバー変数の表示
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:	[1].mysql.password
password: mysql-password	[1].mysql.hosts
hosts:	※mysql は階層を示す変数なのでメンバー変数には表示されません。
- "127.0.0.1"	
- "localhost"	

代入値自動登録設定で登録した情報は、内部の処理で代入値管理と作業対象ホストへ反映されます。

※反映のタイミングは「[7.2 メンテナンス方法について](#)」に記載していますので、そちらをご参照ください。

## ① 代入順序の入力

Ansible-Legacy では、代入順序が未入力の場合は、通常変数として扱います。

代入順序が入力されている場合は、複数具体値変数として扱います。複数具体値変数の場合は複数の具体値が必要ない場合(具体値が 1 個でよい)でも代入順序は入力してください。

Ansible-Legacy Role では、変数名またはメンバー変数名を選択することで、複数具体値変数の場合のみ代入順序が入力可能となります。複数具体値変数の場合に入力してください。

Ansible-Pioneer では、代入順序が未入力の場合は、通常変数として扱います。

代入順序が入力されている場合は、複数具体値変数として扱います。複数具体値変数の場合は、複数の具体値が必要ない場合(具体値が 1 個でよい)でも代入順序を入力してください。

各モードとも、特定の複数具体値変数に対して代入順序が連続していなくても問題ありません。

e.g.)

代入値管理の登録

ホスト	変数	具体値	代入順序
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

HOST\_A のホスト変数ファイルへの出力内容

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

## ② ホスト変数ファイルへの出力

代入値管理で登録した変数の具体値はホスト変数ファイルへ出力されます。

Ansible-Legacy と Ansible-Pioneer では、作業実行時に Playbook または対話ファイルで使用している変数の具体値が代入値管理に登録されていないと作業実行が想定外エラーとなります。

Ansible-Legacy Role では、代入値管理で具体値を登録した変数のみが作業実行時にホスト変数ファイルへ出力されます。多段変数も同様に具体値を登録しているメンバー変数のみとなります。

e.g.)

変数定義

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

代入値管理の登録

ホスト	変数	メンバー変数	具体値	代入順序
HOST_A	VAR_users:	[0].name	value1	
HOST_A	VAR_users	[1].authorized	value2	

HOST\_A のホスト変数ファイルへの出力内容

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

### ③ デフォルト値チェックオプション

複数ロール間でデフォルト値が一致していない変数に対して具体値の登録した場合に、警告メッセージを表示して登録させないパラメータを「ITA 管理コンソール システム設定」で設定することが出来ます。このパラメータはデフォルトでは未登録です。必要に応じて登録して下さい。

システム設定に登録する内容は以下の通りです。尚、システム設定については「利用手順マニュアル\_管理コンソール」を参照してください。

表 5.3-16 システム設定登録内容

項目	入力値	入力必須
識別 ID	ANSIBLE_DEF_VAL_CHK	○
項目名	任意の文字列	-
設定値	1: パラメータ有効 1 以外またはレコード未登録 : パラメータ無効	○
備考	任意の文字列	

### 5.3.12 作業状態確認

- (1) 作業の実行状態を監視します。

The screenshot shows the 'Exastro IT Automation Ansible-Legacy' web interface. On the left, a sidebar menu has '作業状態確認' (Job Status Confirmation) highlighted with a red box. The main content area displays job details in a table format.

項目		値
作業No.		9
実行種別		通常
ステータス		完了
実行エンジン		Ansible Tower
virtualenv		
呼出元Symphony		
呼出元Conductor		
実行ユーザ		システム管理者
Movement	ID	1
	名称	legacy
	遅延タイム(分)	
Ansible利用情報	ホスト指定形式	ホスト名
	WinRM接続	
オペレーション	No.	1
	名称	ope
	ID	1
作業対象ホスト		確認
代入値		確認
入力データ	投入データ	<a href="#">InputData_0000000009.zip</a>
出力データ	結果データ	<a href="#">ResultData_0000000009.zip</a>
作業状況	予約日時	
	開始日時	2020/08/12 16:27:04
	終了日時	2020/08/12 16:28:18

図 5.3-23 サブメニュー画面(作業状態確認)

## ① 実行状態表示

実行状況に即し、「ステータス」が表示されます。

また、実行ログ、エラーログに実行状況の詳細が表示されます。

「実行種別」には、ドライランの場合は「ドライラン」、それ以外は「通常」が表示されます。

ステータスが想定外エラーで終了した場合、Web コンテンツの登録不備が原因であれば、エラーログにメッセージが表示されます。

また、「5.2.1 インターフェース情報」の登録不備等で、Ansible RestAPI との通信に失敗した場合にはエラーログにメッセージが表示されません。この場合は、アプリケーションログにエラー情報が記録されます。必要に応じてアプリケーションログを確認ください。

Symphony から実行した場合に、「呼出元 Symphony」には、どの Symphony から実行されたかを表示します。

Conductor から実行した場合に、「呼出元 Conductor」には、どの Conductor から実行されたかを表示します。

Ansible-Legacy,Pioneer,LegacyRole,ドライバから直接実行した場合は空欄になります。

「実行ユーザ」には、作業実行メニューより「実行」ボタンまたは「ドライラン」ボタンを押下した際のログインユーザが表示されます。

## ② 作業対象ホスト確認

「確認」ボタンで「5.3.10 作業対象ホスト」が表示され、作業対象のオペレーションと Movement に絞り込んだホストが表示されます。

## ③ 代入値確認

「確認」ボタンで「5.3.11 代入値管理」が表示され、作業対象のオペレーションと Movement に絞り込んだ代入値が表示されます。

## ④ 緊急停止/予約取り消し

「緊急停止」ボタンで構築作業を停止させることができます。

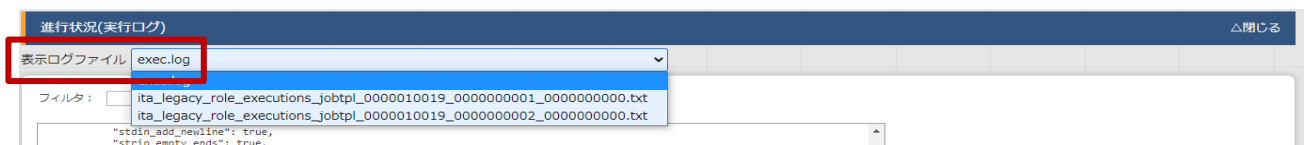
また、実行前の「予約実行」の作業の場合は、「予約取消」ボタンが表示されます。「予約取消」ボタンで予約実行が取り消せます。

## ⑤ 実行ログ表示

AnsibleTower で実行した場合、構築対象機器の機器一覧のユーザー・パスワード・インスタンスグループなどの項目値でグループ化された構築対象機器の単位で Playbook が実行され、ansible の実行ログが分割されます。

さらに、Movement 一覧のオプションパラメータでジョブスライス数を指定することによりグループ化された構築対象機器をさらにジョブスライス数で分割し playbook が実行され、ansible の実行ログも分割されます。

実行ログが分割された場合、表示ログファイルのプルダウンが表示され、表示したいログファイルを選択する事ができます。



表示ログファイルのプルダウンに表示されるログファイル名は以下の 2 種類があります。

exec.log: 全ての実行ログをまとめたログファイルです。

exec.log 以外: 分割された実行ログファイルです。ファイル命名規則は以下になります。

Ita\_<mode 名>\_executions\_jobtpl\_<作業番号>\_<グループ番号>\_<通番>

表 5.3-17 分割された実行ログファイルの命名要素

要素	内容
mode 名	実行したモード名 legacy/pioneer/legacy_role

作業番号	作業管理メニューの作業実行 No
グループ番号	構築対象機器の機器一覧のユーザー・パスワード・インスタンスグループなどの項目値でグループ化した 1 からの通番です。
通番	ジョブスライス数の設定によりグループ内を分割した 1 からの通番です。 0 の場合はジョブスライス数による分割がなかったことを表します。

⑥ ログ検索

実行ログ、エラーログは、フィルタリングができます。各ログのフィルタのテキストボックスに検索したい文字列を入力し、「該当行のみ表示」のチェックボックスをチェックすることで該当する行だけが表示されます。

実行ログ、エラーログのリフレッシュ表示間隔と最大表示行数を、「5.2.1 インターフェース情報」の「状態監視周期(単位ミリ秒)」と「進行状態表示行数」で設定できます。

⑦ 投入データ

実行した Playbook などをダウンロードすることができます。

投入データの構成は「8.1 Ansible 実行時に使用される投入データと ITA メニューの紐づけ」を参照して下さい。

⑧ 結果データ

実行ログ、エラーログなどをダウンロードすることができます。

### 5.3.13 作業管理

- (1) 作業の履歴を閲覧できます。

条件を指定し「フィルタ」ボタンをクリックすると、作業一覧テーブルとグラフを表示します。

それぞれのグラフにマウスカースルを合わせると、グラフのダウンロードボタンが右上に表示されます。表示されたグラフのダウンロードボタンをクリックすることで、グラフのダウンロードが可能です。

「作業状態確認」ボタンで、「5.3.12 作業状態確認」に遷移し、実行状態の詳細を見ることができます。

The screenshot shows the 'Exastro IT Automation Ansible-Legacy' web interface. On the left sidebar, the '作業管理' (Job Management) menu item is highlighted with a red box. The main content area is titled '表示フィルタ' (Display Filter) and contains a table of job execution results. The table has columns for '作業ID' (Job ID), '作業名' (Job Name), '実行種別' (Execution Type), 'ステータス' (Status), '実行エンジン' (Execution Engine), 'virtualenv', '実行元Sympathy', '実行元Conductor', '実行ユーザ', and '実行日時' (Execution Date). The table lists several jobs, all of which are in a '成功' (Success) state. Below the table, there are buttons for 'フィルタ' (Filter) and 'フィルタをリセット' (Reset Filter). At the bottom of the page, there is a 'ログアウト' (Logout) button.

