ITA_利用手順マニュアル

Ansible-driver

ー第1.10版ー

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写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。
- Ansible Automation Controllerは、Red Hat, Inc.の登録商標または商標です。
- Ansible Automation Pratform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。
※ 「Ansible Tower」はAnsible Automation Pratform2.0以降で「Ansible Automation Controller」に名称が変更になりました。本書での「Ansible Automation Controller」の記載は、「Ansible Tower」も含めた記載となっています。

目次

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、Ansible Automation Controller および Ansible driver について説明します。

1.1 Ansible Core について

Ansible Core とは、多数の構築管理対象に対して、アプリーケーション/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 Core 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 Core 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 Core の詳細情報については、Ansible Core のマニュアルを参照してください。

1.2 Ansible Automation Controller について

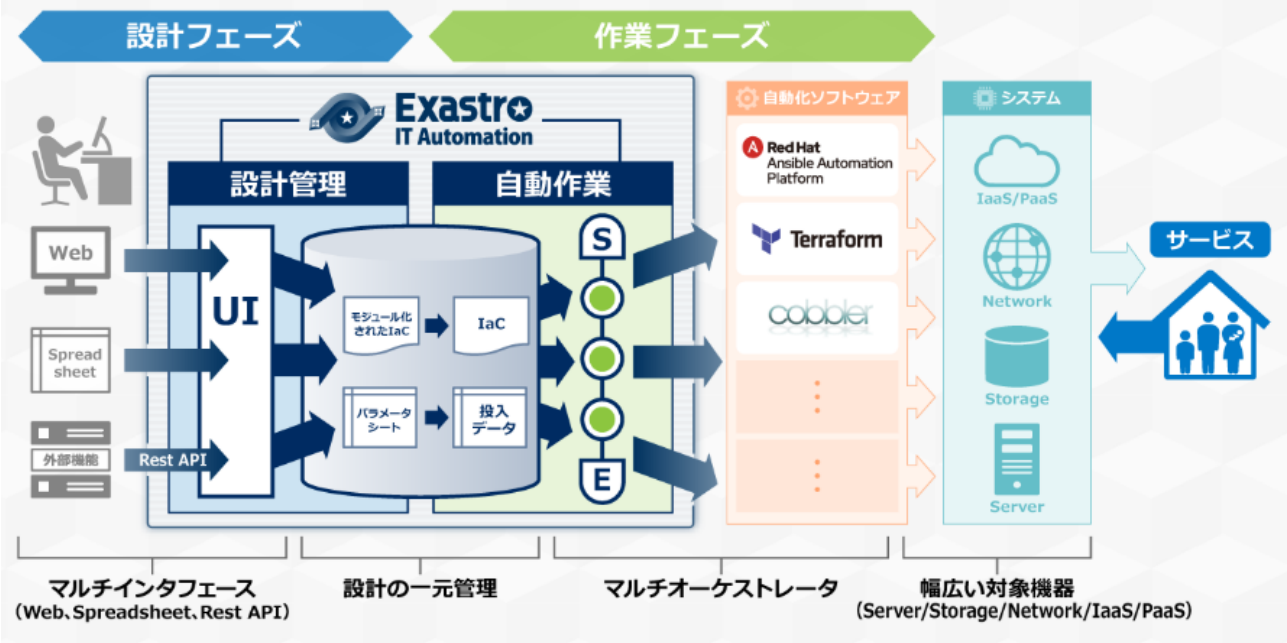
Ansible Automation Controller とは、PF 構築自動化ツールである Ansible にアクセスコントロール、ジョブスケジューリング、タスクの可視化などの機能を拡張した管理プラットフォームです。

“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”を組み合せて”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。

Ansible Automation Controller の詳細情報については、Ansible Automation Controller のマニュアルを参照してください。
ITA で対応可能な Ansible Automation Controller 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 Core、Ansible Automation Controller 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 8 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 {{△VAR_xxx△}} 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 {{△VAR_xxx△}} 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 {{△VAR_xxx△}} 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 Yaml syntax 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。	○	○	○
ﾃﾝﾌﾟﾚｰﾄ埋込変数	「テンプレート管理」メニューから登録された変数です。	○	○	○
ファイル埋込変数	「ファイル管理」メニューから登録された変数です。	○	○	○

<table>
<tr><th>種類</th><th>内容</th><th>Legacy</th><th>Pioneer</th><th>Legacy Role</th></tr>
<tr><td>ITA 独自変数</td><td>
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>
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。

作業実行時のオペレーション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>オペレーション</td><td>__operation__</td></tr>
</table>
設定値 ： 実施予定日時「YYYY/MM/DD HH:MM」_オペレーションID:オペレーション名称

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>
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>
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>
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。
</td><td>○</td><td>○</td><td>○</td></tr>
</table>

<table>
<tr><td></td><td>Conductor の「Status file branch」ノードで参照するステータスファイルのファイル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>ステータスファイルパス</td><td>__movement_status_filepath__</td></tr></table>
Playbook 内で作業ディレクトリパス配下にステータスファイルを作成することでができます。

収集機能の各ファイル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(in)の「_parameters」のパス</td><td>__parameters_dir_for_epc__</td></tr><tr><td>作業ディレクトリ(in)の「_parameters_file」のパス</td><td>__parameters_file_dir_for_epc__</td></tr><tr><td>作業結果ディレクトリ(out)の「_parameters」のパス</td><td>__parameter_dir__</td></tr><tr><td>作業結果ディレクトリ(out)の「_parameters_file」のパス</td><td>__parameters_file_dir__</td></tr></table>「_parameters」 :ソースファイル(パラメータ)格納先用
「_parameters_file」:収集したファイルの格納先用
※パラメータの対象がファイルアップロードカラムの場合のファイル配置
収集機能の詳細については、「ITA_利用手順マニュアル 収集機能」を参照して下さい。</td><td></td><td></td><td></td></tr>
<tr><td>読替変数</td><td>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「 6.5 読替表(Ansible-Legacy Role のみ)の記述」を参照して下さい。</td><td>×</td><td>×</td><td>○</td></tr>
</table>

2.2 変数の抜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出し、各メニューから具体値を登録できます。各メニューから登録した変数の具体値は、作業実行時にホスト変数ファイルに出力されます。
変数の抜出方法は以下のとおりです。

(1) Ansible-Legacy

「Playbook 素材集 (本書:5.3.3Playbook 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抜出します。

書式	具体値の設定
{{△VAR_xxx△}} {{△VAR_xxx△|△ﾌｨﾙﾀｰ△}}	具体値の登録は「5.3.9 代入値自動登録設定」や「5.3.11 代入値管理」メニューより行います。 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。
{{△GBL_xxx△}} {{△GBL_xxx△|△ﾌｨﾙﾀｰ△}}	具体値の登録は「5.2.3 グローバル変数管理」メニューより行います。
{{△TPF_xxx△}} {{△TPF_xxx△|△ﾌｨﾙﾀｰ△}}	具体値の登録は「5.2.4 テンプレート管理」メニューより行います。
{{△CPF_xxx△}} {{△CPF_xxx△|△ﾌｨﾙﾀｰ△}}	具体値の登録は「5.2.5 ファイル管理」メニューより行います。

※ △:半角スペース xxx: 半角英数字とアンダースコア(_)

(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、以下の書式の変数定義を抜出します。

書式	具体値の設定
{{△VAR_xxx△}}	具体値の登録は「5.3.9 代入値自動登録設定」や「5.3.11 代入値管理」メニューより行います。 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。
{{△GBL_xxx△}}	具体値の登録は「5.2.3 グローバル変数管理」メニューより行います。
{{△TPF_xxx△}}	具体値の登録は「5.2.4 テンプレート管理」メニューより行います。
{{△CPF_xxx△}}	具体値の登録は「5.2.5 ファイル管理」メニューより行います。

(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 defaults 変数定義ファイルより変数定義の抜出を行います。
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。

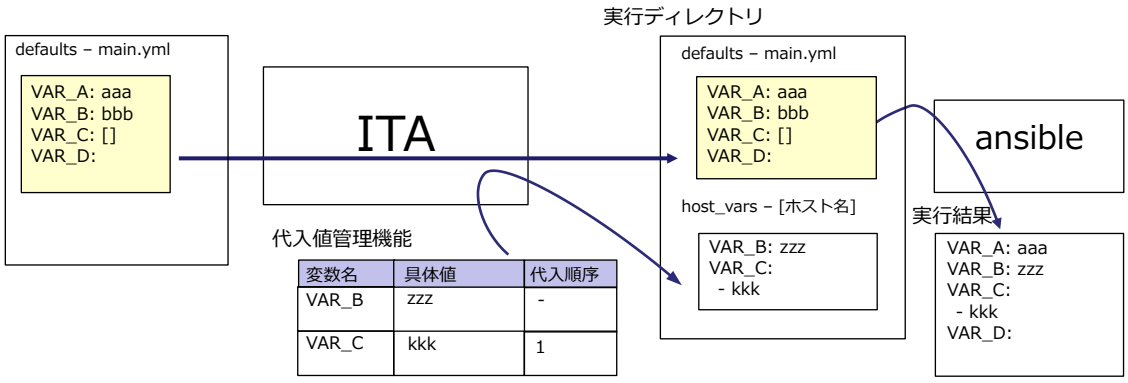
また、アップロードしたロールパッケージ内の playbook より、以下の書式の変数定義を抜出します。

書式	ロールパッケージ内のディレクトリ		具体値の設定
	tasks templates handlers meta	その他	
{{△GBL_xxx△}} {{△GBL_xxx△|△ﾌｨﾙﾀｰ△}}	○	×	具体値の登録は「5.2.3 グローバル変数管理」メニューより行います。
{{△TPF_xxx△}} {{△TPF_xxx△|△ﾌｨﾙﾀｰ△}}			具体値の登録は「5.2.4 テンプレート管理」メニューより行います。
{{△CPF_xxx△}} {{△CPF_xxx△|△ﾌｨﾙﾀｰ△}}			具体値の登録は「5.2.5 ファイル管理」メニューより行います。

※　○：変数定義抜出対象の playbook　　×：変数定義抜出対象外の playbook

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

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	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 Core、Ansible Automation Controller 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 Automation Controller ホスト一覧	Ansible Automation Controller の RestAPI 実行に必要な情報、および構築資材を Ansible Automation Controller 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 モジュールで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 Ansible コンソール			メニュー・画面	非表示メニュー※1	説明
	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	○			Playbook 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を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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）		Movement とプレイブック素材の関連付けを管理します。
8		○		変数ネスト管理		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。
15	○	○	○	変数名一覧	○	Legacy: Playbook 素材集にアップロードした Playbook で使用している変数名を管理します。 pioneer: 対話ファイル素材集にアップロードした対話ファイルで使用している変数名を管理します。 Legacy:-role: ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している変数名を管理します。
16	○	○	○	Movement 変数紐付管理	○	Movement で使用している変数を管理します。
17		○		ロール名管理	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内に登録しているロールを管理します
18		○		ロール変数名管理	○	ロールパッケージ管理にアップロードしたロールパッケージフ

No	メニューグループ Ansible コンソール			メニュー・画面	非表示メニュー※1	説明
	Legacy	Legacy Role	Pioneer			
						ァイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している変数名をロール毎に管理します。
19		○		変数具体値管理	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している変数の具体値を管理します。
20		○		メンバー変数管理	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している多段変数のメンバー変数を管理します。
21		○		多段変数メンバー管理	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している多段変数の構造を管理します。
22		○		多段変数配列組合せ管理	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内のデフォルト変数定義ファイルや ITAreadme ファイルで定義している多段変数の繰り返し数を管理します。
23		○		読替変数一覧	○	ロールパッケージ管理にアップロードしたロールパッケージファイル「zip」内の読替表ファイルで定義している変数を管理します。

※1 非表示メニューは、バックヤード機能でデータの登録・更新を行うメニューです。
Ansible Driver 機能をインストールした状態では表示されないメニューに設定されています。
非表示メニューを表示するには、「管理コンソール/ロール・メニュー紐付管理」で各メニューの復活処理を行います。詳細は「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。
尚、データの更新を行うとバックヤード機能が正しく動作しなくなります。データの更新はしないで下さい。

4 Ansible driver 利用手順

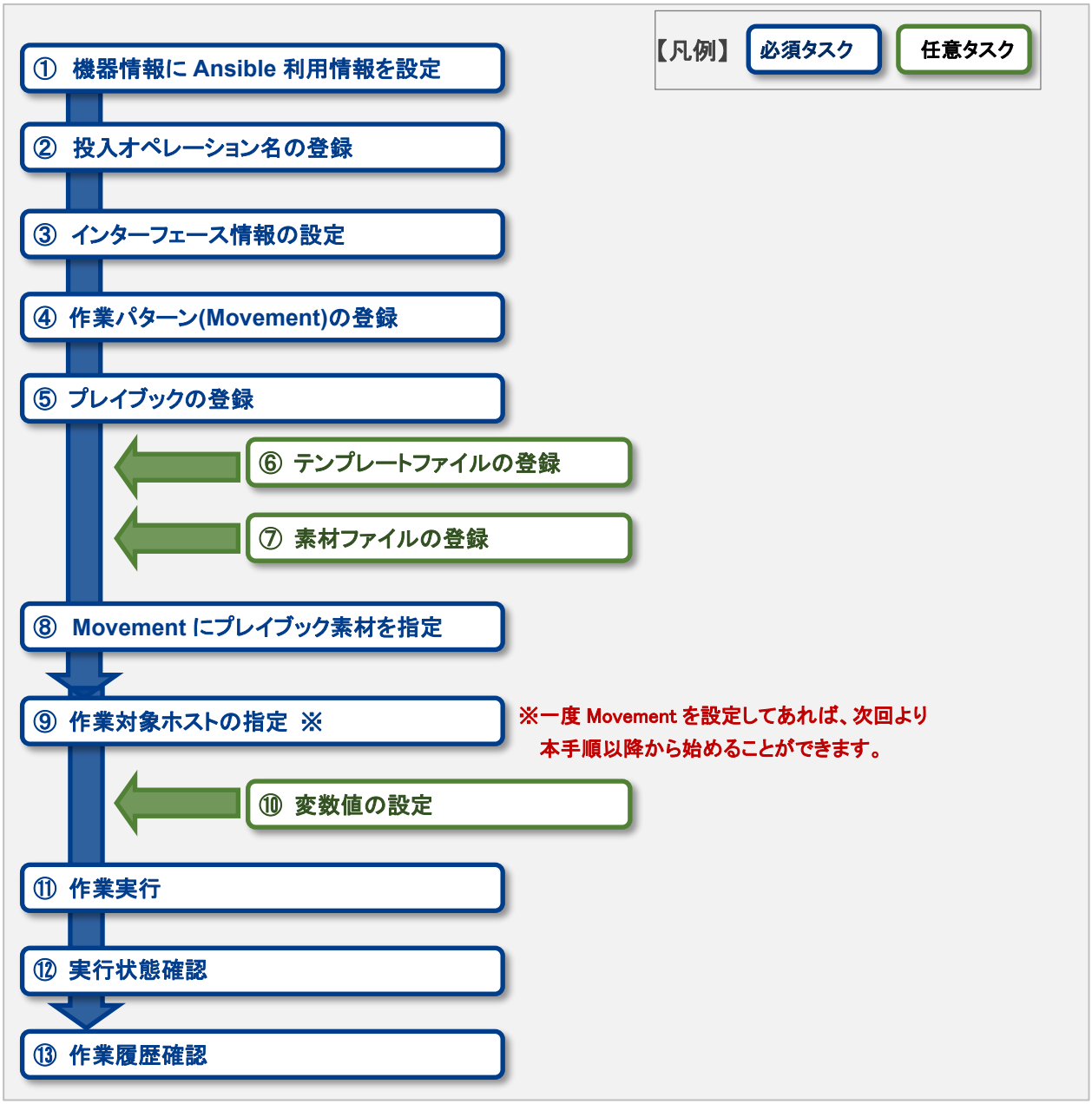
各 Ansible コンソールの利用手順について説明します

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 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 Core、Ansible Automation Controller 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 5.3.2Movement 一覧 を参照してください。

⑤ Playbook の登録

Ansible-Legacy コンソールの Playbook 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 5.3.3Playbook 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付） 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。

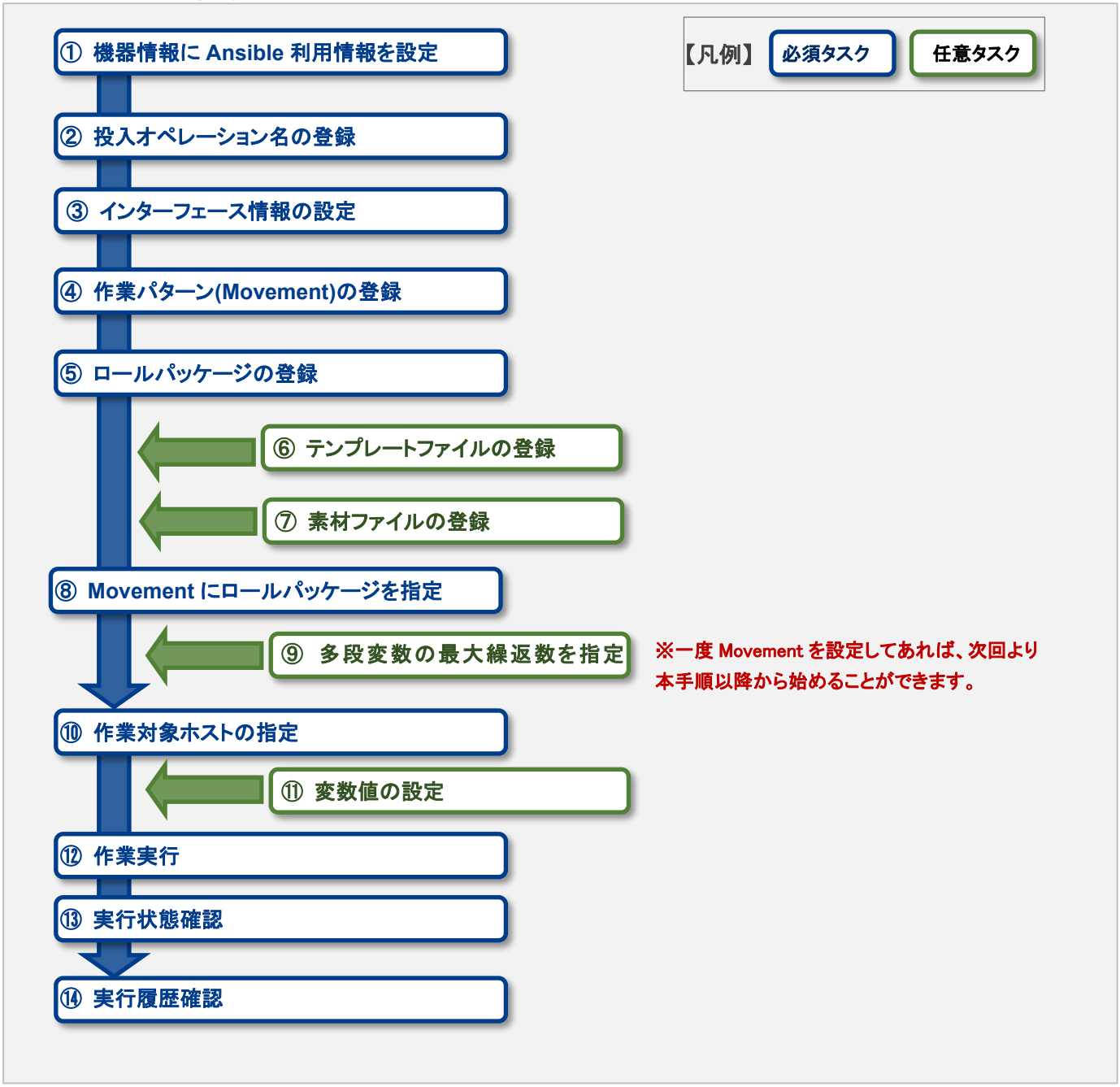
⑬ **作業履歴確認**

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 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 Core、Ansible Automation Controller 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 5.3.2Movement 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付） 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 変数ネスト管理（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。

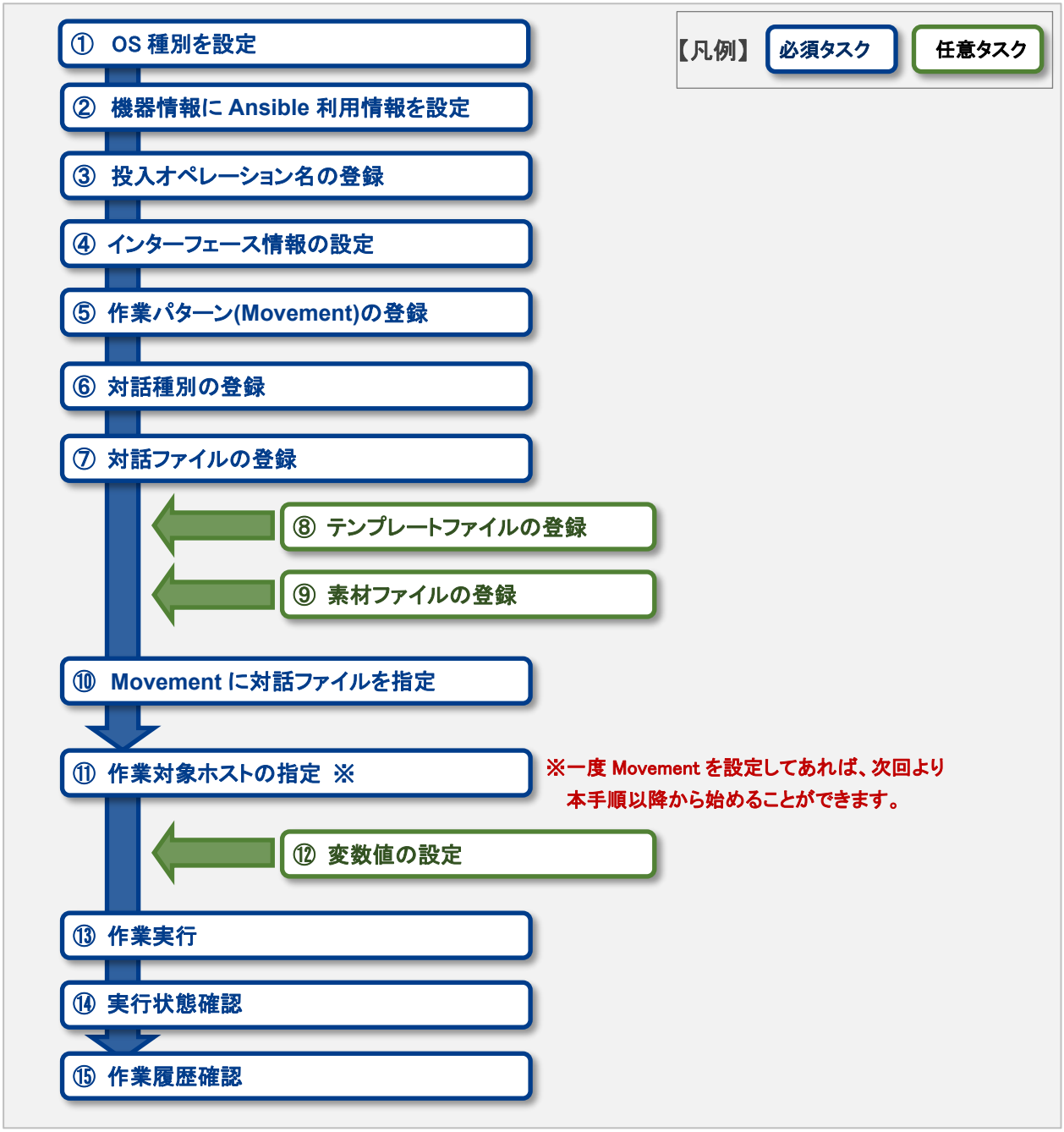
⑭ **作業履歴確認**

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 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 Core、Ansible Automation Controller 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 5.3.2Movement 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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	②	③	④	⑤
項目	説明	入力必須	入力形式	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

・〇: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、主に Ansible driver の動作に必要となる項目について説明します。
「利用手順マニュアル_基本コンソール」と合わせて参照してください。

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)

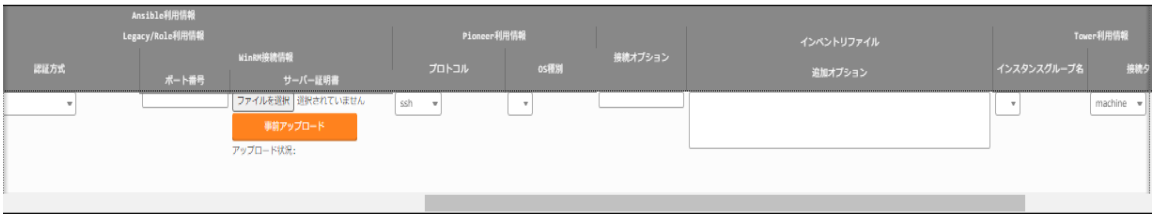


図 5.1-3 登録画面(機器一覧 - Ansible 利用情報)

(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 ※ホスト名を localhost に設定し pioneer で作業対象ホストとし使用する場合、作業実行でエラーになる場合があります。その場合、インベントリファイル追加オプションに下記パラメータで ansible サーバーにインストールされている python3 のパスを追記して下さい。 Exp) ansible_python_interpreter: /usr/bin/python3	○	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
EtherWake 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	パスワードを指定します	○	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
		パスワード				
ssh鍵認証情報		ssh 秘密鍵ファイル	ssh 秘密鍵ファイルを指定して鍵認証する場合の秘密鍵ファイルを入力します。 アップロードしたファイルは暗号化されて保存されます。※登録後はダウンロー不可となります。	－	ファイル選択	最大サイズ 4G バイト
		パスフレーズ	ssh 秘密鍵ファイルにパスフレーズが設定されている場合、パスフレーズを入力します。	－	手動入力	最大長 256 バイト
Ansible 利用情報	Legacy/Role 利用情報	認証方式	Ansible・Ansible Automation Controller から機器へ接続する際の認証方式を選択します。 ●パスワード認証 ログインパスワードの管理で●の選択と、ログインパスワードの入力が必須です。 ●鍵認証（パスフレーズなし） ssh 秘密鍵ファイル(id_ras)のアップロードが必須です。 ●鍵認証（パスフレーズあり） ssh 秘密鍵ファイル(id_ras)のアップロードと、パスフレーズの入力が必須です。 ●鍵認証（鍵交換済み）※1 ssh 秘密鍵ファイル(id_ras)のアップロードは必要ありません。 ●パスワード認証（winrm) 必要に応じて WinRM 接続情報を入力します。 尚、パスワード認証（winrm)以外の認証方式の場合、機器側に以下の設定が必要です。 ログインユーザの sudo 権限を NOPASSWD 付で /etc/sudoers に設定する。 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	WindowsServer に WinRM 接続する際のポート番号を入力します。 未入力の場合はデフォルト(5985)での WinRM 接続となります。	－	手動入力	説明欄記載のとおり
		WinRM 接続情報 サーバ証明書	WinRM 接続ポートで https のポート番号を指定した場合にサーバ証明書を入力します。 アップロードしたファイルは暗号化されて保存されます。※登録後はダウンロー不可となります。 サーバ証明書の認証を省く場合、インベントリファイル追加オプションに下記を追記して下さい。 ansible_winrm_server_cert_validation: ignore	－	ファイル選択	最大サイズ 4G バイト
	Pioneer 利用情報	プロトコル	Pioneer から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。 ●ssh を選択した場合 認証方式はパスワード認証（winrm)以外を選択してください。	○	リスト選択	-

項目			説明	入力必須	入力形式	制約事項
			●telnet を選択した場合 認証方式に設定した値は使用せずに telnet で接続します。			
		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	-
		LANG	Pioneer の対話ファイルを実行するユーザーの LANG を選択します。 空白の場合は utf-8 扱いとなります。	-	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
	インベントリファイル追加オプション 「Pioneer 利用時は、本項目は適用されません。」		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: ios ansible_become: yes ansible_become_method: enable 各パラメータ値を変数で記述することも出来ます。 ansible_become_password: **‘{{ VAR_passwd }}’** 具体値に変数を記述する場合 '{{△**VAR_passwd**△}}' △:半角スペース ‘:シングル・ダブルコーテーションで囲む「必須」 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」メニューから登録します。	-	手動入力	最大長 512 バイト
	Ansible Automation Controller 利用情報	インスタンスグループ名※2	Ansible Automation Controller がクラスタ構成の場合、どのインスタンスグループで実行するかを選択します。ここで設定した、インスタンスグループはインベントリオブジェクトに設定されます。 未選択の場合は Ansible Automation Controller のデフォルトのインスタンスグループになります。 Ansible Automation Controller がクラスタ構成でない場合は、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	Ansible Automation Controller 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_cnnection 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。 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_cnnection 以外)を設定する必要があります。 Exp) インベントリファイル追加オプションの設定例 Network OS が ios の場合の設定値 ansible_network_os: ios ansible_become: yes ansible_become_method: enable Ansible Automation Controller の認証情報の接続タイプについては、ドキュメント 認証情報タイプ を参照して下さい。 Network OS と ansible_connection の関連や Platform Options については、Ansible ドキュメント Platform Options を参照ください。	○	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　認証方式が鍵認証（鍵交換済み）に設定する為に必要な公開鍵ファイルの配布

・Ansible Core の場合

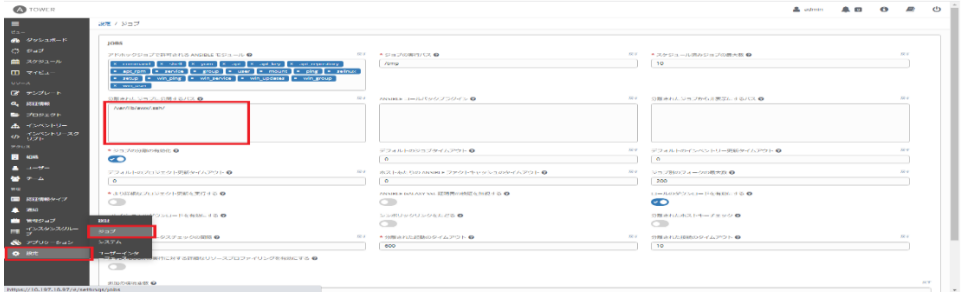
ansible がインストールされているサーバーの実行ユーザー「Ansible 共通コンソール=>インターフェース情報に設定されている実行ユーザー」から作業対象ホストに ssh 接続します。