図 5.3-24 サブメニュー画面(作業管理)

### 5.3.14 作業実行

- (1) 作業の実行を指示します。Movement 一覧、オペレーション一覧からそれぞれラジオボタンで選択し、実行ボタンを押すと、「5.3.12 作業状態確認」に遷移し、実行されます。

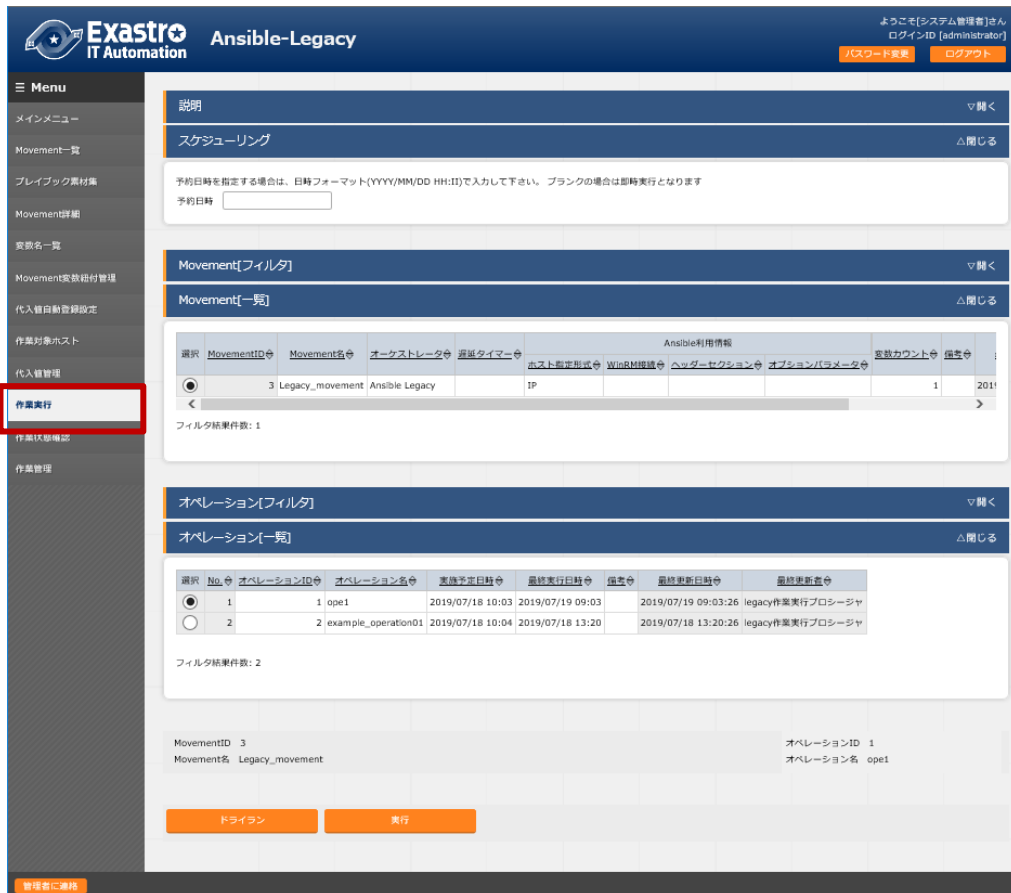


図 5.3-25 サブメニュー画面(作業実行)

※画面は Ansible-Legacy のものです。

#### ① ドライラン

「ドライラン」ボタンをクリックすると、実際に対象機器に対して構築作業をせず、ドライランを行うことができます。ドライランを行った場合の、モード毎の動作は以下のとおりです。

Driver	動作
Ansible-Legacy	Ansible-Playbook コマンドの—check パラメータを指定し Playbook を実行します。
Ansible-Legacy Role	Ansible-Playbook コマンドの—check パラメータを指定し role を実行します。
Ansible-Pioneer	対象機器への接続チェックのみを行います。

#### ② 予約日時の指定

「予約日時」を入力することで、実行を予約することができます。

「予約日時」には、未来の日時のみ登録可能です。



## 6 構築コード記述方法

### 6.1 Playbook (Ansible-Legacy) の記述

5.3.3. [プレイブック素材集 \(Ansible-Legacy のみ\)](#) でアップロードされた Playbook は、ITA で生成するマスタ Playbook より include 形式で実行されます。

ITA で作成するマスタ Playbook はヘッダーセクションと tasks セクションで構成されます。

#### (1) ヘッダーセクション

アップロードする Playbook にはヘッダーセクションは含む必要はありません。

ヘッダーセクションは、デフォルト値が決まっていますが、「5.3.2. [Movement 一覧](#)」のヘッダーセクションで変更することが出来ます。

ヘッダーセクションのデフォルト値

・Ansible engine の場合

- hosts: all

remote\_user: "{{ \_\_loginuser\_\_ }}"

gather\_facts: no

become: yes

・AnsibleTower の場合

- hosts: all

gather\_facts: no

become: yes

#### (2) tasks セクション

アップロードされた Playbook は、ITA で生成するマスタ Playbook より include 形式で実行されます。

Playbook 基本書式については Ansible の公式マニュアルを参照してください。

Playbook 内のインデントは 2 倍数で調整してください。

文字コードは、UTF-8 の BOM なしで作成して下さい。

e.g.)

-△name: コメント

△△template:

△△△△src: "{{ item.src }}"

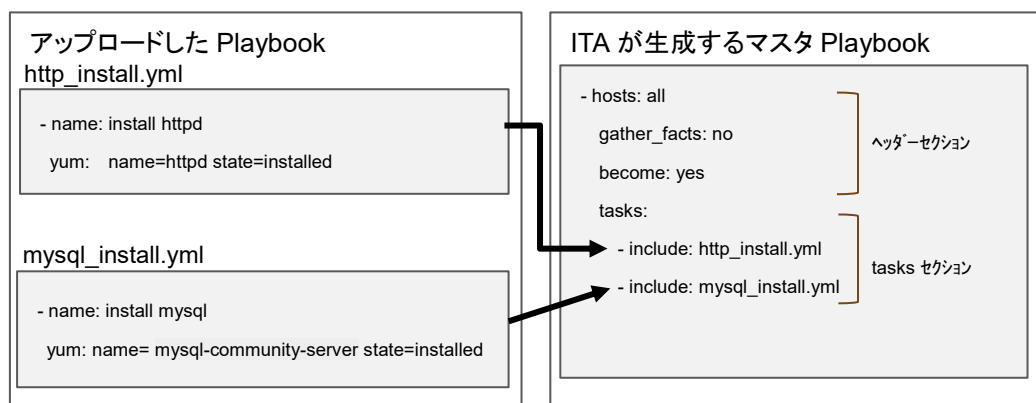
△△△△dest: "{{ item.dest }}"

△△△△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) }}"

アップロードした Playbook は、「5.3.7. [Movement 詳細](#)」のインクルード順序に従い include します。



## 6.2 対話ファイル（Ansible-Pioneer）の記述

Ansible-Pioneer では、ITA 独自モジュールを Ansible に組み込んでいます。  
対話ファイルは ITA 独自書式となります。  
文字コードは、UTF-8 の BOM なで作成して下さい。

### (1) 対話ファイルの構成

対話ファイルは 2 種類のセクションにより構成されます。

セクション名	用途
Conf	timeout パラメータによりタイムアウト値を指定します。 タイムアウト値:1~3600(単位:秒)
exec_list	4 種類の対話コマンドにより作業対象ホストの構築を行います。

対話ファイルの先頭に timeout パラメータを記述。以降に対話コマンドを記述します。  
コメントは Ansible の基本書式と同様の記述が出来ます。

```
e.g.)
# コメント
conf:
△△timeout: 10
exec_list:
※△:半角スペース
timeout:の記述の前に半角スペース 2 文字を付与してください。
```

### (2) 対話コマンド

対話コマンドは以下の 5 種類があります。

モジュール	用途
exec	作業対象ホストにコマンドを投入します。
Expect	作業対象ホストが標準出力に出力する内容より、期待する文字列(プロンプト)の出力を待ち合せます。
State	作業対象ホストにコマンドを投入し、標準出力にプロンプトを出力するまでの標準出力の内容を外部 Shell で解析し結果判定をします。
Command	作業対象ホストにコマンドを投入する前後において、繰り返しや条件分岐を行うことができます。
localaction	Ansible/Ansible Tower サーバ上でコマンドを実行します。

#### ① expect モジュール

作業対象ホストが標準出力に出力する内容より、期待する文字列(プロンプト)の出力を待ち合せます。  
期待する文字列を**正規表記**で記述できます。  
期待する文字列を受取ると次へ進みます。また、timeout パラメータで指定された時間内に受取れない場合は対話ファイルを異常終了します。

```
e.g.) telnet 接続でパスワード入力のプロンプトを待ち合せます。
△△-△expect:△'Password'
※△:半角スペース
- expect: の記述の前に半角スペース 2 文字を付与してください。
待ち合わせる文字列をコーテーションで囲むことを推奨します。
```