実行ユーザーの公開鍵ファイルをログイン先ユーザーの authorized_keys にコピーして下さい。

・Ansible Automation Controller の場合

Ansible Automation Controller の awx ユーザーから作業対象ホストに ssh 接続しています。

awx ユーザーの公開鍵ファイルをログイン先ユーザーの authorized_keys にコピーして下さい。ブラウザより Ansible Automation Controller にログインし、「設定」→「ジョブ」→「分離されたジョブに公開するパス」に「/var/lib/awx/.ssh/」を設定します。



尚、AnsibleTower4.x 以降、awx ユーザーの.ssh ディレクトリが扱えない為、作業対象ホストと鍵認証（鍵交換済み）での接続は出来ません。

※2　Ansible driver のバックヤード機能「Ansible Automation Controller データ同期」により取得したデータから選択します。

5.1.2 オペレーション一覧

(1) [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。

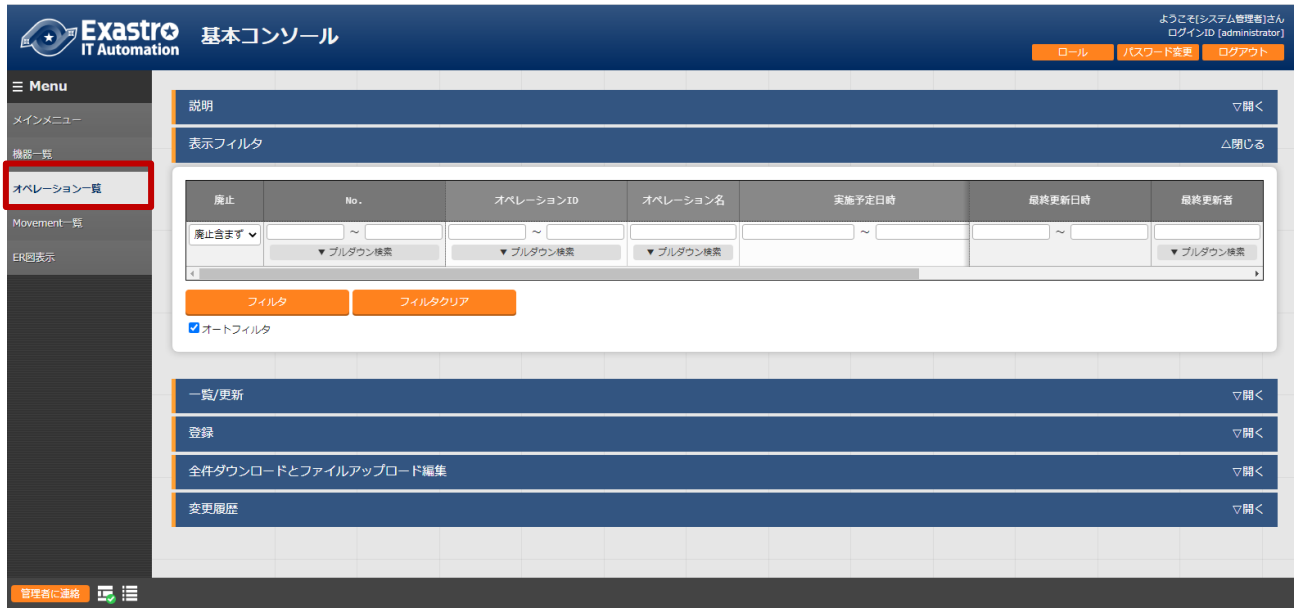


図 5.1-4 サブメニュー画面（オペレーション一覧）

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

(1) [インターフェース情報]では、Ansible Core、AnsibleTower、Ansible Automation Controller のいずれかを実行エンジンに選択し、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の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。

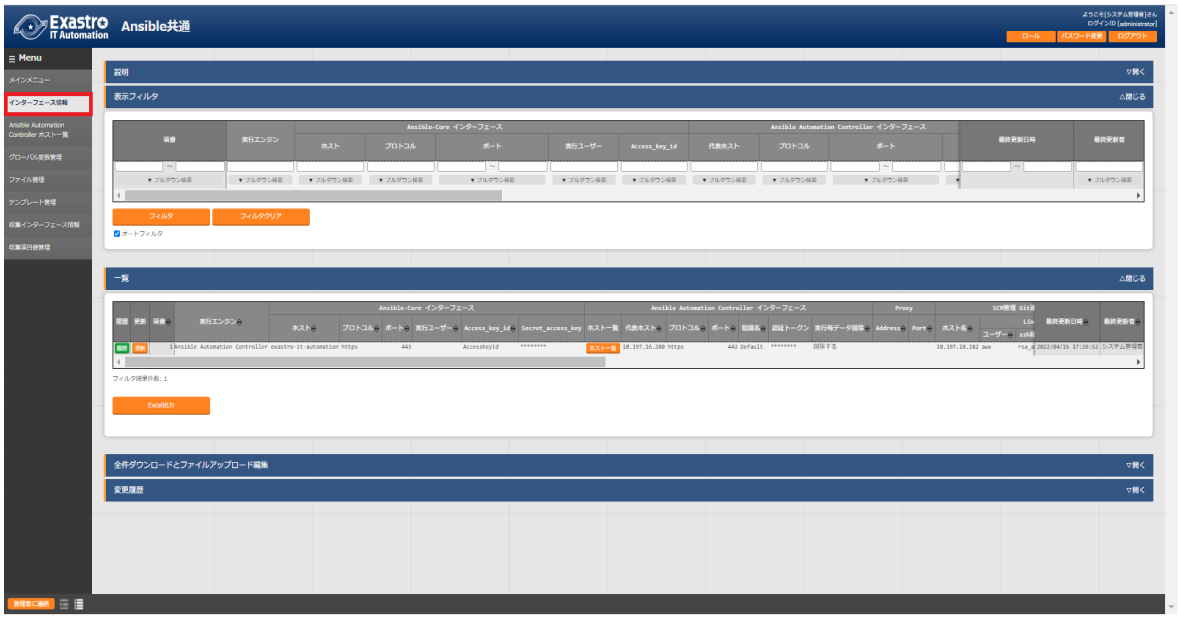


図 5.2-1 サブメニュー画面（インタフェース情報）

(2) 「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。

図 5.2-2 登録画面（インタフェース情報）

(3) 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		実行するエンジンを下記の 3 種類から選択します。 ・Ansible Core ・Ansible Tower ・Ansible Automation Controller	○	リスト選択	
Ansible Core インターフェース	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 実行エンジンが Ansible Core 以外でも ansible-vault コマンドを実行する Ansible Core インターフェースの設定が必要です。
	プロトコル	Ansible 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 Ansible Core 以外でも ansible-vault コマンドを実行する Ansible Core インターフェースの設定が必要です。
	ポート	Ansible 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 Ansible Core 以外でも ansible-vault コマンドを実行する Ansible Core インターフェースの設定が必要です。
	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 実行エンジンが Ansible Core 以外でも ansible-vault コマンドを実行する Ansible Core インターフェースの設定が必要です。
	ACCESS_KEY_ID	Ansible サーバと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 実行エンジンが Ansible Core 以外でも ansible-vault コマンドを実行する Ansible Core インターフェースの設定が必要です。
	SECRET_ACCESS_KEY	Ansible サーバと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 実行エンジンが Ansible Core 以外でも ansible-vault コマンドを実行する Ansible

項目			説明	入力必須	入力形式	制約事項
						Core インターフェースの設定が必要です。
Ansible Automation Controller インターフェース	代表ホスト		Ansible Automation Controller 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 Ansible Automation Controller を選択します。	-	リスト選択	実行エンジンが Ansible Core 以外の場合に入力必須
	プロトコル		Ansible Automation Controller 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 Ansible Core 以外の場合に入力必須
	ポート		Ansible Automation Controller 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 Ansible Core 以外の場合に入力必須
	組織名		Ansible Automation Controller 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 Ansible Core 以外の場合に入力必須
	認証トークン		ITA から Ansible Automation Controller 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 Ansible Core 以外の場合に入力必須
	実行時データ削除		作業実行時にITAとAnsible Automation Controller 内に一時的に生成したデータリソースを作業終了後に削除するかを選択します。 「削除する」を選択した場合に削除されるデータリソースは「 8.5実行時データ削除で削除されるデータリソース」を参照して下さい。	-	リスト選択	実行エンジンが Ansible Core 以外の場合に入力必須
Proxy	Adress		プロキシサーバのアドレスを入力します。 ITAがプロキシ環境下にある場合、Ansible/ Ansible Automation Controllerサーバまでの疎通のために設定が必要な場合があります。 プロキシサーバのURLが http://procy.gate.co.jp:8080の場合 Addressにはhttp://procy.gate.co.jp を入力します。 Portには 8080を入力します。	-	手動入力	最大 128 バイト
	Port		プロキシサーバのポートを入力します。 ITAがプロキシ環境下にある場合、Ansible/ Ansible Automation Controllerサーバまでの疎通のために設定が必要な場合があります。	-	手動入力	
SCM 管理 Git 連携先情報	ホスト名		Ansible Automation Controller から連携先の Git リポジトリへ ssh プロトコルで接続する為のホスト名(または IP アドレス)を入力します。 Ansible driverのバックヤード機能がインストールされているホストにAnsible Automation Controllerと連携するGitリポジトリが作成されます。	-	手動入力	最大長 128 バイト 実行エンジンが Ansible Automation Controller の場合に必須入力です。
	Linux	ユーザー	Ansible Automation Controllerから連携先のGitリポジトリへsshプロトコルで接続する為のユーザーを入力します。	-	手動入力	最大長 128 バイト 実行エンジンが Ansible Automation Controller

項目			説明	入力必須	入力形式	制約事項
						の場合に必須入力です。
		ssh 秘密鍵ファイル	Ansible Automation Controllerから連携先のGitリポジトリへsshプロトコルの鍵認証で接続する為の秘密鍵ファイルを入力します。	-	ファイル	ファイル選択 最大サイズ 4G バイト 実行エンジンが Ansible Automation Controller の場合に必須入力です。
		パスフレーズ	ssh秘密鍵ファイルに設定されているパスフレーズを入力します。	-	手動入力	最大長 256 バイト
データリレイストレージパス(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
データリレイストレージパス (Ansible)			Ansible サーバー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 実行エンジンが Ansible Core 以外では使用しない情報です。
Symphony インスタンスデータリレイストレージパス (Ansible)			Symphony 実行時、各 Movement で共有するディレクトリです。Ansible サーバーから見た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 実行エンジンが Ansible Core 以外では使用ない情報です。
Conductor インスタンスデータリレイストレージパス (Ansible)			Conductor 実行時、各 Movement で共有するディレクトリです。Ansible サーバーから見た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 実行エンジンが Ansible Core 以外では使用ない情報です。
オプションパラメータ			Movement 共通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 Core の場合は ansible-playbook コマンドのオプションパラメータ、実行エンジンが Ansible Core 以外の場合はジョブテンプレートのパラメータを入力します。 Movement 固有のオプションパラメータは Movement 一覧で入力します。 オプションパラメータの詳細については、8.3 オプションパラメーター一覧」覧を参照してください。	-	手動入力	最大長 512 バイト
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

項目	説明	入力必須	入力形式	制約事項
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。 ステータスが[未実行]、[準備中]、[実行中]、[実行中(遅延)]の場合、指定した行数でログを出力します。 ステータスが[完了]、[完了(異常)]、[想定外エラー]、[緊急停止]、[未実行(予約)]、[予約取消]の場合、指定した行数ではなくすべてのログを出力します。 環境毎にチューニングを要しますが、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の具体値が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。	○	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、ITA と Ansible が別サーバで運用する構成もサポートするので、ディレクトリパスを別々に管理します。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 Automation Controller ホスト一覧

[Ansible Automation Controller ホスト一覧]では、Ansible Automation Controller の RestAPI 実行に必要な情報、および構築資材を Ansible Automation Controller 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。
クラスタ構成の場合は、構成している全てのサーバを登録して下さい。
ただし、Ansible Automation Controller の hop node の登録は不要です。

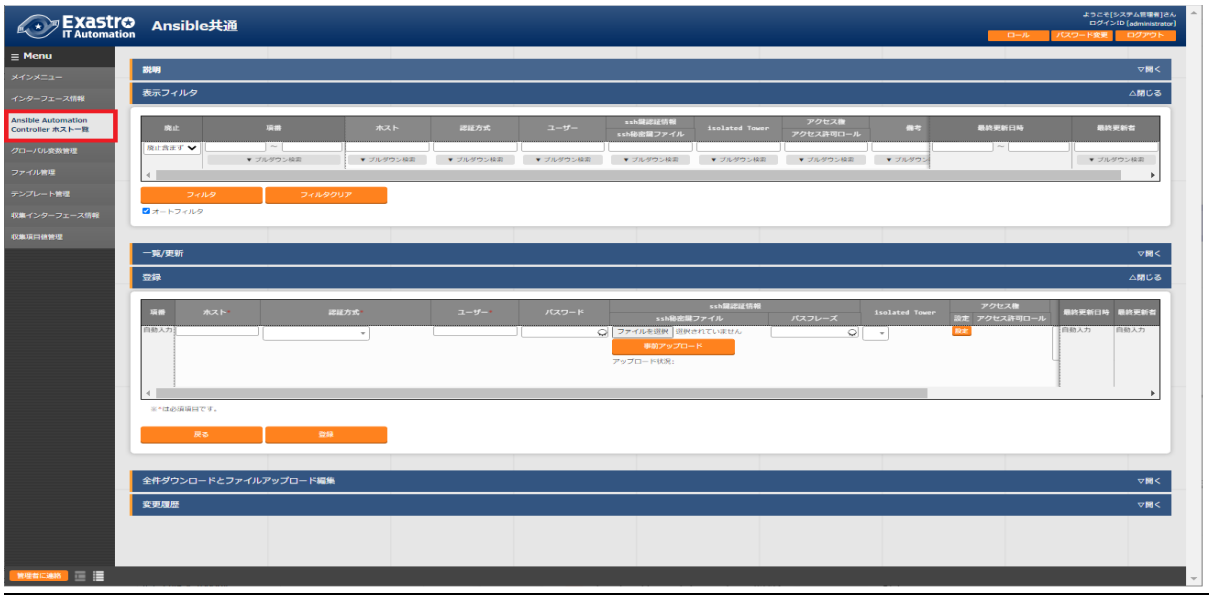


図 5.2-3 サブメニュー画面（Ansible Automation Controller ホスト一覧）

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

図 5.2-4 登録画面（Ansible Automation Controller ホスト一覧）

(2) Ansible Automation Controller 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 Automation Controller ホスト一覧）

項目	説明	入力必須	入力形式	制約事項
ホスト	Ansible Automation Controller 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	Ansible Automation Controller 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。 ●パスワード認証	○	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
		ログインユーザとパスワードの入力が必須です。 ●鍵認証（パスフレーズなし） ssh 秘密鍵ファイル(id_ras)のアップロードが必須です。 ●鍵認証（パスフレーズあり） ssh 秘密鍵ファイル(id_ras)のアップロードと、パスフレーズの入力が必須です。 ●鍵認証（鍵交換済み）※1 ssh 秘密鍵ファイル(id_ras)のアップロードは必要ありません。			
ログインユーザ		Ansible Automation Controller 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。 ログインユーザは、Ansible Automation Controller 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、使用することを強く推奨します。	○	手動入力	最大長 30 バイト
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。	ー	手動入力	最大長 30 バイト
ssh 鍵認証情報	ssh 秘密鍵ファイル	ssh秘密鍵ファイルを指定して鍵認証する場合の秘密鍵ファイルを入力します。 アップロードしたファイルは暗号化されて保存されます。※登録後はダウンロー不可となります。	-	ファイル選択	最大サイズ 4G バイト
	パスフレーズ	ssh 秘密鍵ファイルにパスフレーズが設定されている場合、パスフレーズを入力します。	-	手動入力	最大長 256 バイト
isolated Tower		クラスタ構成で構築されている場合で、対象ノードが Ansible Tower の isolated node ／ Ansible Automation Controller の execution node の場合、「●」を選択します。	ー	選択方式	
備考		自由記述欄です。	ー	手動入力	最大長 4000 バイト

※1　認証方式が鍵認証（鍵交換済み）に設定する為に必要な公開鍵ファイルの配布
ITA がインストールされているサーバーの root から Ansible Automation Controller の awx ユーザーで ssh 接続します。
root の公開鍵ファイルを Ansible Automation Controller の awx ユーザーの authorized_keys にコピーして下さい。

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利用するグローバル変数名を登録／更新／廃止を行います。

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。

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 の対話ファイル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テンプレート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 ファイルをテンプレート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。

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
ﾃﾝﾌﾟﾚｰﾄ 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
ﾃﾝﾌﾟﾚｰﾄ素材	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 最大サイズ 4G バイト
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のす。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。

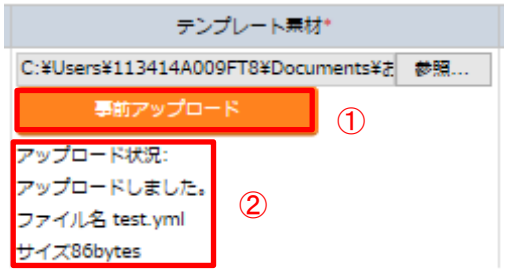


表 5.2-5 変数定義の留意事項

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.)

	VAR_sample_1: - none VAR_sample_2: []
多段変数	階層化された変数構造が定義出来ます。 e.g.) VAR_sample_1: - item1: none item2: VAR_sample_2: - array: - item1: none item2: 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。 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。
グローバル変数	具体値は省略可能です e.g.) GBL_sample_1: none GBL_sample_2:
ITA 独自変数	変数の定義は不要です。
読替変数	定義出来る変数の種類は以下の 3 種類です。 ・通常変数 ・複数具体値変数 ・多段変数 各変数定義の留意事項は同様です。 e.g.) LCA_sample_1: LCA_sample_2: [] LCA_sample_3: - item1: none item2: 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。

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。

※ファイル名にスペースなどが含まれている場合、適切にクォーテーションで囲まないと、作業実行時にエラーになる場合があります。

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、登録したﾃﾝﾌﾟ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※1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。

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。

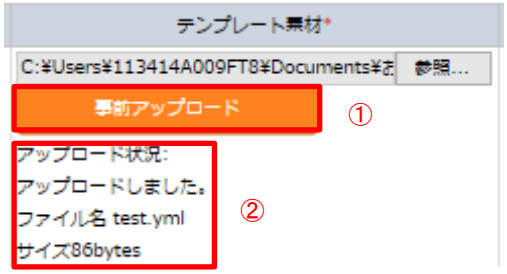
図 5.2-10 登録画面（ファイル管理）

(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 4G 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。

※ファイル名にスペースなどが含まれている場合、適切にクォーテーションで囲まないと、作業実行時にエラーになる場合があります。

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。

5.2.6 収集インターフェース情報

[収集インターフェース情報]では、収集機能で利用するITAの標準RESTAPIを利用する為、RESTAPIアクセス用の接続インターフェース情報の更新を行います。

詳細は、別紙「Exastro-ITA_利用手順マニュアル_収集機能」を参照。

5.2.7 収集項目値管理

[収集項目値管理]では、収集対象項目とパラメータシートの項目の紐付設定を行います。

詳細は、別紙「Exastro-ITA_利用手順マニュアル_収集機能」を参照。

① 一覧/更新のメニューID または、メニュー名称のリンクをクリックすると、対象メニューへ遷移する。

図 5.2-13 サブメニュー画面（収集項目値管理）

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。

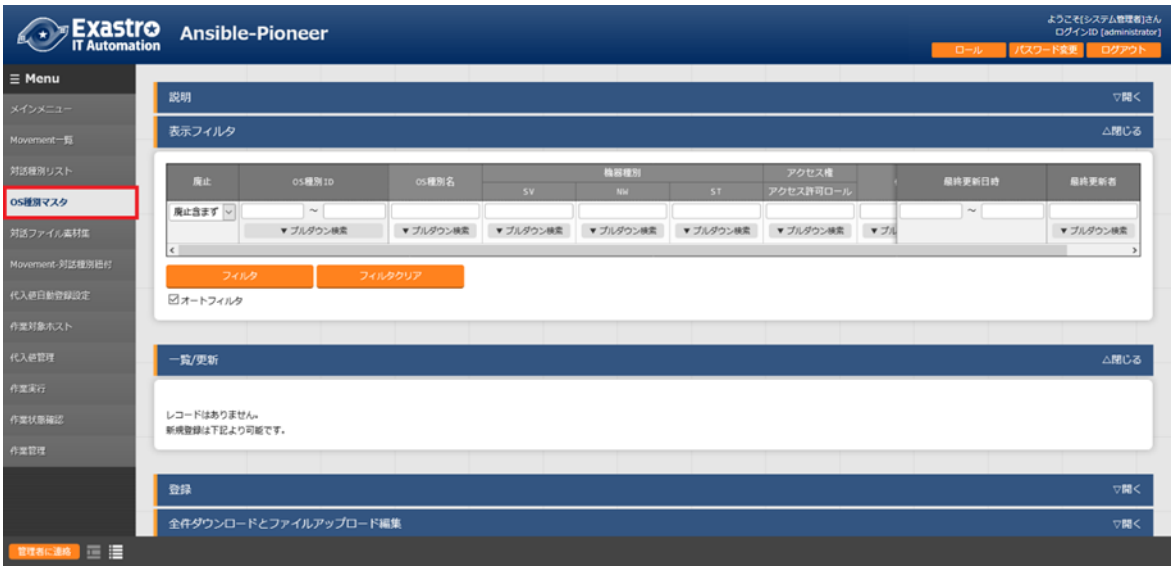


図 5.3-1 サブメニュー画面（OS種別マスタ）

(2) 「登録」-「登録開始」ボタンより、OS 情報の登録を行います。

図 5.3-2 登録画面（OS種別マスタ）

(3) 一覧/更新の対話素材ファイル集ボタンをクリックすると、対象の 5.3.6 対話素材ファイル集へ遷移する。

図 5.3-3 サブメニュー画面（OS種別マスタ）

(4) 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。

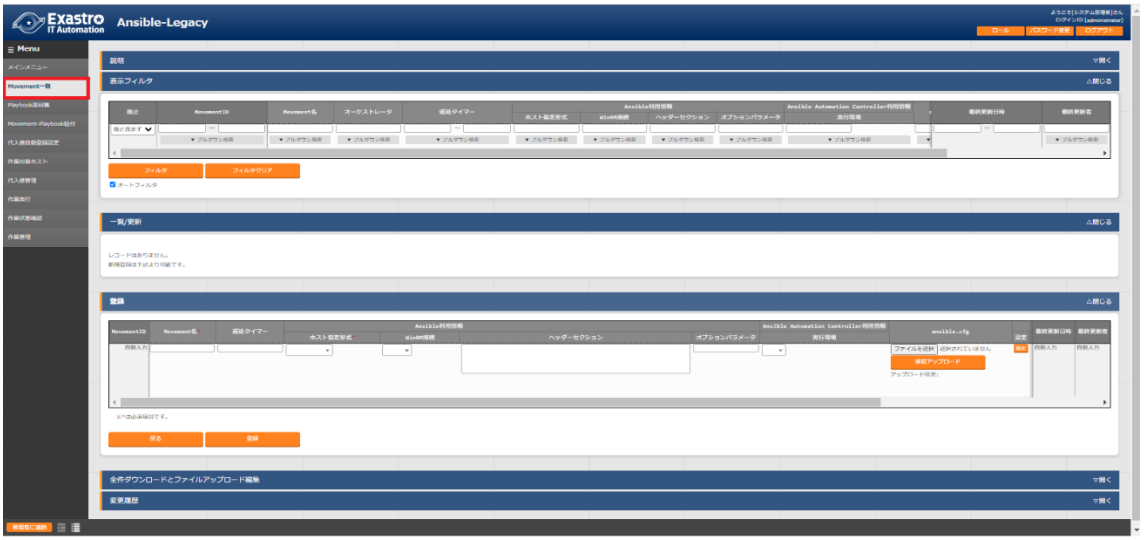


図 5.3-4 サブメニュー画面（Movement 一覧）

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

図 5.3-5 登録画面（Movement 一覧）

(3) Movement-Playbook 紐付(Movement-対話種別紐付、Movement-ロール紐付)ボタンをクリックすると、対象の 5.3.7 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）へ遷移する。

図 5.3-6 サブメニュー画面（Movement 一覧）

(4) 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	-	手動入力	-

	す。(単位:分) 未入力の場合は警告表示しません。			
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	-
virtualenv	Ansible サーバー内に virtualenv で構築した ansible 実行環境で作業実行したい場合、virtualenv のフルパスを入力して下さい。 virtualenv 環境が存在しなかったり virtualenv 環境内に ansible がインストールされていない場合、作業実行時に「想定外エラー」となります。 未入力の場合は ansible サーバーにインストールされている ansible 環境で作業実行を行います。 Exp) /virtualenv 配下に ansible11.1.0 という名前で virtualenv 環境を作成した場合の virtualenv のフルパスは下記となります。 virtualenv 環境の作成 cd /virtualenv virtualenv ansible11.1.0 virtualenv のフルパス /virtualenv/ansible11.1.0	-	手動入力	最大長 512 バイト
並列実行数 ※ Pioneer の Movement 一覧でのみ表示されます。	Ansible-playbook コマンドのオプションパラメータ「forks」を入力します。 ■未入力時の振る舞いについて 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。	-	手動入力	NULL または整数
WinRM 接続	対象ホストが WindowsServer 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 Automation Controller - 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 ※ Pioneer の Movement 一覧で	Movement 固有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 Core の場合は ansible-playbook コマンドのオプションパラメータ、実行エンジンが Ansible Core 以外の場合はジョブテンプレートのパラ	-	手動入力	最大長 512 バイト

は表示されません。	メータを入力します。 オプションパラメータの詳細については、8.3 オプションパラメーター覧」覧を参照してください。			
Virtualenv ※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 Ansible Automation Controller インストール時にインストールされた仮想環境を使用します。	-	リスト選択	
実行環境 ※実行エンジンがAnsible Automation controller の場合に表示されます。	Ansible Automation Controller サーバに構築されている実行環境が表示されています。 作業実行する実行環境を選択します。 未選択の場合は、デフォルト「Default execution environment」が使用されます。	-	リスト選択	
Ansible.cfg	作業実行時に使用するansible.cfgをアップロードします。 未アップロードの場合は、デフォルトが使用されます。 また、ロールパッケージ管理でアップロードされている zip ファイルに ansible.cfg が含まれている場合は、この項目でアップロードした ansible.cfg で上書きされます。	-	ファイル選択	最大サイズ 4G バイト
備考	自由記述欄です。	-	手動入力	最大長 4000 バイト

【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 Playbook 素材集（Ansible-Legacy のみ）

(1) [Playbook 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。

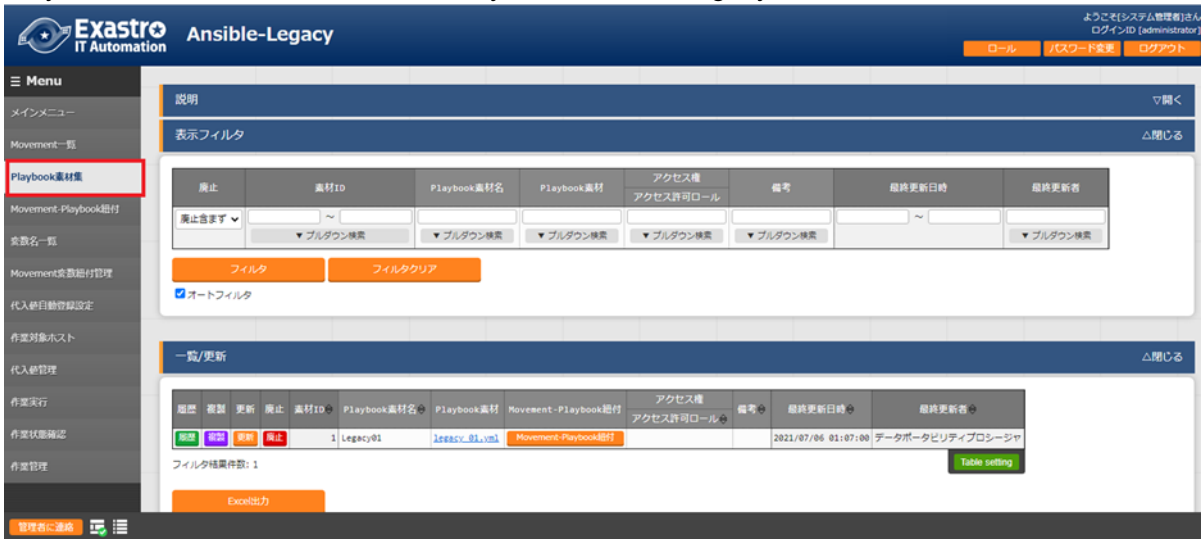


図 5.3-7 サブメニュー画面（Playbook 素材集）

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

図 5.3-8 登録画面（Playbook 素材集）

(3) Movement-Playbook 紐付ボタンをクリックすると、対象の 5.3.7 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）へ遷移する。

図 5.3-9 サブメニュー画面（Playbook 素材集）

(4) 登録画面の項目一覧は以下のとおりです。

表 5.3-3 登録画面項目一覧（Playbook 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	○	手動入力	最大長 256 バイト

	を入力します。			
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 4G 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抜出します。抜出した変数は、「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。
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、「5.3.12 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 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

5.3.4 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「ロールパッケージ（Ansible-Legacy Role）の記述」を参照してください。

図 5.3-10 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。

図 5.3-11 登録画面（ロールパッケージ管理）

(3) 一覧/更新の Movement-ロール紐付ボタンをクリックすると、5.3.7 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）へ遷移します。