## ② exec モジュール

作業対象ホストにコマンドを投入します。

exec モジュールと expect モジュールは対で使用します。

e.g.) telnet 接続でパスワード入力のプロンプトを待ち合せてパスワードを投入します。

△△-△expect:△'Password'

△△△△exec:△itapassword

※△:半角スペース

- exec: の記述の前に半角スペース 4 文字を付与してください。

## ③ state モジュール

作業対象ホストにコマンドを投入し、標準出力にプロンプトを出力するまでの標準出力の内容を外部 Shell で解析し結果判定をします。

state モジュールの書式

パラメータ	必須/ 任意	説明
△△-△state:△xxx	必須	投入するコマンドを指定します。
△△△△prompt:△xxx	必須	待受けプロンプトを指定します。 <b>正規表記</b> で記述できます。
△△△△shell:△xxx	任意	作成した shell で結果を確認する場合に、shell ファイル名を指定します。 作成した shell の exit コードが 0 の場合は正常、他は異常と判定します。 デフォルトの shell で結果を確認する場合、本パラメータは不要となります。デフォルトの shell は parameter(-)で指定された文字列で標準出力の内容を grep します。マッチする行が 1 行でもあれば正常とし、マッチする行がなければ異常と判定します。また、parameter を指定しなかった場合、異常と判定されます。コマンドの結果(標準出力)を stdout_file で指定したファイルに退避したい目的で使用する場合、ignore_errors で「yes」を指定して下さい。
△△△△parameter: △△△△△△-△xxx △△△△△△-△xxx	任意	投入するコマンドの結果(標準出力)を検索する文字列を指定します。 shell を指定している場合、作成した shell の実行時パラメータとなります。複数ある場合は検索文字列を列挙します。
△△△△stdout_file:△xxx	任意	投入するコマンドの結果(標準出力)を退避するファイルです。
△△△△success_exit: △ xxx	任意	検索結果が正常の場合で対話ファイルを正常終了する場合に「yes」を指定します。「no」の場合は正常の場合は次に進みます。デフォルトは「no」。
△△△△ignore_errors: △ xxx	任意	検索結果が異常でも次に進む場合に「yes」を指定します。「no」の場合は、異常の場合に対話ファイルを異常終了とします。デフォルトは「no」。
※△:半角スペース		

#### Exp1)

hosts ファイルを cat し、表示結果を parameter 値で grep している。139.0.0.1、lalhost を含む行あれば正常と判定し次に進みます。行がなければ異常と判定し対話ファイルを異常終了します。

exec\_list:

- state: cat /etc/hosts
- prompt: root@{{ \_\_loginhostname\_\_ }}
- parameter:
  - 139.0.0.1
  - lalhost

#### Exp2)

hosts ファイルを cat し、表示結果を parameter 値で grep している。139.0.0.1、lalhost を含む行あれば正常と判定しますが success\_exit: yes の設定により対話ファイルを正常終了します。行がなければ異常と判定し対話ファイルを異常終了します。

exec\_list:

- state: cat /etc/hosts
- prompt: root@{{ \_\_loginhostname\_\_ }}
- parameter:
  - 139.0.0.1
  - lalhost

success\_exit: yes

#### Exp3)

hosts ファイルを cat し、表示結果を parameter 値で grep している。139.0.0.1、lalhost を含む行あれば正常と判定し次に進みます。行がなければ異常と判定しますが ignore\_errors: yes の設定により次に進みます。

exec\_list:

- state: cat /etc/hosts
- prompt: root@{{ \_\_loginhostname\_\_ }}
- parameter:
  - 139.0.0.1
  - lalhost

ignore\_errors: yes

Exp4)

hosts ファイルを cat し、ユーザー作成の shell で表示結果を parameter 値で grep している。139.0.0.1、lalhost を含む行あれば正常と判定し次に進みます。行がなければ異常と判定し対話ファイルを異常終了します。

exec\_list:

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

ユーザー作成 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
/bin/rm -rf ${STDOUT} ${STDERR} >/dev/null 2&>1
exit ${EXIT_CODE}
```

Exp5)

hosts ファイルを cat し、表示結果を stdout\_file で指定したファイルに保存し次に進みます。デフォルトの shell は parameter の設定がないと異常と判定します。次に進める為に ignore\_errors: yes を設定します。

exec\_list:

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

④ command モジュール

作業対象ホストにコマンドを投入する前後において、繰り返しや条件分岐を行うことができます。

command モジュールの書式

パラメータ	必須/ 任意	説明
△△-△command:△xxx	必須	投入するコマンドを指定します。
△△△△prompt:△xxx	必須	待受けプロンプトを指定します。 <b>正規表記</b> で記述できます。
△△△△timeout:△xxx	任意	コマンドを送ってからのプロンプト待ちタイマを指定します。 省略されている場合は、conf->timeout を使用します。
△△△△register:△xxx	任意	コマンドを送信後に標準出力の情報を退避する任意の文字列です。 with_items でループしている場合は、最後のコマンド送信後の標準出力の情報を退避。この変数で条件判定が出来ます。(使用できるのは条件判定のみ) ただし、変数名ごとに標準出力の情報を退避することはできません。 前の情報を上書きします。
△△△△with_items: △△△△△-△xxx △△△△△-△xxx	任意	with_items に設定する変数は複数具体値変数です。 with_items に設定する変数の具体値数は同じでなくても良いです。 同じでない場合は、各変数の具体値数の最大値数でループします。 具体値が不足している変数の具体値は空として扱います。 ただし、prompt または timeout の場合は上記に当てはまりません。 prompt、timeout で具体値が不足していると、エラーになります。 各変数のスコープは item.X(X は 0 から 99)とします。 変数(item.X)の適用範囲は register/when 以外です。 prompt、timeout で with_items を利用する場合の変数名は下記の通りにしてください。 prompt: {{△VAR_prompt_XXX△}} timeout: {{△VAR_timeout_XXX△}} (△は半角スペース。XXX は任意の半角英数字とアンダースコア)
△△△△when: △△△△△-△xxx △△△△△-△xxx	任意	command 実行前の条件判定です。 条件にマッチして command 実行します。 条件にマッチしていなければ次の command 行に移ります。 条件式 変数定義判定 VAR_xx is define      変数が定義されている      true VAR_xx is undefine      変数が未定義      true ※define/undefine は ITA の変数(VAR_xx)のみ指定可能 変数具体値判定 VAR_xx/register 変数 比較演算子 文字列 VAR_xx/register 変数 比較演算子 VAR_xx VAR_xx/register 変数 match(正規表記文字列/VAR_xx) VAR_xx/register 変数 no match(正規表記文字列/VAR_xx) ※比較演算子は「=」、「!=」、「>」、「>=」、「<」、「<=」 ※比較演算子の「>」、「>=」、「<」、「<=」は数値を想定しています。 ※文字列と正規表記文字列はシングル・ダブルコーテーションで囲む必要はありません。 VAR_status match(active)      VAR_status == active and/or による複合条件

パラメータ	必須/ 任意	説明
		<p>or 条件を行いたい場合、判定条件の間に OR を記述します。</p> <ul style="list-style-type: none"> <li>- VAR_status == 1 OR VAR_status == 2</li> </ul> <p>and 条件を行いたい場合、複数行に分けて記述すると and 条件になります。</p> <ul style="list-style-type: none"> <li>- VAR_status == 1 OR VAR_status == 2</li> <li>- VAR_sub_status == 1</li> </ul>
△△△△exec_when: △△△△△△-△xxx △△△△△△-△xxx	任意	<p>ループ毎の条件判定です。(continue 条件)</p> <p>with_items が記述されている場合に条件判定を行います。</p> <p>条件にマッチしていれば該当ループの command を実行します。</p> <p>マッチしていなければ次のループへ移ります。</p> <p>条件式</p> <p>変数定義判定</p> <p>VAR_xx is define      変数が定義されている      true</p> <p>VAR_xx is undefine      変数が未定義      true</p> <p>※define/undefine は ITA の変数(VAR_xx)のみ指定可能</p> <p>変数具体値判定</p> <p>VAR_xx/register 変数 比較演算子 文字列</p> <p>VAR_xx/register 変数 比較演算子 VAR_xx</p> <p>VAR_xx/register 変数 match(正規表記文字列/VAR_xx)</p> <p>VAR_xx/register 変数 no match(正規表記文字列/VAR_xx)</p> <p>※比較演算子は「==」、「!=」、「&gt;」、「&gt;=」、「&lt;」、「&lt;=」</p> <p>※比較演算子の「&gt;」、「&gt;=」、「&lt;」、「&lt;=」は数値を想定しています。</p> <p>※文字列と正規表記文字列はシングル・ダブルコーテーションで囲む必要はありません。</p> <p>VAR_status match(active)      VAR_status == active</p> <p>and/or による複合条件</p> <p>when:と同様の記述を行います。</p>
△△△△failed_when: △△△△△△-△xxx △△△△△△-△xxx  ※△:半角スペース	任意	<p>command 実行後(ループ毎)の stdout の内容に対する条件判定です。</p> <p>with_items が記述されている場合に条件判定を行います。</p> <p>条件にマッチしていれば正常とします。</p> <p>マッチしていなければ異常とし、対話ファイルを異常終了させます。</p> <p>条件式</p> <p>変数具体値判定</p> <p>stdout 比較演算子 文字列</p> <p>stdout 比較演算子 VAR_xx</p> <p>stdout match(正規表記文字列/VAR_xx)</p> <p>stdout no match(正規表記文字列/VAR_xx)</p> <p>※比較演算子は「==」、「!=」、「&gt;」、「&gt;=」、「&lt;」、「&lt;=」</p> <p>※比較演算子の「&gt;」、「&gt;=」、「&lt;」、「&lt;=」は数値を想定しています。</p> <p>※文字列と正規表記文字列はシングル・ダブルコーテーションで囲む必要はありません。</p> <p>and/or による複合条件</p> <p>when:と同様の記述を行います。</p>