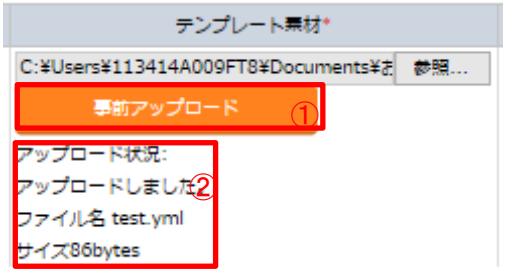
図 5.3-12 サブメニュー画面（ロールパッケージ管理）

(4) 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 4G 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。
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 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。

図 5.3-13 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。

図 5.3-14 登録画面（対話種別リスト）

(3) 一覧/更新の Movement-対話種別紐付ボタンをクリックすると、対象の 5.3.7 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）へ遷移します。対話素材ファイル集ボタンをクリックすると、対象の 5.3.6 対話素材ファイル集へ遷移します。

図 5.3-15 サブメニュー画面（対話種別リスト）

(4) 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 5.3-16 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 5.3-17 登録画面（対話ファイル素材集）

(4) 一覧/更新の対話種別のリンクをクリックすると、対象の 5.3.5 対話種別リストへ遷移する。
また、os 種別のリンクをクリックすると、対象の 5.3.1 OS 種別マスタへ遷移する。

図 5.3-18 サブメニュー画面（対話ファイル素材集）

(5) 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 4G 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。

対話ファイルは YAML 形式のファイルとして扱います。YAML 形式に準じていない記述があると、アップロード時にエラーとなります。詳しくは、6.2 章の「(7) 対話ファイルを yaml 形式で記載する際の注意事項」を参照して下さい。

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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）**

※メニュー名は Ansible-Legacy は Movement-Playbook 紐付、Ansible-Pioneer は Movement-対話種別紐付、Ansible-LegacyRole は Movement-ロール紐付です。

(1) Movement で実行する素材の登録／更新／廃止を行います。

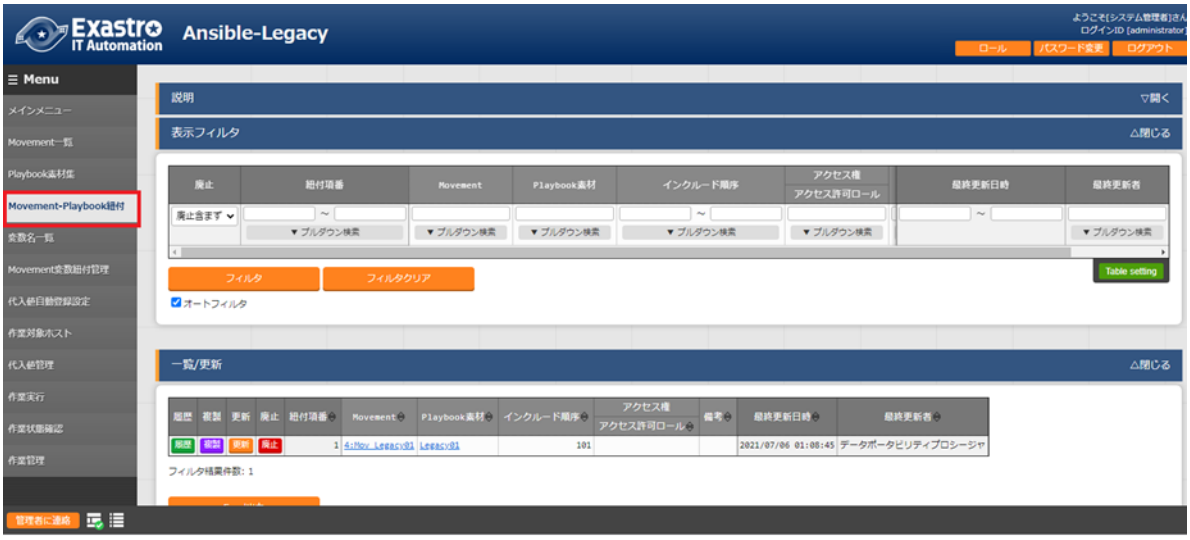


図 5.3-19 サブメニュー画面（Movement-Playbook 紐付）
※画面は Ansible-Legacy のものです。

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

図 5.3-20 登録画面（Movement-Playbook 紐付）

(3) Movement のリンクをクリックすると、対象の 5.3.2.Movement 一覧へ遷移する。
また、Playbook 素材のリンクをクリックすると、5.3.3.Playbook 素材集へ遷移する。
※Movement-対話種別紐付（Ansible-Pioneer）の場合は、Movement と対話種別、Movement-ロール紐付（Ansible-LegacyRole）の場合は、Movement とロールパッケージ名がそれぞれリンクになっています。

図 5.3-21 サブメニュー画面（Movement-Playbook 紐付）

(4) 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-Playbook 紐付の場合）

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	「5.3.3Playbook 素材集（Ansible-Legacy のみ）」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-ロール紐付の場合）

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。 同一 **Movement** に複数のロールパッケージは登録出来ません。	○	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

- **Ansible-Pioneer の場合**

表 5.3-9 登録画面項目一覧（Movement-対話種別紐付の場合）

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	「対話種別リスト（Ansible-Pioneer のみ）」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 変数ネスト管理（Ansible-Legacy Role のみ）

(1) [変数ネスト管理]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。

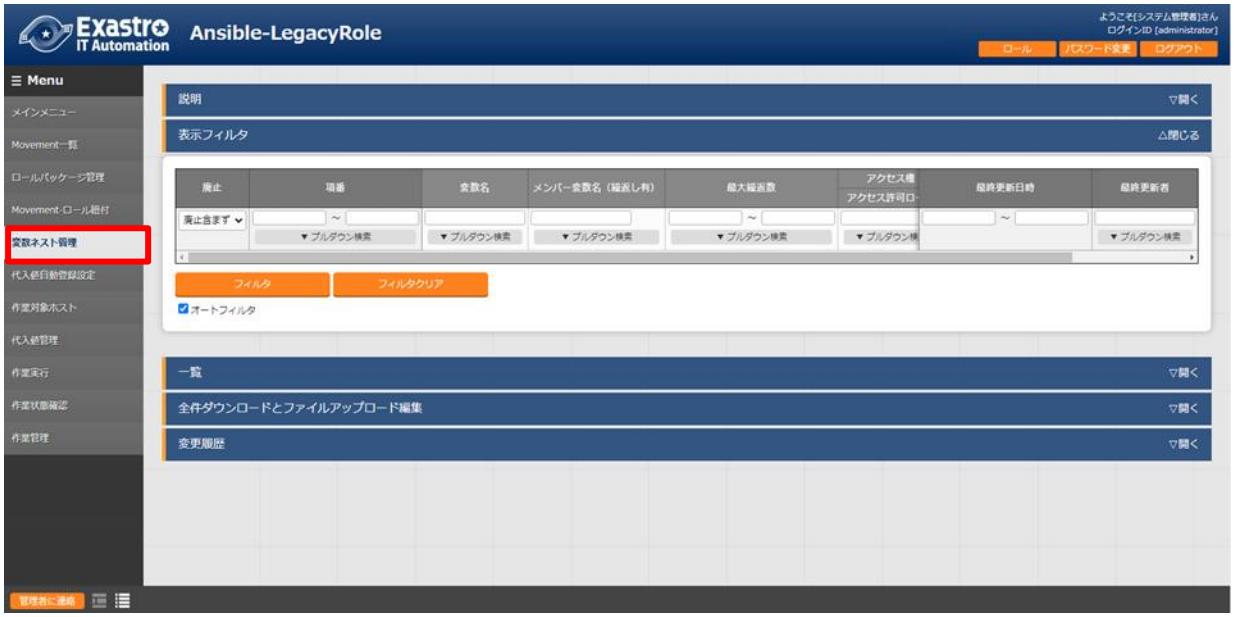


図 5.3-22 サブメニュー画面（変数ネスト管理）

(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）

図 5.3-23 登録画面（変数ネスト管理）

(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（変数ネスト管理）

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		
nested:	nested	2
- craete_users:		
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、変数ネスト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※1 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

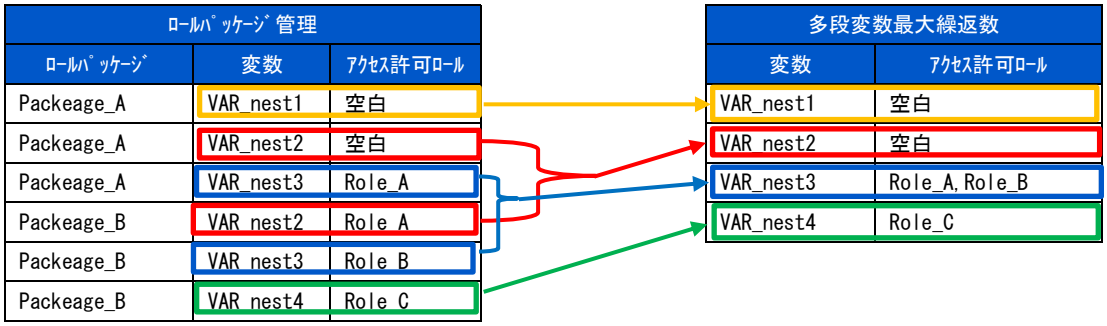
(4) 変数名の一意管理

変数名の管理は、全ロールパッケージで一意管理しています。ロールパッケージを跨って同じ変数名を使用している場合、変数ネスト管理で設定した繰返し数は、全ロールパッケージの変数に適用されます。

(5) アクセス許可ロール

変数ネスト管理に設定されるアクセス許可ロールは、該当の変数が定義されているロールパッケージ管理のアクセス許可ロールが設定されます。複数のロールパッケージ管理で変数が定義されている場合、各ロールパッケージ管理のアクセス許可ロールが全て設定されます。また、アクセス許可ロールが空の場合、全てのロールへのアクセスが可能として扱われます。各ロールパッケージ管理のアクセス許可ロールが空白の場合、変数ネスト管理のアクセス許可ロールも空白が設定されます。

アクセス許可ロールの詳細については「Exastro-ITA_利用手順マニュアル_データレコード毎のロールベースアクセス制御」を参照して下さい。



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。

図 5.3-24 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。

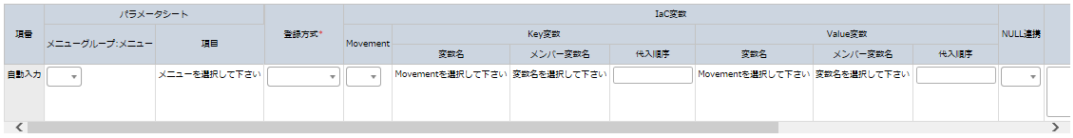


図 5.3-25 登録画面（代入値自動登録設定）

(3) 一覧/更新のメニューID または、メニュー名称のリンクをクリックすると、対象メニューへ遷移します。

図 5.3-26 サブメニュー画面（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	ー	▲	ー
	代入順序	△	△	△
Value 情報	変数名	○	○	○
	メンバー変数名	ー	▲	ー
	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

ー:非表示

(4) 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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		パラメータシートの具体値が 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「NULL 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) パラメータシートの変数項目

「Ansible 共通:テンプレート管理:テンプレート変数名」「Ansible 共通:ファイル管理:ファイル埋込変数名」をパラメータシートの項目として代入値自動登録設定で使用した場合、項目の設定値は選択されている変数名となり紐付ている変数の具体値は‘{{ 変数名 }}’として代入値管理に反映されます。
ファイルパスはITAが作業実行時に配置したファイルパスをホスト変数経由でansibleが解釈し処理します。

パラメータシート定義

パラメータシート

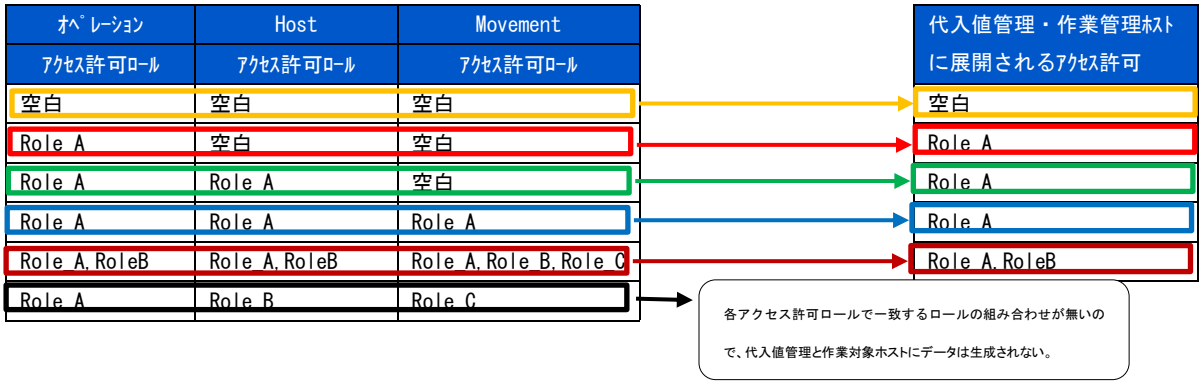
代入値自動登録設定

代入値登録

(6) アクセス許可ロール

代入値自動登録の情報から生成される代入値管理と作業対象ホストに設定されるアクセス許可ロールは、代入値自動登録に設定されている Movement とパラメータシートに設定されているホスト(機器一覧)とオペレーションの各アクセス許可ロールで一致しているロールが設定されます。また、アクセス許可ロールが空の場合、全てのロールへのアクセスが可能として扱われます。各アクセス許可ロールで一致するロールの組み合わせが無いデータは、代入値管理と作業対象ホストにデータが生成されません。

アクセス許可ロールの詳細については「Exastro-ITA_利用手順マニュアル_データレコード毎のロールベースアクセス制御」を参照して下さい。



(7) パラメータシート(縦メニュー)

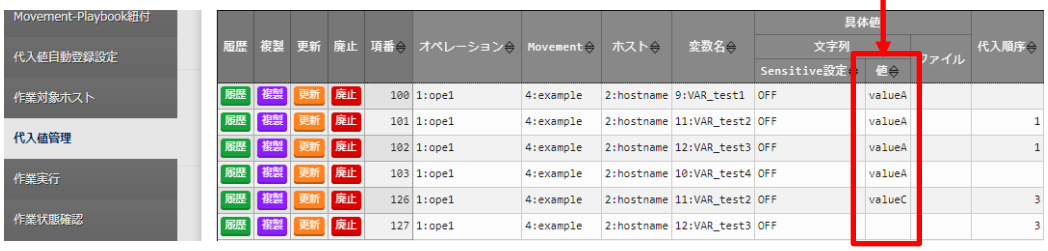
パラメータシート(縦メニュー)を使用する際に、リピート設定されている項目において、
まとまり単位で値が空の場合、代入値自動登録設定で NULL 連携を有効にしていても、
代入値管理に登録はされません。