```
Exp1)
conf:
    timeout: 30

exec_list:
    # プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
    # パスワードが必要な場合
    - expect: 'password:'
      exec:   '{{ __loginpassword__ }}'

    # VAR_hosts_make という ITA 変数がホスト変数ファイルに記載されている場合、
    # hosts ファイルを cat します。記載されていない場合は、スキップします。
    - command: cat /etc/hosts
      prompt: root@{{ __loginhostname__ }}
      when:
        - VAR_hosts_make is define
```



```

Exp2)
conf:
    timeout: 30

exec_list:
# プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
# パスワードが必要な場合
- expect: 'password:'
  exec:   '{{ __loginpassword__ }}'

# VAR_hosts_make という ITA 変数がホスト変数ファイルに記載されている場合、
# hosts ファイルを cat します。記載されていない場合は、スキップします。
# cat により、標準出力された hosts ファイルの内容を result_stdout に退避します。
- command: cat /etc/hosts
  prompt: root@{{ __loginhostname__ }}
  register: result_stdout
  when:
    - VAR_hosts_make is define

# VAR_hosts_make という ITA 変数がホスト変数ファイルに記載されている場合、
# command 実行します。記載されていない場合は、スキップします。
# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# ループ毎の条件判定として、hosts ファイルに「ip アドレス ホスト名」が該当しない場合
# command 実行します。
# hosts ファイルの最終行に echo による、「IP アドレス ホスト名」を追記します。

- 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 }}')

```

```

Exp3)
conf:
    timeout: 30

exec_list:
# プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
# パスワードが必要な場合
- expect: 'password:'
  exec: '{{ __loginpassword__ }}'

# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# 自動起動設定を実行します。
- command: 'systemctl enable {{ item.0 }}'
  prompt: 'root@{{ __loginhostname__ }}'
  with_items:
    - '{{ VAR_service_name_list }}' # item.0

# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# サービスの起動を実行します。
- command: 'systemctl start {{ item.0 }}'
  prompt: 'root@{{ __loginhostname__ }}'
  with_items:
    - '{{ VAR_service_name_list }}' # item.0

# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# サービスのステータスを標準出力します。
# 標準出力された結果の内容に、item.1 の正規表現がある場合、正となります。
# 例えば、VAR_service_status_list の具体値を running と設定し、サービスが起動している場合、
# 「Active: active (running)」の runnig が一致するので正となります。(次のループに移ります)
# そうでない場合は、異常と判断し、対話ファイルは異常終了となります。
- 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 }}')

```

```

Exp4)
conf:
    timeout: 30

exec_list:
# プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
# パスワードが必要な場合
- expect: 'password:'
  exec:   '{{ __loginpassword__ }}'

# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# command に「{{ item.0 }}」のみの記述をする場合は、ダブルクォーテーションで囲みます。
# prompt や timeout で with_items を利用する場合、具体値数に注意が必要です。
# prompt→command→prompt→command→prompt ... (以下ループ)となり、command 数+1
# 設定する必要があります。(timeout も同様)
- 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

```

```

Exp5)
conf:
    timeout: 30

exec_list:
# プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
# パスワードが必要な場合
- expect: 'password:'
  exec:   '{{ __loginpassword__ }}'

# with_items の複数具体値変数に設定されている具体値数分 command 実行します。
# 代入値管理の具体値に{{ item.X }} を設定することができます。その際は対話ファイルに記載する
# item.X より具体値に記載する item.X の数値が大きくなるようにしてください。
# 今回の例で実行する command は「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

```

変数名	具体値
VAR_command_list	systemctl status {{ item.1 }}
VAR_service_name_list	ky_pioneer_execute-workflow.service

```

Exp6)
conf:
    timeout: 30

exec_list:
    # プロンプト以外の文字列で待合せが必要な場合は、expect/exec の組合せです。
    # パスワードが必要な場合
    - expect: 'password:'
      exec:   '{{ __loginpassword__ }}'

    # and/or による複合条件の記述例です。
    # or 条件を行いたい場合、if 文を横に記述することができます。
    # and 条件を行いたい場合、複数行に分けて記述すると and 条件になります。
    # 今回、when を例にしていますが、exec_when、failed_when も同様です。
    - command: echo aaa
      prompt: 'root@{{ __loginhostname__ }}'
      when:
        - 10 == 9 OR 10 != 9 OR 10 >= 9
        - 10 > 9 OR 10 <= 9 OR 10 < 9

```

and 条件

#### ⑤ localaction モジュール

Ansible/Ansible Tower サーバ上でコマンドを実行します。

localaction モジュールの書式

パラメータ	必須/ 任意	説明
△△-△localaction:△xxx	必須	実行するコマンドを指定します。
△△△△ignore_errors:△ xxx	任意	コマンドの実行結果が異常でも次に進む場合に「yes」を指定します。 「no」の場合は、異常の場合に対話ファイルを異常終了します。 デフォルトは「no」。
※△:半角スペース		

```

Exp1)
Symphony 実行時の各 Movement で共有するディレクトリ('{{ __symphony_workflowdir__ }}')に
ホスト毎のディレクトリを設けて、hosts ファイルを退避します。
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

```

### (3) 正規表記

下記のコマンド及びパラメータに記述された文字列は正規表記で評価されます。

- expect モジュール
- state モジュールの prompt パラメータ
- command モジュールの prompt パラメータ

正規表記で記述した文字列にメタ文字「(){}.\$\*[]^<」を含む場合、メタ文字の前にエスケープ文字「¥」を挿入する必要があります。

Exp1)

以下のような文字列を待ち受ける場合、赤字がメタ文字となります。

XAMPP Developer Files [Y/n] exec\_list:

メタ文字の前にエスケープ文字「¥」を挿入する必要があります。

XAMPP Developer Files ¥[Y¥/n¥] exec\_list:

state モジュールと command モジュールは、実行したコマンドの結果(標準出力)の取り出しを行っています。取り出す上での留意事項を以下に記述します。

#### ① 実行したコマンドの結果(標準出力)とプロンプトの区切り

実行したコマンドの結果(標準出力)とプロンプトの区切りを prompt パラメータで指定された文字列で行います。実行したコマンドの結果(標準出力)の判定やファイルへの保存を行う場合は、正規表記で .\* 付の後方一致は記述しないで下さい。実行したコマンドの結果(標準出力)が取り出せません。

.\* 付の後方一致の正規表記の例

‘.\*[¥#¥\$¥%] \$’

#### ② エスケープシーケンスの対応

構築対象機器に依存しますが、構築対象機器から送られてくるプロンプトの直前に Operating System Command シーケンスが付加されている場合があります。prompt パラメータで指定された文字列の直前にあるエスケープシーケンスを排除しています。

### (4) 複数具体値変数使用時の注意事項

対話ファイルで複数具体値変数が使用出来るパラメータは、command モジュールの with\_items パラメータのみです。これ以外で使用した場合、作業実行時にエラーとなります。

## 6.3 ロールパッケージ (Ansible-Legacy Role) の記述

基本書式については Ansible ベストプラクティスの公式マニュアルを参照してください。  
「5.3.4 ロールパッケージ管理」でアップロードするロールパッケージファイルの Zip に含めるべきディレクトリと、ITA での扱いについて記述します。

(上位ディレクトリ)

—site.yml	site.yml (マスターPlaybook) は ITA で作成します。 存在する場合は上書きします。
—hosts	hosts ファイルは ITA で作成します。 存在する場合は上書きします。
—group_vars	ホストグループ変数は扱えません。 group_vars ディレクトリが存在する場合は削除します。
—host_vars	ホスト変数は ITA で作成します。 host_vars ディレクトリが存在する場合は上書きします。
—ITA readme	ITA readme はロール毎に定義します。無くてもエラーにはなりません。 ITA readme は、文字コードが UTF-8 の BOM なしで作成して下さい。 ITA readme のファイル名の命名規則 ita_readme_[ロール名].yml e. g.) ロール名: mysql           ファイル名: ita_readme_mysql.yml ロール名: mysql/install   ファイル名: ita_readme_mysql%install.yml ※role のディレクトリ階層が深い場合、ロール名に 含まれる / を % に置き換える必要があります。
—読替表	読替表はロール毎に定義します。無くてもエラーにはなりません。 読替表のファイル名の命名規則 ita_translation-table_[ロール名].txt e. g.) ロール名: mysql           ファイル名: ita_translation-table_mysql.txt ロール名: mysql/install   ファイル名: ita_translation-table_mysql%install.txt ※role のディレクトリ階層が深い場合、ロール名に 含まれる / を % に置き換える必要があります。
	上記以外のディレクトリやファイルが存在する場合、ITA は関知しません。
—roles	roles ディレクトリが存在しない場合はアップロードでエラーになります。
—[role 名①]	role 名ディレクトリが存在しない場合はアップロードでエラーになります。 tasks ディレクトリを含むディレクトリを role として扱います。 ディレクトリ階層が深くても問題ありません。