メニュー作成

パラメータシート

代入値自動登録設定

代入値管理



代入値自動登録設定で NULL 連携「有効」にしているが、

valueB の行はリピート項目全て空なので代入値管理に登録されない

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。

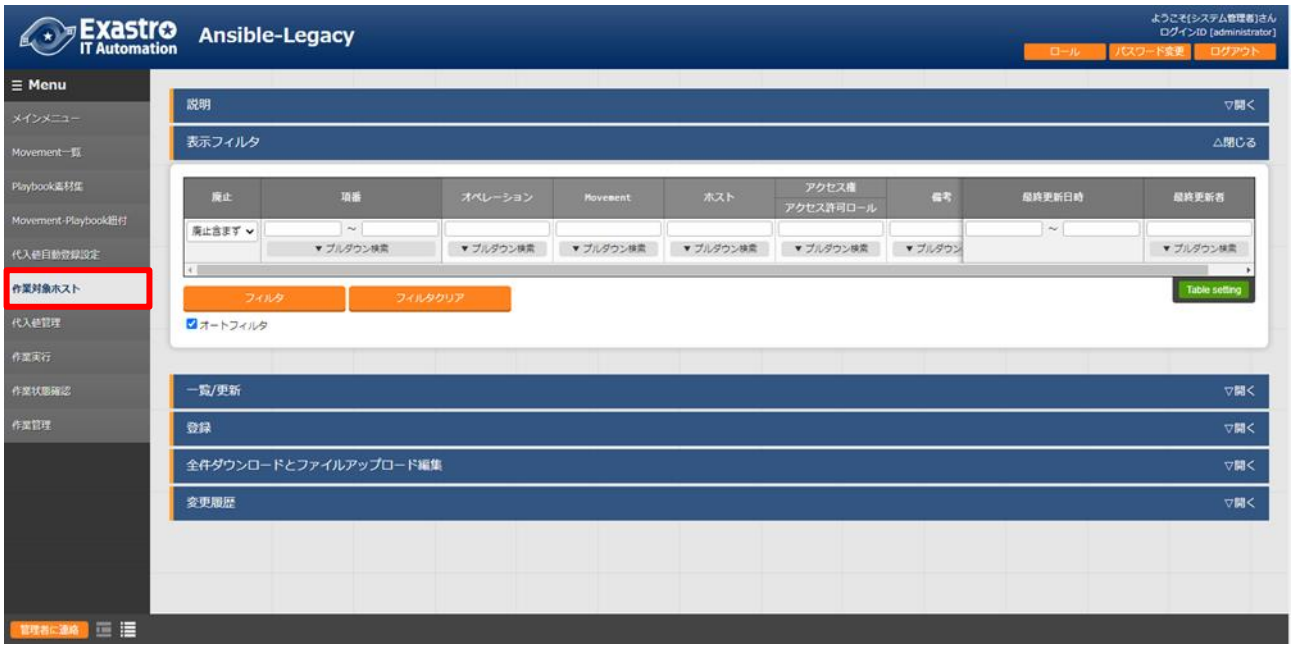


図 5.3-27 サブメニュー画面（作業対象ホスト）

(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 5.3-28 登録画面（作業対象ホスト）

(3) 一覧/更新の Movement のリンクをクリックすると、対象の 5.3.7 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）へ遷移します。
また、代入値管理ボタンをクリックすると、対象の 5.4.11 代入値管理へ遷移します。

図 5.3-29 サブメニュー画面（作業対象ホスト）

(4) 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	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。

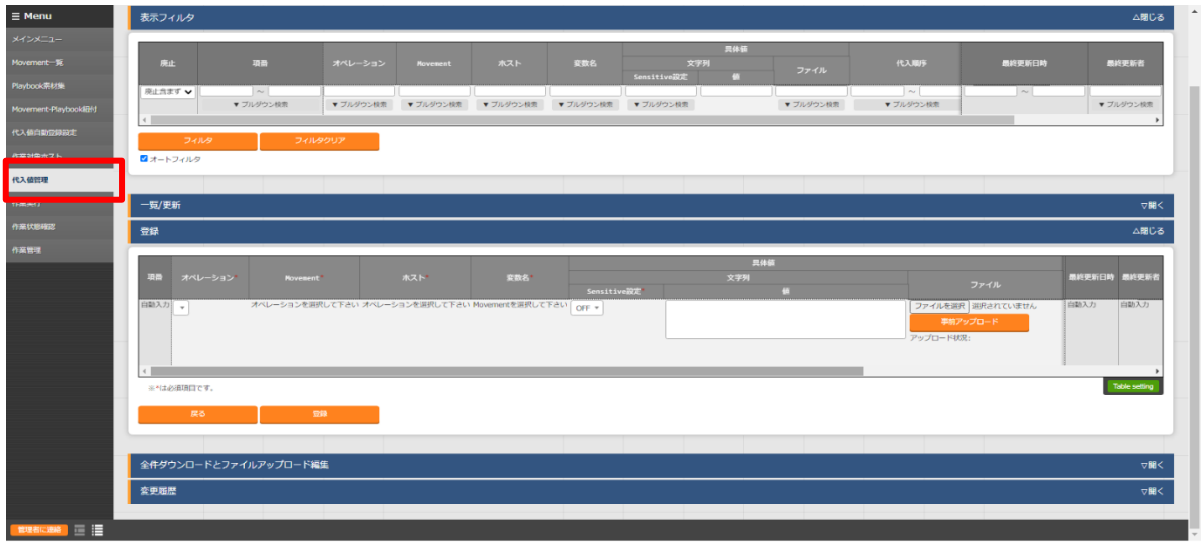


図 5.3-30 サブメニュー画面（代入値管理）
※画面は Ansible-Legacy Role のものです。

(2) 「登録」-「登録開始」ボタンより代入値管理を行います。

図 5.3-31 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	ー	▲	ー
代入順序	△	△	△
デフォルト値(表示のみ)	ー	○	ー

○: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
ー: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
ｵﾍﾟﾚｰｼｮﾝ			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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）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	文字列	Sensitive 設定	「OFF」または「ON」を選択します。 「ON」を選択した場合、具体値を暗号化しITA上で表示させないようにします。 ・Legacy/Legacy-Role の場合 ansible に渡すホスト変数ファイルには、ansible-vault で暗号化された内容が設定されます。 ・Pioneer の場合 ansible に渡すホスト変数ファイルには、ITA 独自で暗号化した内容が設定されます。	ー	ボタン選択	
		値※1	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	ー	手動入力	最大長 8192 バイト

			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。 Exp) 　具体値に TPF_sample を入力する場合 　'{{△TPF_sample△}}' 　　△:半角スペース 　　':シングル・ダブルコーテーションで囲む「必須」 尚、 具体値に値とファイルの両方を設定することは出来ません。Sensitive 設定 ON で設定されている値をクリアしたい場合、 Sensitive 設定を OFF にして更新して下さい。			
	ファイル		オペレーション/Movement/ホストで使用する変数にファイルを埋込む場合に、埋込むファイルをアップロードします。	ー	手動入力	最大サイズ 4G 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「6.4 ITAreadme（Ansible-Legacy Role のみ）の記述」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

※1 具体値にファイル埋込変数「CPF_」・テンプレート埋込変数「TPF_」を設定する場合、 Sensitive 設定は「OFF」に設定して下さい。 Sensitive 設定が「ON」の場合、各変数として扱われません。

【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。多段変数も同様で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。

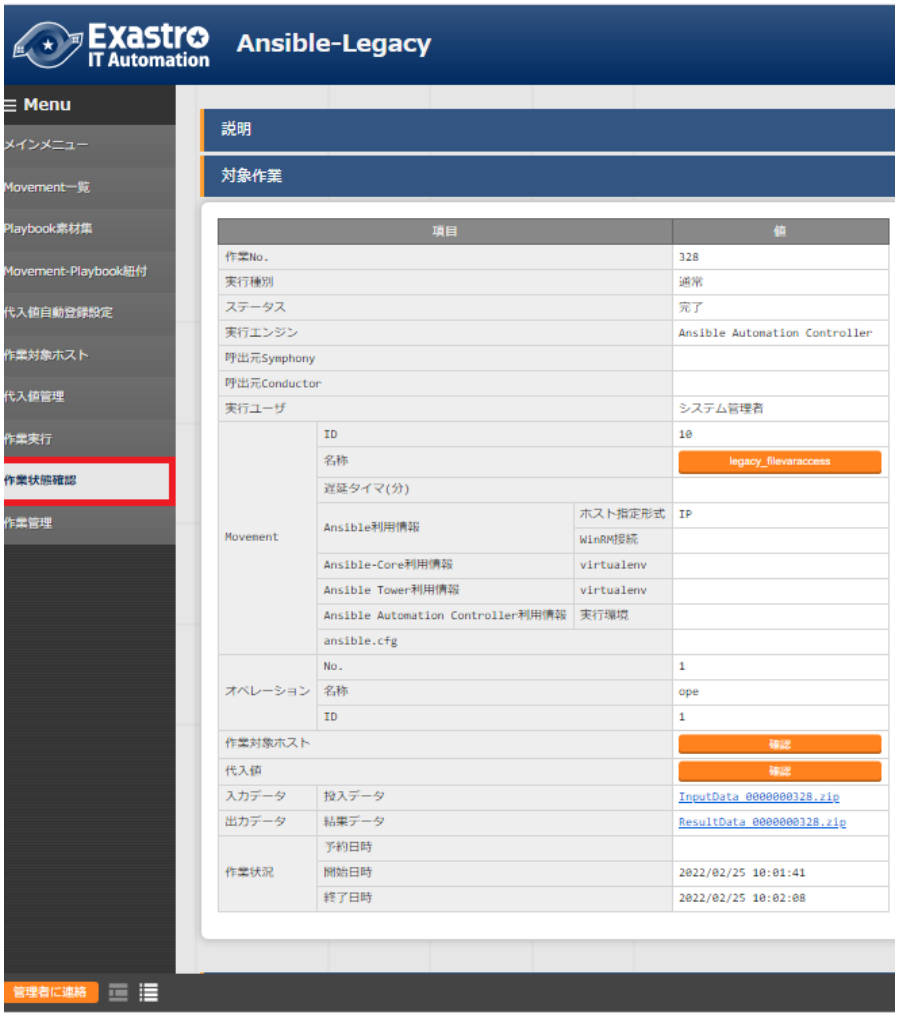


図 5.3-32 サブメニュー画面（作業状態確認）

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**

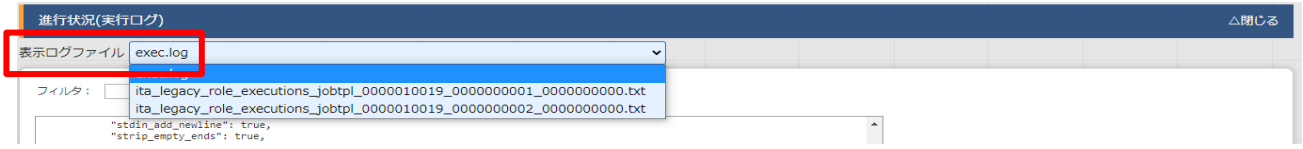
「確認」ボタンで「5.3.11 代入値管理」が表示され、作業対象のオペレーションと Movement に絞り込んだ代入値が表示されます。

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**

Ansible Automation Controller 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の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のを表します。