```

|   ├──readme.mc      ITA は関知しません。
|   ├──tasks          tasks ディレクトリは必須です。
|   |   |              playbook ファイルは、文字コードが UTF-8 の BOM なで作成して下さい。
|   |   ├──main.yml   main.yml がいない場合はアップロードでエラーになります。
|   |   └──user_files  main.yml 以外のファイルも配置できます。
|   |       └──user.yml サブディレクトリに main.yml 以外のファイルを配置できます。
|   |
|   ├──handlers       handlers ディレクトリの有無は関知しません。
|   |   |              playbook ファイルは、文字コードが UTF-8 の BOM なしで作成して下さい。
|   |   ├──main.yml   main.yml の有無は関知しません。
|   |   └──user_files  main.yml 以外のファイルも配置できます。
|   |       └──user.yml サブディレクトリにファイルを配置できます。
|   |
|   ├──templates       templates ディレクトリの有無は関知しません。
|   |   ├──hosts.j2    サブディレクトリにファイルを配置できます。
|   |   └──user_files
|   |       └──user.j2
|   |
|   ├──files           files ディレクトリの有無は関知しません。
|   |   └──sudoers      ファイル及びサブディレクトリの有無は関知しません。
|   |                   ファイル内容は関知しません。
|   |
|   ├──vars            vars ディレクトリの有無は関知しません。
|   |   |              playbook ファイルは、文字コードが UTF-8 の BOM なしで作成して下さい。
|   |   └──main.yml     ファイル及びサブディレクトリの有無は関知しません。
|   |                   ファイル内容は関知しません。
|   |
|   ├──defaults        defaults ディレクトリの有無は関知しません。
|   |   |              playbook ファイルは、文字コードが UTF-8 の BOM なしで作成して下さい。
|   |   ├──main.yml   main.yml の有無は関知しません。
|   |   └──user_files  main.yml 以外のファイルも配置できます。
|   |       └──user.yml サブディレクトリに main.yml 以外のファイルを配置できます。
|   |
|   ├──meta            meta ディレクトリの有無は関知しません。
|   |   |              playbook ファイルは、文字コードが UTF-8 の BOM なしで作成して下さい。
|   |   └──main.yml    ファイル及びサブディレクトリの有無は関知しません。
|   |                   ファイル内容は関知しません。
|   |
|   上記以外のディレクトリやファイルが存在する場合、 ITA は関知しません。
|
└──[role 名②]          ロールの数に特に制限はありません。

```

## (1) マスターPlaybook

ITA で作成するマスターPlaybook はヘッダーセクションと roles セクションで構成されます。

### ① ヘッダーセクション

ヘッダーセクションは、デフォルト値が決まっていますが、「5.3.2.[Movement 一覧](#)」のヘッダーセクションで変更することが出来ます。

ヘッダーセクションのデフォルト値

・Ansible engine の場合

- hosts: all

remote\_user: "{{ \_\_loginuser\_\_ }}"

gather\_facts: no

become: yes

・AnsibleTower の場合

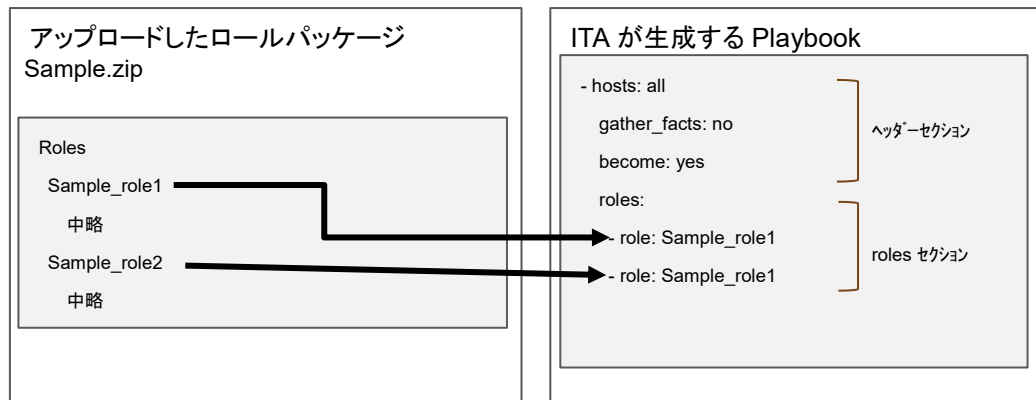
- hosts: all

gather\_facts: no

become: yes

### ② roles セクション

アップロードしたロールパッケージ内のロールを、「5.3.7.[Movement 詳細](#)」のインクルード順序に従い role で実行します。



## (2) 変数名の一意管理

ITA の代入値管理で登録された変数の情報はホスト変数として扱います。変数名はドライバー毎の全ロールパッケージで一貫管理します。ロールを跨って同じ変数名を使用しているが変数構造が違う場合は、アップロード時にエラーとなります。

例えば、通常変数と多段変数や多段変数同士で多段構造が違う場合など。

## (3) デフォルト変数定義ファイル(defaults->main.yml)の ITA 独自仕様

デフォルト変数定義ファイルの記述(変数定義)には ITA 独自仕様があります。

詳しくは、別資料「利用手順マニュアル\_Ansible-driver\_別紙\_Ansible 利用ガイドライン\_追加ルール」を参照してください。

## (4) Ansible Role Directory Structure における所定ディレクトリのサブディレクトリについて注意事項

Ansible Role Directory Structure における所定ディレクトリの下位に当該の所定ディレクトリ名と同名のサブディレクトリ(例えば、files ディレクトリの下位にそれと同名の files ディレクトリなど)を作成されていると、作業実行時にエラーになります。\*



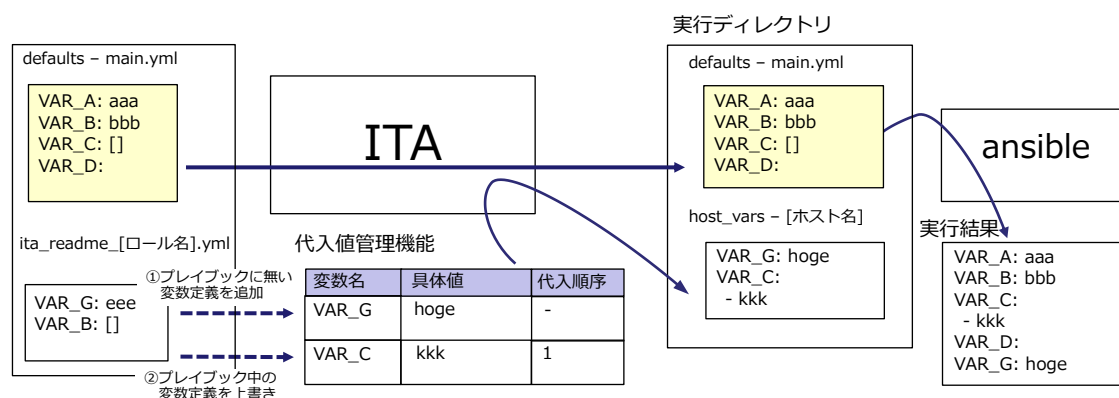
## 6.4 ITAreadme (Ansible-Legacy Role のみ) の記述

代入値管理機能は、defaults 変数定義ファイルに定義した変数の型を解釈して、各変数およびそのメンバー変数などに変数の値を設定します。

Playbook 中に直接変数を定義したくない場合など、defaults 変数定義ファイルに変数が定義されていない場合、ITA readme ファイルに変数の定義を設定することで、代入値管理機能で変数の値を指定することができます。

ITA readme ファイルは、文字コードが UTF-8 の BOM なしで作成して下さい。

ITA readme ファイルと、代入値管理機能の関係を以下の図に示します。



Playbook 中に無い変数を ITA readme ファイルで定義した場合、定義した変数を代入値管理機能で値を設定することができます。

また、Playbook 中の変数と異なる型を ITA readme ファイルで定義した場合、代入値管理機能には、上書きした変数の型で値を登録することができます。

代入値管理機能で設定した値は、各ホスト用に変数定義ファイル(`host_vars`)に出力され、Ansible で元の Playbook と変数定義用ファイルを入力として各ホストに実行されます。

ITA readme ファイルは代入値管理機能に変数情報を与えるために使用され、ITA readme に定義した変数および、変数の値は Ansible 実行には影響を与えません。

ITA readme の作成は任意です。ITA readme と defaults 変数定義ファイルで変数定義が重なった場合など、以下のルールで処理されます。

表 6.4-1 変数採用ルール

defaults 変数 定義ファイル	ITA readme	変数定義の採用先
定義あり	定義なし	デフォルト変数定義ファイル
定義なし	定義あり	ITA readme
定義あり	定義あり	ITA readme

また、「5.3.11 代入値管理」に表示するデフォルト値は以下のルールで処理されます。

表 6.4-2 デフォルト値表示ルール

defaults 変数 定義ファイル	ITA readme	デフォルト値の扱い
あり	なし	デフォルト変数定義ファイルを採用。
なし	あり	デフォルト値なしの扱いとなる。
あり	あり	デフォルト変数定義ファイルを採用。 ただし、変数定義が一致している場合のみ。変数定義が一致していない場合はデフォルト値なしの扱いとなる。

ITA readme は、作業実行時はロールパッケージから切り離されます。

ITA readme に記載した変数と具体値は適用されません。

## 6.5 読替表（Ansible-Legacy Role のみ）の記述

defaults 変数定義ファイルまたは ITA readme に定義されている「VAR\_xxx」以外の変数に対して、「5.3.11 代入値管理」機能で変数の具体値を設定出来るようにするための設定を行うファイルです。defaults 変数定義ファイルまたは ITA readme に定義されている「VAR\_xxx」以外の変数「任意変数」に対して代入値管理機能で扱う変数「読替変数」の紐付を定義します。

### (1) 読替表のフォーマット

テキスト形式で下記フォーマットとなります。

ロール内で読替変数と任意変数の組合せは一意である必要があります。

読替変数(\$s\*):(\$s+)任意変数

読替変数:LCA\_\*\*\*

\*\*\*: 半角英数字とアンダースコア( \_ )が利用可能です。(最小値:1 バイト、最大値:256 バイト)

任意変数: (最小値:1 バイト、最大値:256 バイト)

(\$s\*): 半角スペース 0 個以上

(\$s+): 半角スペース 1 個以上

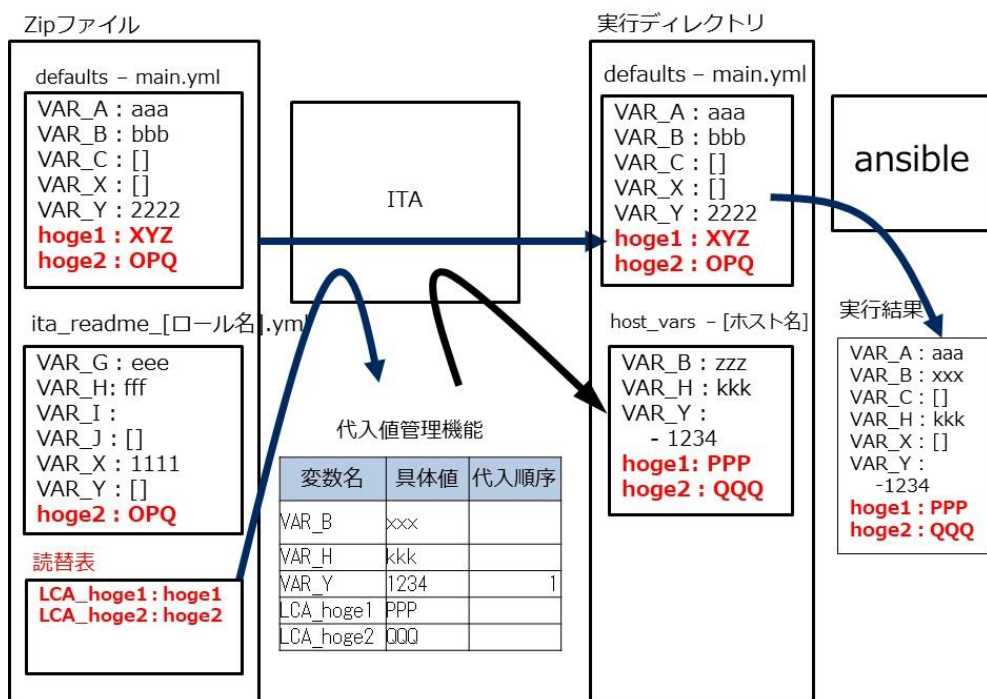
e.g.)

LCA\_var1: var1

# #から始まる行はコメント行

LCA\_var2: var2

代入値管理機能の関係を以下の図に示します。



## (2) 注意事項

読替表を作成する際の注意事項を列挙します。

ケース	ITA の動作	備考
読替表はあるが、defaults 変数定義ファイルと ITA readme が無い。(ロール毎)	読替表を読込まない。	
任意変数に VAR_ で始まる変数が定義されている。	ロールパッケージアップロード時にエラーになります。	
defaults 変数定義ファイルと ITA readme に定義されていない任意変数を使用している。	ロールパッケージアップロード時にエラーになります。	
ロール内で読替変数が重複定義されている	ロールパッケージアップロード時にエラーになります。	パッケージ A->ロール A LCA_A: user_A/LCA_A: user_B
ロール内で任意変数が重複定義されている	ロールパッケージアップロード時にエラーになります。	パッケージ A->ロール B LCA_A: user_A/LCA_B: user_A
ロール間で任意変数の構造に差異がある。	ロールパッケージアップロード時にエラーになります。	パッケージ A->ロール A/B LCA_C: user_C
ロールパッケージ内で置換変数と任意変数の組合せが一意でない。	ロールパッケージアップロード時にエラーになります。	パッケージ A ロール A LCA_D: user_D ロール B LCA_D: user_E
ロールパッケージ間で任意変数の構造に差異がある。	ロールパッケージアップロード時にエラーになりませんが、読替変数が代入値管理に表示されません。	パッケージ A->ロール A LCA_F: user_F パッケージ B->ロール A LCA_F: user_F
ロールパッケージ間で多段の任意変数を定義している	多段の構造が一致しているのでエラーにはなりませんが、多段繰返回数の設定は各パッケージで共通の設定となります。	パッケージ A->ロール B LCA_H: user_H パッケージ B->ロール A LCA_H: user_H

<p>パッケージ A</p> <pre>default/ITAreadme ロールA user_A: xxx user_B: xxx user_C: xxx  user_D: xxx user_E: xxx  user_F: xxx VAR_A: xxx</pre> <p>読替表</p> <p>ロールA</p> <pre>LCA_A: user_A LCA_A: user_B LCA_C: user_C LCA_D: user_D LCA_E: user_E  LCA_F: user_F LCA_G: VAR_A</pre>		<p>ロールB</p> <pre>user_A: xxx user_B: xxx user_C: - xxx user_D: xxx user_E: xxx  user_F: xxx  user_H: xxx - item1: xxx item2: xxx</pre> <p>読替表</p> <p>ロールB</p> <pre>LCA_A: user_A LCA_B: user_A LCA_C: user_C LCA_D: user_E LCA_E: user_E  LCA_H: user_H</pre>	
<p>パッケージ B</p> <pre>default/ITAreadme ロールA user_A: xxx user_B: xxx user_C: xxx  user_D: xxx user_E: xxx - xxx user_F: xxx  user_H: xxx - item1: xxx item2: xxx</pre> <p>読替表</p> <p>ロールA</p> <pre>LCA_F: user_F  LCA_H: user_H</pre>			

## 6.6 BackYard コンテンツ

### (1) 変数自動登録

変数解析対象の資材をアップロードした場合、アップロードされた資材から変数を取り出します。

表 6.6-1 モード別アップロードした資材の変数の扱い

メニュー	Legacy	Legacy Role	Pioneer
プレイブック素材集	○	×	×
ロールパッケージ管理	×	○	×
対話ファイル素材集	×	×	○

なお、取出すタイミングは 自動プロセスの起動周期 に依存します。

#### ※変数名の一意管理

取出した変数名は、モード毎に全資材で一意管理されます。

デフォルト変数定義ファイルで変数構造を定義しますので、各資材で変数構造が違う場合の留意事項を以下に明記します。

##### ●単一ロールパッケージ

ロールを跨って同じ変数名を使用しているが変数構造が違う場合。

※通常変数と多段変数や多段変数同士で多段構造が違う場合など

⇒ アップロード時にエラーとなります。

##### ●全ロールパッケージ

ロールパッケージを跨って同じ変数名を使用しているが変数構造が違う場合。

⇒ アップロード時にエラーとなります。

### (2) 代入自動値登録設定

連携対象としたパラメータシートのオペレーションとホスト毎の項目の設定値と紐付けた Movement と変数の情報を代入値管理と作業対象ホストに反映されます。

なお、反映のタイミングは前述と同様に 自動プロセスの起動周期 に依存します。

作業対象ホストと代入値管理は複数の操作者が更新を行います。最終更新者が他操作者の場合は反映処理をしません。

代入値自動登録設定のデータを反映したい場合は、代入値管理で該当レコードを廃止にする。他 BackYard 処理で該当レコードの更新を無効にする。などの操作を行ってください。

作業対象ホストと代入値管理への反映ルールを以下に明記します。

#### ① 代入値自動登録に登録されている情報を代入値管理へ反映時

代入値管理 の状態	該当レコード なし	該当レコードあり			該当レコード 廃止中
		=具体値	≠具体値		
			最終更新者		
			BackYard 処理	他操作者	
代入値管理 への反映	新規レコード 追加	-	該当レコードの 具体値更新	-	廃止レコード 復活

※ 該当レコード: オペレーション+ホスト+Movement+変数名+(メンバー変数)+(代入順序)が同一のレコードの意

② 代入値自動登録に登録されていない情報(代入値管理のみに登録)を代入値管理へ反映時

代入値管理 の状態	該当レコードあり	
	最終更新者	
	BackYard 処理	他操作者
代入値管理への反映	該当レコード廃止	-

③ 代入値自動登録に登録されている情報を作業対象ホストへの反映時

作業対象ホストの状態	該当レコードなし	該当レコードあり	該当レコード廃止中
作業対象ホストへの反映	新規レコード追加	-	廃止レコード復活

※ 該当レコード: オペレーション+ホスト+Movement が同一のレコードの意

④ 代入値自動登録に登録されていない情報(作業対象ホストのみに登録)を作業対象ホストへ反映時

代入値管理の状態	該当レコードあり	
	最終更新者	
	BackYard 処理	他操作者
代入値管理への反映	該当レコード廃止	-

- ⑤ 同一 Movement と変数と代入順序に、複数項目の紐付が登録されている場合。  
複数のパラメータシートで同一のオペレーションとホストが紐付されている場合は、ランダムに 1 項目を選択し代入値管理へ反映を行います。

パラメータシート: sheetA

No	オペレーション	ホスト	項目 1
1	Op1	Host01	Value1
2	Op1	Host02	Value2

代入値自動登録設定

パラメータシート		Movement	登録方式	IaC 変数	
メニュー	項目			Value 変数	代入順序
sheetA	項目 1	設定	Value	VAR_val	
sheetB	項目 1	設定	Value	VAR_val	

オペレーション:Op1 とホスト:Host01 が複数紐付けられているので、ランダムに選ばれた、パラメータシート:sheetA の項目の設定値が代入値管理に反映されます。

パラメータシート: sheetB

No	オペレーション	ホスト	項目 1
1	Op1	Host01	Value4
2	Op1	Host03	Value3

代入値管理

オペレーション	ホスト	Movement	変数名	代入順序
Op1	Host01	設定	VAR_val	Value1
Op1	Host02	設定	VAR_val	Value2
Op1	Host03	設定	VAR_val	Value3

## 6.7 Ansible 利用ガイドライン ITA 追加ルール

ITA を使用して、Ansible で実行する為の Playbook 作成ガイドラインを記述します。

詳しくは、別資料「利用手順マニュアル\_Ansible-driver\_別紙\_Ansible 利用ガイドライン\_追加ルール」を参照してください。

## 7 運用操作

本機能を活用する操作は、クライアント PC のブラウザ画面からのユーザー利用による入力だけでなく、システム運用・保守による操作もあります。用意している運用・保守の操作は次のとおりです。

### 7.1 メンテナンス

Ansible-driver のプロセスの開始/停止/再起動に必要なファイルは以下となります。

説明	対象ファイル名
Legacy/pioneer/legacyRole 実行監視 未実行作業の実行を行う。	ky_ansible_execute-workflow.service
legacy 変数自動登録 アップロードした資材から変数の取出しを行う。	ky_legacy_varsautolistup-workflow.service
legacy 自動登録設定 自動登録設定に設定された情報を代入値管理と作業対象メニューに反映を行う。	ky_legacy_valautostup-workflow.service
pioneer 自動登録設定 自動登録設定に設定された情報を代入値管理と作業対象メニューに反映を行う。	ky_pioneer_valautostup-workflow.service
legacyRole 変数自動登録 アップロードした資材から変数の取出しを行う。	ky_legacy_role_varsautolistup-workflow.service
legacyRole 自動登録設定 自動登録設定に設定された情報を代入値管理と作業対象メニューに反映を行う。	ky_legacy_role_valautostup-workflow.service
AnsibleTower データ同期 Ansible-tower から設定情報の取得を行う。	ky_ansible_towermasterSync-workflow.service

対象ファイルは「/usr/lib/systemd/system」に格納されています。  
プロセス起動/停止/再起動の方法は次の通りです。  
root 権限でコマンドを実行してください。



① プロセス起動

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

② プロセス停止

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

③ プロセス再起動

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

各対象ファイル名に置き換えて起動/停止/再起動を行ってください。

## 7.2 メンテナンス方法について

① NORMAL レベルへの変更

以下のファイルの 8 行目「DEBUG」を「NORMAL」に書き換えます。

ログレベル設定ファイル: <インストールディレクトリ>/ita-root/conf/backyardconfs/ita\_env

② DEBUG レベルへの変更

以下のファイルの 8 行目「NORMAL」を「DEBUG」に書き換えます。

ログレベル設定ファイル: <インストールディレクトリ>/ita-root/conf/backyardconfs/ita\_env

③ 起動周期の変更

各対象ファイルの ExecStart の 5 番目のパラメータを変更します。(単位:秒)

例外を除き起動周期はデフォルト値の使用をしてください。