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、作業一覧テーブルを表示します。

「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。

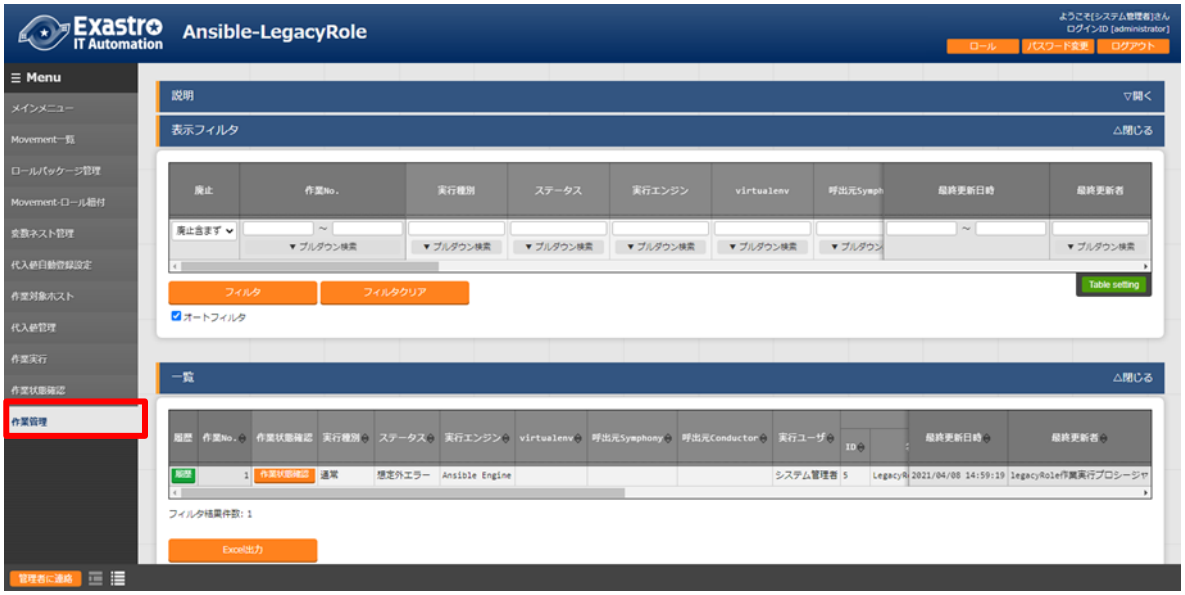


図 5.3-33 サブメニュー画面（作業管理）

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。

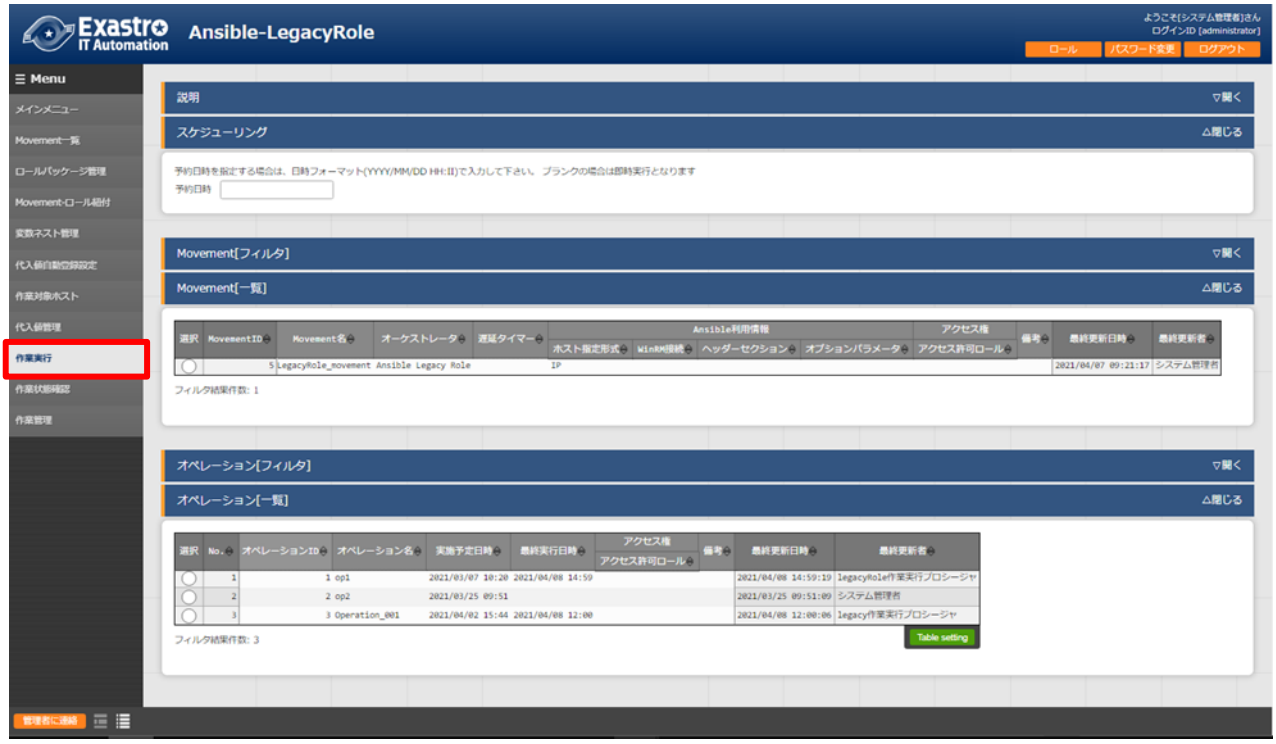


図 5.3-34 サブメニュー画面（作業実行）

※画面は Ansible-LegacyRole 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。

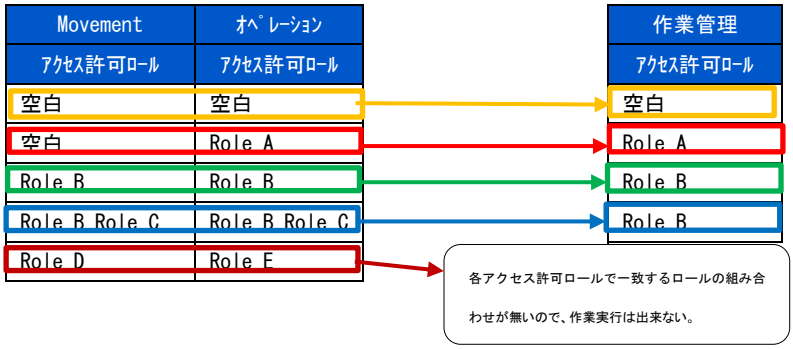
② 予約日時の指定

「予約日時」を入力することで、実行を予約することがきます。

「予約日時」には、未来の日時のみ登録可能です。

(1) 作業実行時にアクセス許可ロールの適合判定

Movement 一覧、オペレーション一覧で選択した Movement とオペレーションの各アクセス許可ロールで一致するロールがあるかを判定します。一致するロールが無い場合、その旨のエラーメッセージが表示され作業実行は出来ません。また、一致するロールが作業管理のアクセス許可ロールとして設定されます。また、アクセス許可ロールが空白の場合は、全てのロールへのアクセスが可能として扱われます。各アクセス許可ロールが空白の場合、作業管理のアクセス許可ロールも空白に設定されます。
アクセス許可ロールの詳細については「Exastro-ITA_利用手順マニュアル_データレコード毎のロールベースアクセス制御」を参照して下さい。



6　構築コード記述方法

6.1　Playbook（Ansible-Legacy）の記述

5.3.3.Playbook 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 で作成するマスター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 Core の場合

```
- hosts: all
  remote_user: "{{ __loginuser__ }}    "
  gather_facts: no
  become: yes
```

・Ansible Automation Controller の場合

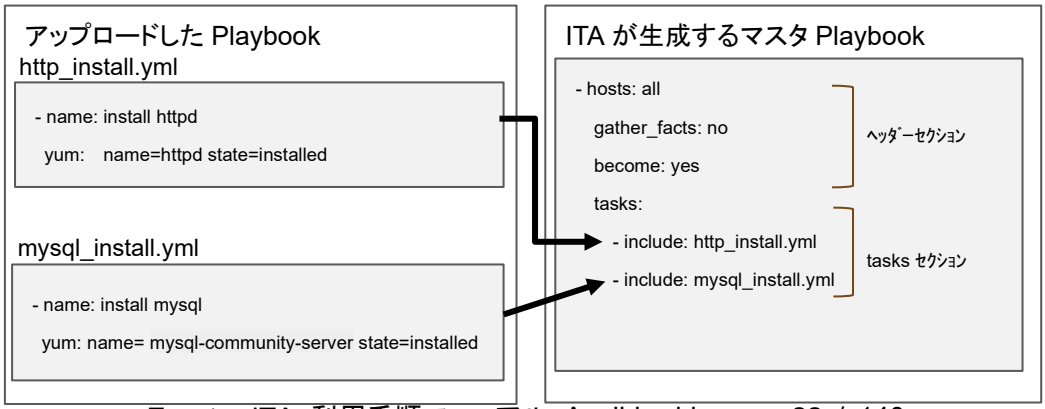
```
- hosts: all
  gather_facts: no
  become: yes
```

(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) }}"
```

アップロードさた Playbook は、「5.3.7.Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）」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。
対話ファイルは YAML 形式のファイルとして扱います。YAML 形式に準じていない記述があると、対話ファイルのアップロード時や作業実行時にエラーとなります。詳しくは、本章の「(7) 対話ファイルを yaml 形式で記載する際の注意事項」を参照して下さい。

(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。

```
Exp1-1.)
# 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 Automation Controller 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。

```
Exp2-1.)　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。

Exp2-2.)　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。
必要に応じコーテーションで囲むことを推奨します。

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。正規表記で記述できます。
△△△△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 ※△:半角スペース	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」。

Exp2-3.)
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'
- expect: root@{{ _loginhostname_ }}
  exec: exit
```

Exp2-4.)
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
- expect: root@{{ _loginhostname_ }}
    exec: exit
```

Exp2-5.)
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
- expect: root@{{ _loginhostname_ }}
    exec: exit
```

Exp2-6.)
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
  - expect: root@{{ _loginhostname_ }}
    exec: exit
```

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
```

Exp2-7.)
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
  - expect: root@{{ _loginhostname_ }}
    exec: exit
```

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。正規表記で記述できます。
△△△△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設定する変数「任意の文字列」を指定します。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が設定されます。設定した変数は command モジュールの条件判定(when・exec_when・failed_when)でのみ使用できます。 設定した変数は、1 つのみ保持できます。次に register で別の変数に値を設定した場合、前に設定した変数は削除されます。 1 exec_list: 2 - expect: 'assword:' 3 exec: '{{ __loginpassword__ }}' 4 - command: 'systemctl status httpd' 5 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 6 register: httpd_status_register 7 - command: 'systemctl restart httpd' 8 when: 9 - httpd_status_register no match(running) 10 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 11 - command: 'systemctl status mysql' 12 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 13 register: mysql_status_register 14 - command: 'systemctl restart mysql' 15 when: 16 - mysql_status_register no match(running) 17 prompt: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 18 - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}' 19 exec: exit 6 行目の httpd_status_register は、11 行目の command モジュール内の register で別の変数(mysql_status_register)に値を設定している為、10 行目までが有効範囲となります。
△△△△with_items: △△△△△△△-△'{{ VAR_x }}' △△△△△△△-△'{{ VAR_y }}' **定義する変数はシングルクォーテーションテーションで囲んで下さい。**	任意	with_items にコマンドをループして投入する場合に複数具体値変数の変数名を設定します。各変数のスコープは item.X(X は 0 から 99)とします。 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) with_items に設定する各変数の具体値数が同じでない場合、各変数の具体

パラメータ	必須/任意	説明
		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。 また、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 の具体値を設定する必要があります。(timeout も同様) prompt、timeout の変数の具体値数が不足していると、作業実行時にエラーになります。 command モジュールで、下記コマンドを実行したい場合 ・systemctl start httpd ・systemctl start mysql 対話ファイルは記述と with_items で使用する変数の具体値は以下の様になります。 - command: "systemctl {{ item.0 }} {{ item.1 }}" prompt: '{{ item.2 }}' timeout: '{{ item.3 }}' with_items: - '{{ VAR_status_list }}' # item.0 - '{{ VAR_service_list }}' # item.1 - '{{ VAR_prompt_list }}' # item.2 - '{{ VAR_timeout_list }}' # item.3 VAR_status_list: VAR_service_list: - start - httpd - start - mysql VAR_prompt_list: VAR_timeout_list: - コマンドプロンプト - 10 - コマンドプロンプト - 10 - コマンドプロンプト - 10 with_items で定義した変数は register/when 以外で item.X でスコープすることが出来ます。 Exp） with_items: ー ‘{{ VAR_item1 }}’ #item.0 ー ‘{{ VAR_item2 }}’ #item.1 exec_when: ー ‘{{ item.0 }} == active’ ー ‘{{ item.0 }} == {{ VAR_status }}’ ー ‘register 変数 match({{ item.0 }}) failed_when: ー ‘stdout match({{ item.1 }})’
△△△△when: △△△△△△-△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。

パラメータ	必須/任意	説明
△△△△△△-△xxx		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 Exp） － ‘VAR_status is define’ － ‘VAR_status is undefine’ ※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。 Exp） － ‘{{ VAR_status }} match(active)’ － ‘{{ VAR_status }} == active’ － ‘register 変数 match(active)’ ※active などの条件判定する文字列をクォーテーションで囲む必要はありません。 and/or による複合条件 or 条件を行いたい場合、判定条件の間に OR を記述します。 Exp） － ‘{{ VAR_status }} == 1 OR {{ VAR_status }} == 2’ 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。 Exp） － ‘{{ VAR_status }} == 1 OR {{ VAR_status }} == 2’ － ‘{{ VAR_sub_status }} == 1’
△△△△exec_when: △△△△△△-△xxx △△△△△△-△xxx	任意	ループ毎の条件判定です。(continue 条件) with_items が記述されている場合に条件判定を行います。 条件にマッチしていれば該当ループの command を実行します。 マッチしていなければ次のループへ移ります。 条件式 when:と同様の記述が行えます。
△△△△failed_when: △△△△△△-△xxx △△△△△△-△xxx	任意	command 実行後(ループ毎)の stdout の内容に対する条件判定です。 with_items が記述されている場合に条件判定を行います。 条件にマッチしていれば正常とします。 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。 条件式 変数具体値判定 stdout 比較演算子 文字列

パラメータ	必須/任意	説明
※△:半角スペース		stdout 比較演算子 VAR_xx stdout match(正規表記文字列/VAR_xx) stdout 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。 Exp） － ‘stdout == {{ VAR_status}}’ － ‘stdout match(active)’ － ‘stdout match({{ VAR_when }})’ ※active などの条件判定する文字列をクォーテーションで囲む必要はありません。 and/or による複合条件 when:と同様の記述を行います。

```
Exp3-1.)
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 の組合せでする。
# 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
  - expect: root@{{ __loginhostname__ }}
    exec: exit
```

```
Exp3-2)
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 の組合せでする。
# 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 }})

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

```
Exp3-3.)
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 の組合せでする。
# 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 }})

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

```
Exp3-4.)
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 の組合せでする。
# 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
   - expect: root@{{ __loginhostname__ }}
      exec: exit
```

```
Exp3.5.)
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 の組合せでする。
# 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

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

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

```
Exp3.6.)
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 の組合せでする。
# 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

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

and 条件

⑤ localaction モジュール

Ansible/Ansible Automation Controller 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。 conf セクションの timeout パラメータでのタイマ監視は適用外です。コマンドが完了するまで次のステップに進みません。
△△△△ignore_errors: △xxx ※△:半角スペース	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」。

```
Exp4-1.)
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
  - expect: root@{{ _loginhostname_ }}
    exec: exit
```

(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「(){}.など」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。

② 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

‘.*[¥#¥$¥%] $’

③ 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。

(5) コマンドプロンプト以外のプロンプトを処理する場合の注意事項

コマンドプロンプト以外のプロンプトを処理する場合、exec モジュールと expect モジュールの組合せで対話ファイルを作成して下さい。command と state モジュールでは処理できません。

Exp)

ssh-keygen を対話ファイルで処理する。

変数	具体値
VAR_id_rsa_path	秘密鍵ファイルのパスを設定します。
VAR_passphrase	パスフレーズを設定します。

リターンしたい場合
変数使用の場合は具体値を空白(未入力)にします。
変数を使用していない場合は空文字列(コーテーションを 2 個並べます)を記載します。
exec: ''

```
conf:
  timeout: 10

exec_list:
  # ssh 接続　パスワード認証
  - expect: 'assword:'
    exec: '{{ __loginpassword__ }}'

  # ssh-keygen コマンド実行
  - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'
    exec: ssh-keygen

  # 以降がコマンドプロンプト以外のプロンプトに対する処理
  # 秘密鍵ファイルのパスを設定
  # expect は正規表記で評価されるので、エスケープが必要なメタ文字にはエスケープ文字(¥)を挿入する必要があります。
  - expect: 'id_rsa¥):'
    exec: '{{ VAR_id_rsa_path }}'

  # パスフレーズを設定
  - expect: ' passphrase¥):'
    exec: '{{ VAR_passphrase }}'

  # パスフレーズを確認
  - expect: ' passphrase again:'
    exec: '{{ VAR_passphrase }}'

  # 生成された 秘密鍵ファイルを確認
  - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'
    exec: 'ls -al {{ VAR_id_rsa_path }}'

  # ssh 接続クローズ
  - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'
    exec: exit
```

(6) 対話ファイル終了時の注意事項

対話ファイルの最後に、セッションを終了するコマンドを投入するようにして下さい。
最終行のモジュールが終了するとセッションをクローズします。最終行にファイルコピーなど処理に時間がかかるコマンドが記載されている場合、コマンド終了前にセッションがクローズされコマンドが異常終了してしまう場合があります。

```
Exp)
conf:
  timeout: 10

exec_list:
  # ssh 接続　パスワード認証
  - expect: 'assword:'
    exec: '{{ __loginpassword__ }}'

  # ファイルコピー
  - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'
    exec:  ‘cp -rfp {{ VAR_src_path }} {{ VAR_dest_path }}