```
ExecStart=/exastro/ita-root/backyards/common/ky_loopcall-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
```

書き換え後、**プロセス再起動(restart)後に有効になります。**

ログファイルの出力先: <インストールディレクトリ>/ita-root/logs/backyardlogs

## 8 付録

### 8.1 Ansible 実行時に使用される投入データと ITA メニューの紐づけ

各 ITA メニューより情報を抽出して Ansible 実行に必要な[投入データ]を作ります。この際、機器一覧のパスワードは暗号化されます。Ansible-Legacy と Ansible-LegacyRole は ansible-vault で暗号化を行い、pioneer は ITA 独自方式で暗号化を行います。

[投入データ]は ZIP 形式で「[5.3.12 作業状態確認](#)」よりダウンロードが可能です。以下のコマンドで Ansible を直接実行することも可能です。

#### Ansible-Legacy

```
ansible-playbook (オプション) -i hosts --vault-password-file パスワードファイル playbook.yml
```

#### Ansible-Pioneer

```
ansible-playbook (オプション) -i hosts playbook.yml
```

#### Ansible-LegacyRole

```
ansible-playbook (オプション) -i hosts --vault-password-file パスワードファイル site.yml
```

パスワードファイル名は任意です。パスワードファイルに設定するパスワードは、ITA インストール先 /ita-root/conf/commonconfs/ansible\_vault\_accesskey.txt の内容を、rot13、base64 の順でデコードした値を使用して下さい。

各種データと ITA メニューの関係性は以下の通りです。

### 8.1.1 Ansible-Legacy 投入データ

#### 【上位ディレクトリ】

child_playbooks	ユーザーが作成した Playbookt が格納されるディレクトリ			
	{ Ansible-Legacy	プレイブック素材集	プレイブック素材	
template_files	実行する Playbook 内で使用するテンプレート管理のファイルが格納されるディレクトリ			
	{ Ansible-Legacy	テンプレート管理	テンプレート素材	
copy_files	実行する Playbook 内で使用しているファイル管理のファイルを格納するディレクトリ			
	{ Ansible-Legacy	ファイル管理	ファイル素材	
host_vars	ホスト変数ファイルが格納されるディレクトリ			
	{	Ansible 共通	グローバル変数管理	変数名/具体値
		Ansible-Legacy	代入値管理	変数名/具体値
		Ansible-Legacy	template 管理	テンプレート素材
		Ansible-Legacy	ファイル管理	ファイル変数名
		Ansible-Legacy	インターフェース情報	データリレイストレージパス (ITA)
		Ansible-Legacy	インターフェース情報	Symphony インスタンスデータリレイストレージパス(ANS)
		Ansible-Legacy	インターフェース情報	Conductor インスタンスデータリレイストレージパス(ANS)
		基本コンソール	機器一覧	プロトコル
		基本コンソール	機器一覧	ログインユーザ ID
		基本コンソール	機器一覧	ログインパスワード
				※ansible-vault で暗号化
		基本コンソール	機器一覧	ホスト名

ssh_key_files	認証方式が鍵方式の場合、ssh 認証鍵ファイルが格納されるディレクトリ		
	{	基本コンソール      機器一覧	ssh 認証鍵ファイル
winrm_ca_files	https で WinRM 接続する場合のサーバ証明書が格納されるディレクトリ		
	{	基本コンソール      機器一覧	サーバ証明書
AnsibleExecOption.txt	AnsiblePlaybook 実行時のオプションパラメータ		
	{	Ansible 共通      インターフェース情報      オプションパラメータ	
		Ansible-Legacy      Movement 一覧      オプションパラメータ	
hosts	インベントリファイル		
	{	基本コンソール      機器一覧	ログインユーザ ID
		基本コンソール      機器一覧	ログインパスワード
			※ansible-vault で暗号化
		基本コンソール      機器一覧	ホスト名
		基本コンソール      機器一覧	IP アドレス
		基本コンソール      機器一覧	インベントリファイル追加オプション
		基本コンソール      機器一覧	WinRM 接続情報
		基本コンソール      機器一覧	接続オプション
			※ansible_ssh_extra_args の値
playbook.yml	Movement 詳細で指定された Playbook を実行するマスタ Playbook		
	{	Ansible-Legacy      プレイブック素材集      プレイブック素材	
		Ansible-Legacy      Movement 詳細      インクルード順序	

## 8.1.2 Ansible-Pioneer 投入データ

### 【上位ディレクトリ】

child_playbooks	ユーザーが作成した Playbookt が格納されるディレクトリ		
	{ Ansible-Legacy	プレイブック素材集	プレイブック素材
template_files	実行する Playbook 内で使用するテンプレート管理のファイルが格納されるディレクトリ		
	{ Ansible-Legacy	テンプレート管理	テンプレート素材
copy_files	実行する Playbook 内で使用しているファイル管理のファイルを格納するディレクトリ		
	{ Ansible-Legacy	ファイル管理	ファイル素材
host_vars	ホスト変数ファイルが格納されるディレクトリ		
	Ansible 共通	グローバル変数管理	変数名/具体値
	Ansible-Pioneer	代入値管理	変数名/具体値
	Ansible-Pioneer	template 管理	テンプレート素材
	Ansible-Pioneer	ファイル管理	ファイル変数名
	Ansible-Pioneer	インターフェース情報	データリレイストレージパス (ITA)
	Ansible-Pioneer	インターフェース情報	Symphony インスタンスデータリレイストレージパス(ANS)
	Ansible-Pioneer	インターフェース情報	Conductor インスタンスデータリレイストレージパス(ANS)
	基本コンソール	機器一覧	プロトコル
	基本コンソール	機器一覧	ログインユーザ ID
	基本コンソール	機器一覧	ログインパスワード ※base64、rot13 の順でエンコードした値
	基本コンソール	機器一覧	ホスト名
	基本コンソール	機器一覧	Pioneer 利用情報
	Ansible-Pioneer	Movement 一覧	並列実行数

ssh_key_files	認証方式が鍵方式の場合、ssh 認証鍵ファイルが格納されるディレクトリ		
	<div> <div></div> <div>基本コンソール</div> <div>機器一覧</div> <div>ssh 認証鍵ファイル</div> </div>		
AnsibleExecOption.txt	AnsiblePlaybook 実行時のオプションパラメータ		
	<div> <div>Ansible 共通</div> <div>インターフェース 情報</div> <div>オプションパラメータ</div> </div>		
	<div> <div>Ansible-Pioneer</div> <div>Movement 一覧</div> <div>オプションパラメータ</div> </div>		
hosts	インベントリファイル		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>ログインユーザ ID</div> </div>		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>ログインパスワード</div> </div>		
	※base64、rot13 の順でエンコードした値		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>ホスト名</div> </div>		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>IP アドレス</div> </div>		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>インベントリファイル追加オプション</div> </div>		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>Pioneer 利用情報</div> </div>		
	<div> <div>基本コンソール</div> <div>機器一覧</div> <div>接続オプション</div> </div>		
	※ansible_ssh_extra_args の値		
playbook.yml	Movement 詳細で指定された対話ファイルを実行するマスタ Playbook		
	<div> <div>Ansible-Pioneer</div> <div>話ファイル 素材 集</div> <div>話ファイル素材</div> </div>		
	<div> <div>Ansible-Pioneer</div> <div>Movement 詳細</div> <div>インクルード順序</div> </div>		

### 8.1.3 Ansible-LegacyRole 投入データ

#### 【上位ディレクトリ】

child_playbooks	ユーザーが作成した Playbookt が格納されるディレクトリ		
	Ansible-LegacyRole	プレイブック素材集	プレイブック素材
template_files	実行する Playbook 内で使用するテンプレート管理のファイルが格納されるディレクトリ		
	Ansible-LegacyRole	テンプレート管理	テンプレート素材
copy_files	実行する Playbook 内で使用しているファイル管理のファイルを格納するディレクトリ		
	Ansible-LegacyRole	ファイル管理	ファイル素材
host_vars	ホスト変数ファイルが格納されるディレクトリ		
	Ansible 共通	グローバル変数管理	変数名/具体値
	Ansible-LegacyRole	代入値管理	変数名/具体値
	Ansible-LegacyRole	template 管理	テンプレート素材
	Ansible-LegacyRole	ファイル管理	ファイル変数名
	Ansible-LegacyRole	インターフェース情報	データリレイストレージパス (ITA)
	Ansible-LegacyRole	インターフェース情報	Symphony インスタンスデータリレイストレージパス(ANS)
	Ansible-LegacyRole	インターフェース情報	Conductor インスタンスデータリレイストレージパス(ANS)
	基本コンソール	機器一覧	プロトコル
	基本コンソール	機器一覧	ログインユーザ ID
	基本コンソール	機器一覧	ログインパスワード
			※ansible-vault で暗号化
	基本コンソール	機器一覧	ホスト名

ssh_key_files	認証方式が鍵方式の場合、ssh 認証鍵ファイルが格納されるディレクトリ		
	{ 基本コンソール	機器一覧	ssh 認証鍵ファイル
winrm_ca_files	https で WinRM 接続する場合のサーバ証明書が格納されるディレクトリ		
	{ 基本コンソール	機器一覧	サーバ証明書
AnsibleExecOption.txt	AnsiblePlaybook 実行時のオプションパラメータ		
	{ Ansible 共通	インターフェース情報	オプションパラメータ
	{ Ansible-Legacy	Movement 一覧	オプションパラメータ
hosts	インベントリファイル		
	{ 基本コンソール	機器一覧	ログインユーザ ID
	{ 基本コンソール	機器一覧	ログインパスワード
			※ansible-vault で暗号化
	{ 基本コンソール	機器一覧	ホスト名
	{ 基本コンソール	機器一覧	IP アドレス
	{ 基本コンソール	機器一覧	インベントリファイル追加オプション
	{ 基本コンソール	機器一覧	WinRM 接続情報
	{ 基本コンソール	機器一覧	接続オプション
			※ansible_ssh_extra_args の値
playbook.yml	Movement 詳細で指定された Role を実行するマスタ Playbook		
	{ Ansible-Legacy	プレイブック素材集	プレイブック素材
	{ Ansible-Legacy	Movement 詳細	インクルード順序



## 8.2 Ansible 実行時に作成される結果データ

[投入データ]を ansible で実行した結果を[結果データ]として ZIP 形式で保存します。  
[結果データ]は ZIP 形式で「5.3.12 作業状態確認」よりダウンロードが可能です。

### 8.2.1 Legacy/LegacyRole 結果データに保存されるファイル一覧

表 8.2.1-1 Legacy/LegacyRole 結果データの保存されるファイル一覧

ファイル名	記録内容	Ansible Engine の場合	Ansible Tower の場合
result.txt	Ansible の実行結果を記録	○	
xxx.pid	Ansible-playbook コマンドのプロセス ID を記録するファイル。 xxx:Ansible-playbook コマンドのプロセス ID	○	
error.log	作業実行時に ITA がなにかしらのエラーによりエラーメッセージを出力した場合、または、Ansible-playbook コマンドがなにかしらのエラーによりエラーメッセージを出力した場合のエラー出力先ファイル。 作業実行確認のエラーログに表示に表示される内容。	○	○
exec.log	Ansible-playbook が出力した実行ログを一部加工したログファイル。作業実行確認の実行ログに表示される内容。	○	○
exec.log.org	Ansible-playbook が出力した実行ログ	○	○
Ita_<mode 名>_executions.jobtpl_<作業番号>_<グループ番号>	分割された実行ログファイルです。 ファイル名の命名規則は 5.3.12 作業状態確認の⑥実行ログ表示を参照下さい。		○
forced.txt	緊急停止をした場合の記録ファイル	○	
user_files	実行した playbook で ITA 独自変数「__workflowdir__」になにかしらのファイル出力をした場合のファイルが記録されるディレクトリ。	○	○

## 8.2.2 Pioneer 結果データに保存されるファイル一覧

表 8.2.1-2 Pioneer 結果データの保存されるファイル一覧

ファイル名	記録内容	Ansible Engine の場合	Ansible Tower の場合
result.txt	Ansible の実行結果を記録	○	
xxx.pid	Ansible-playbook コマンドのプロセス ID を記録するファイル。 xxx: Ansible-playbook コマンドのプロセス ID	○	
pioneer.xxx	Pioneer モジュールのプロセス ID を記録するファイル。 xxx: Pioneer モジュールのプロセス ID	○	○
error.log	作業実行時に ITA がなにかしらのエラーによりエラーメッセージを出力した場合、または、Ansible-playbook コマンドがなにかしらのエラーによりエラーメッセージを出力した場合のエラー出力先ファイル。 作業実行確認のエラーログに表示に表示される内容。	○	○
exec.log	Ansible-playbook が出力した実行ログを一部加工したログファイル。作業実行確認の実行ログに表示される内容。	○	○
exec.log.org	Ansible-playbook が出力した実行ログ	○	○
Ita_<mode 名>_executions.jobtpl_<作業番号>_<グループ番号>	分割された実行ログファイルです。 ファイル名の命名規則は 5.3.12 作業状態確認の⑥実行ログ表示を参照下さい。		○
forced.txt	緊急停止をした場合の記録ファイル	○	
user_files	実行した playbook で ITA 独自変数「__workflowdir__」になにかしらのファイル出力をした場合のファイルが記録されるディレクトリ。	○	○

(1) Legacy-Role

表 9.2.1-3 Legacy-Role 結果データの保存されるファイル一覧

ファイル名	記録内容	Ansible Engine の場合	Ansible Tower の場合
result.txt	Ansible の実行結果を記録	○	
xxx.pid	Ansible-playbbok コマンドのプロセス ID を記録するファイル。 xxx:Ansible-playbbok コマンドのプロセス ID	○	
error.log	作業実行時に ITA がなにかしらのエラーによりエラーメッセージを出力した場合、または、Ansible-playbbok コマンドがなにかとらのエラーによりエラーメッセージを出力した場合のエラー出力先ファイル。 作業実行確認のエラーログに表示に表示される内容。	○	○
exec.log	Ansible-playbbok が出力した実行ログを一部加工したログファイル。作業実行確認の実行ログに表示される内容。	○	○
exec.log.org	Ansible-playbbok が出力した実行ログ	○	○
Ita_<mode 名>_ executions.jobtpl_ <作業番号>_<グループ番号>	分割された実行ログファイルです。 ファイル名の命名規則は 5.3.12 作業状態確認の⑥実行ログ表示を参照下さい。		○
forced.txt	緊急停止をした場合の記録ファイル	○	
user_files	実行した playbook で ITA 独自変数「__workflowdir__」になにかしらのファイル出力をした場合のファイルが記録されるディレクトリ。	○	○