  # 直前のコマンド終了をコマンドプロンプトで待ち合わせ、exit コマンドを投入する記載を対話ファイルの最後に挿入
  - expect: '{{ __loginuser__ }}@{{ __loginhostname__ }}'
    exec: exit
```

(7) 対話ファイルを yaml 形式で記載する際の注意事項

対話ファイルは yaml 形式のファイルとして扱います。以下のような YAML 形式に準じていない記載があると対話モジュールのアップロード時や作業実行時にエラーとなります。

・各モジュールのパラメータに変数を記載している場合でパラメータ全体をクォーテーションで囲んでいない場合。

・各パラメータを定数のみで記載している場合で、定数の終端が「:」の場合など、パラメータ全体をクォーテーションで囲んでいない場合。

各モジュールのパラメータは、パラメータ全体をクォーテーションで囲むことを推奨します。

YAML 形式に準じていない記載(赤字)

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

YAML 形式に準じていない記載箇所のパラメータ全体をクォーテーションで囲み修正

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

with_items に列挙する変数は**シングルコーテーション**で囲って下さい。

(8) 構築対象機器のログインユーザーの LANG についての注意事項

ログインユーザーの「LANG」は、「UTF-8/euc/shift_jis」の 3 種類についてサポートしています。
ログインユーザーの「LANG」の設定は機器一覧より行ってください。
「euc/shift_jis」を設定した場合、構築対象機器との通信制御で使用している pexpect モジュールの UTF-8 へのデコード処理の特性で対話ファイルを正しく処理出来ない場合があります。
・一部の全角文字(①② 等)を UTF-8 にデコード出来ません。デコード出来ない文字は??で表示されます。
・一部の全角文字(ー 等)を expect 等のプロンプト待ちで使用した場合、「LANG」が UTF-8 では待ち受けが正しく出来ますが、LANG が「euc/shift_jis」では待ち受けがタイムアウトしてしまいます。

(9) 構築対象機器へ投入するコマンドの終端コードについての注意事項

構築対象機器へ投入するコマンドの終端コードは「LF」を送信します。構築対象機器のコマンド終端コードが「CRLF」の場合、対話ファイルで構築対象機器に投入するコマンドの末尾に「¥r」を追加して下さい。

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

6.3 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が上書きします。
|
├──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.md          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 Core の場合

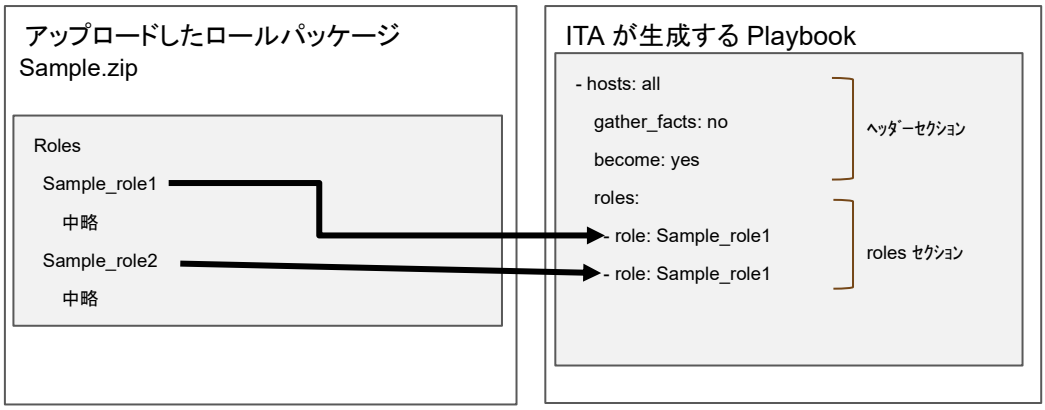
```
- hosts: all
  remote_user: "{{ __loginuser__ }}  "
  gather_facts: no
  become: yes
```

・Ansible Automation Controller の場合

```
- hosts: all
  gather_facts: no
  become: yes
```

② roles セクション

アップロードさたロールパッケージ内のロールを、「5.3.7.Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）」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意管理します。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 における所定ディクレトリのサブディレクトリについて注意事項

Ansible Role Directory Structure における所定ディクレトリの下位に当該の所定ディクレ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。*

(5) ロールパッケージ内のロール名をディレクトリ階層にした場合の留意点

下記のようなディレクトリ階層のロールパッケージを例に説明します。

```
└──roles
    ├──parent
    │    ├──sample_role1
    │    │    ├──defaults
    │    │    └──tasks
    │    └──sample_role2
    │         ├──defaults
    │         ├──sample_role3
    │         │    ├──defaults
    │         │    └──tasks
    │         ├──sample_role4
    │         │    ├──defaults
    │         │    └──tasks
    │         └──tasks
    ├──sample_role5
    │    └──defaults
    └──sample_role6
         ├──defaults
         └──tasks
```

① ロールとして認識するディレクトリは、tasks ディレクトリがあるディレクトリになります。

この例だと。ロールして扱うディレクトリ階層(ロール名)は以下の 3 個になります。

- parent/sample_role1
- parent/sample_role2
- sample_role6

② tasks ディレクトリが複数あるディレクトリ階層の除外

parent/sample_role2/sample_role3 と parent/sample_role2/sample_role4 にも tasks ディレクトリがありますが、parent/sample_role2 に tasks ディレクトリがありロールとして認識していますので、ロールとして扱いません。

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。

(1) 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

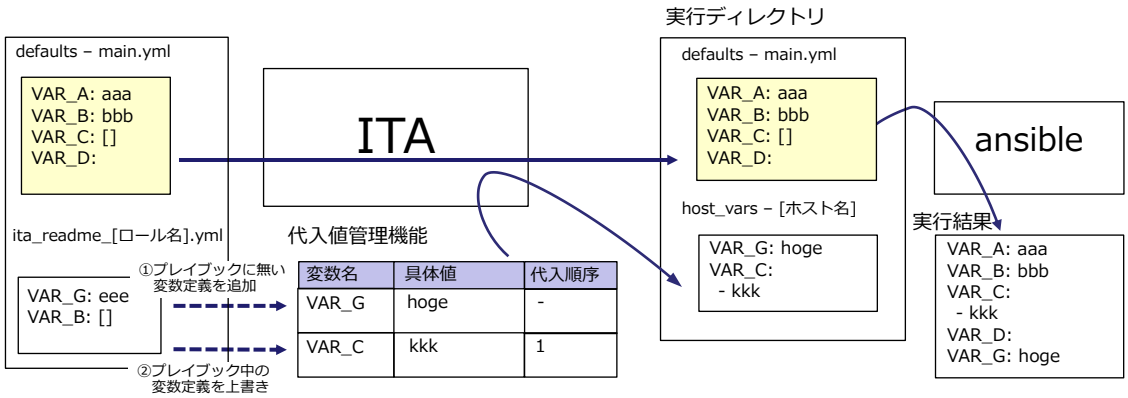
※role のディレクトリ階層が深い場合、ロール名に含まれる / を % に置き換える必要があります。

(2) 読替表のフォーマット

フォーマットは YAML 形式となります。

文字コードは 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だけに使用され、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。

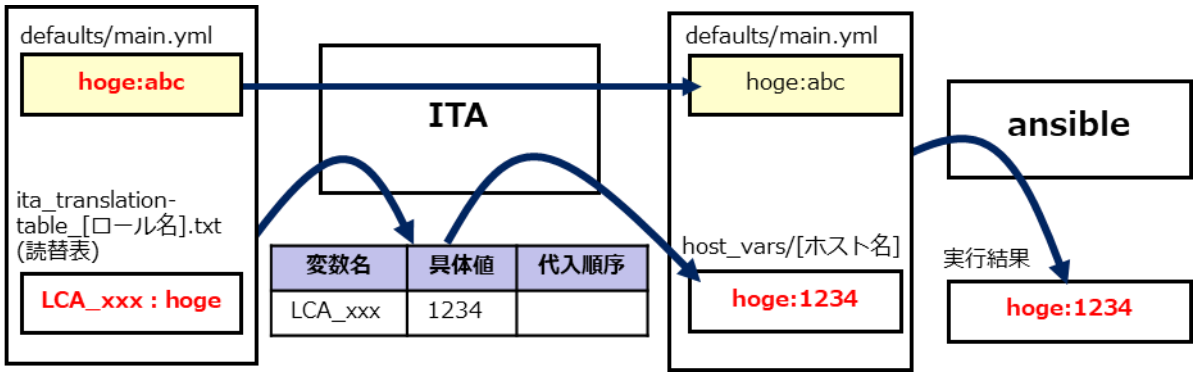
(1) 読替表のファイル名は以下の命名規則
ita_translation-table_[ロール名].txt
e.g.)
ロール名: mysql　　ファイル名: ita_translation-table_mysql.txt
ロール名: mysql/install　ファイル名: ita_translation-table_mysql%install.txt
※role のディレクトリ階層が深い場合、ロール名に含まれる / を % に置き換える必要があります。

(2) 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。
文字コードは UTF-8 の BOM なしで作成して下さい。
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_***
***：半角英数字とｱﾝﾀﾞｽｺｱ（ _ ）が利用可能です。(最小値:1 ﾊﾞｲﾄ、最大値:256 ﾊﾞｲﾄ)
任意変数:：(最小値:1 ﾊﾞｲﾄ、最大値:256 ﾊﾞｲﾄ)
($s*)：半角スペース 0 個以上
($s+)：半角スペース 1 個以上

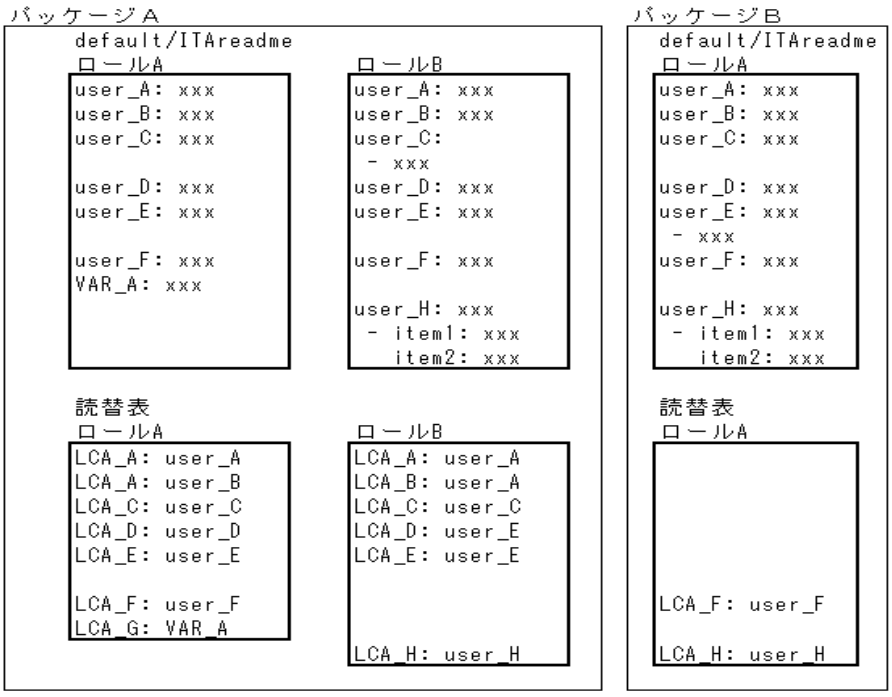
e.g.)
LCA_var1: var1
#から始まる行はコメント行
LCA_var2: var2



(3) 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 がない。(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



6.6 「ita_readme」と「読替表」の活用例（Ansible-Legacy Role のみ）

Ansible-Legacy Role における「ita_readme」と「読替表」の活用例について、観点 1～9 を列挙します。

前提として、Ansible-Legacy Role（「roles」ディレクトリ）は外部から取得したものとします。
以下は、「ita_readme」と「読替表」を用いてアップロードから結果確認までを表した全体イメージ図です。

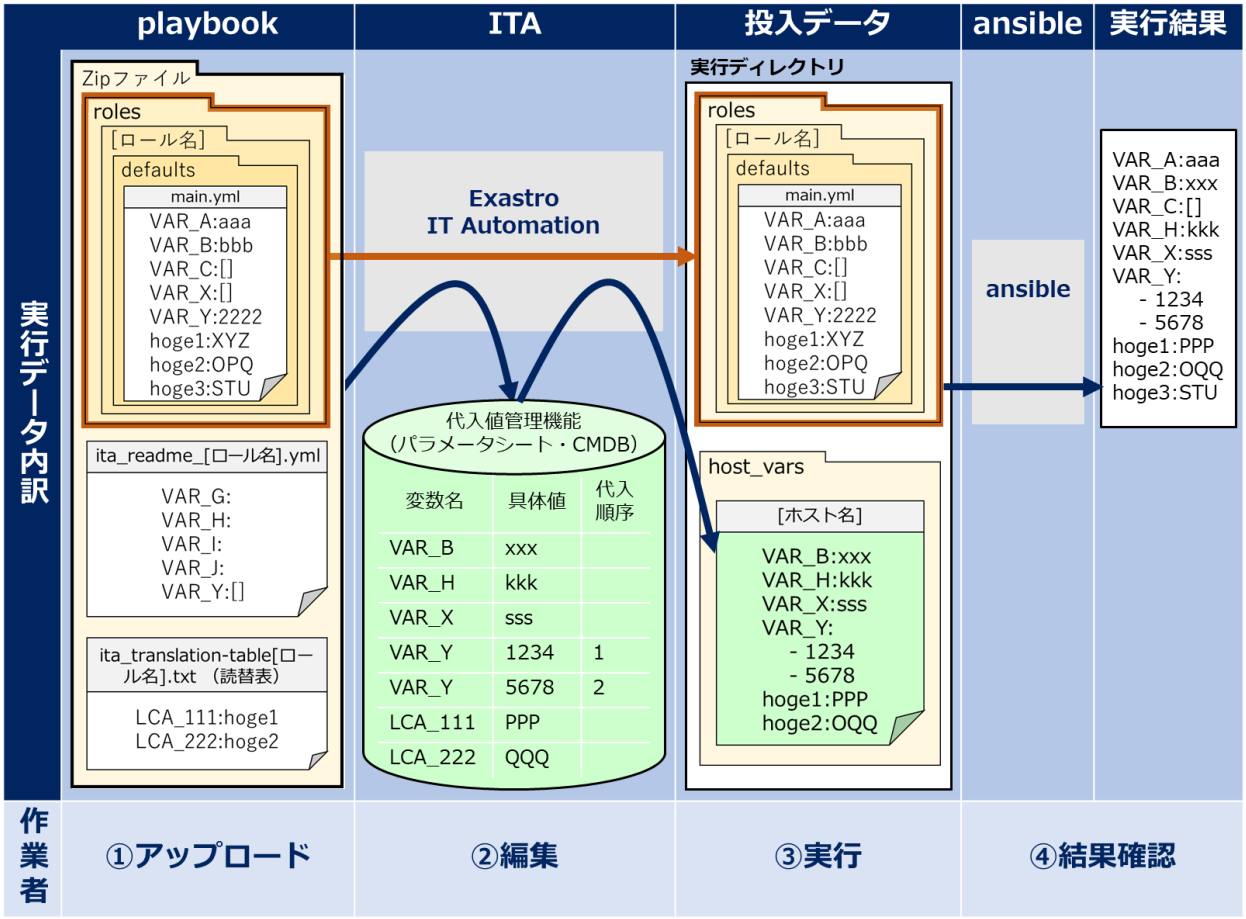


図 6.6-1 全体イメージ図

以降、上記の図をベースに観点 1～9 について掘り下げていきます。

No.	観点
1	外部から取得した Ansible-Legacy Role を編集せず利用する
2	「ita_readme」と「読替表」の役割
3	「defaults/main.yml」に記載の変数定義およびデフォルト値について
4	「host_vars ファイル」と「ITA の CMDB」について
5	「defaults/main.yml」に追記したい場合の救済処置
6	変数名の先頭“VAR_”の有無について
7	「ita_readme」と「読替表」の連携活用
8	playbook における length 評価への応用
9	playbook における defined 評価への応用

- **観点1:外部から取得した Ansible-Legacy Role を編集せず利用する**

外部(Galaxy 等)から取得した Ansible-Legacy Role(「roles」ディレクトリ)は編集を加えずに利用いただくことが可能です。
そのため「ita_readme」や「読替表」を「roles」ディレクトリの外に置いて、Ansible-Legacy Role(「roles」ディレクトリ)内で使われている変数にパラメータを与えることが可能となっております。

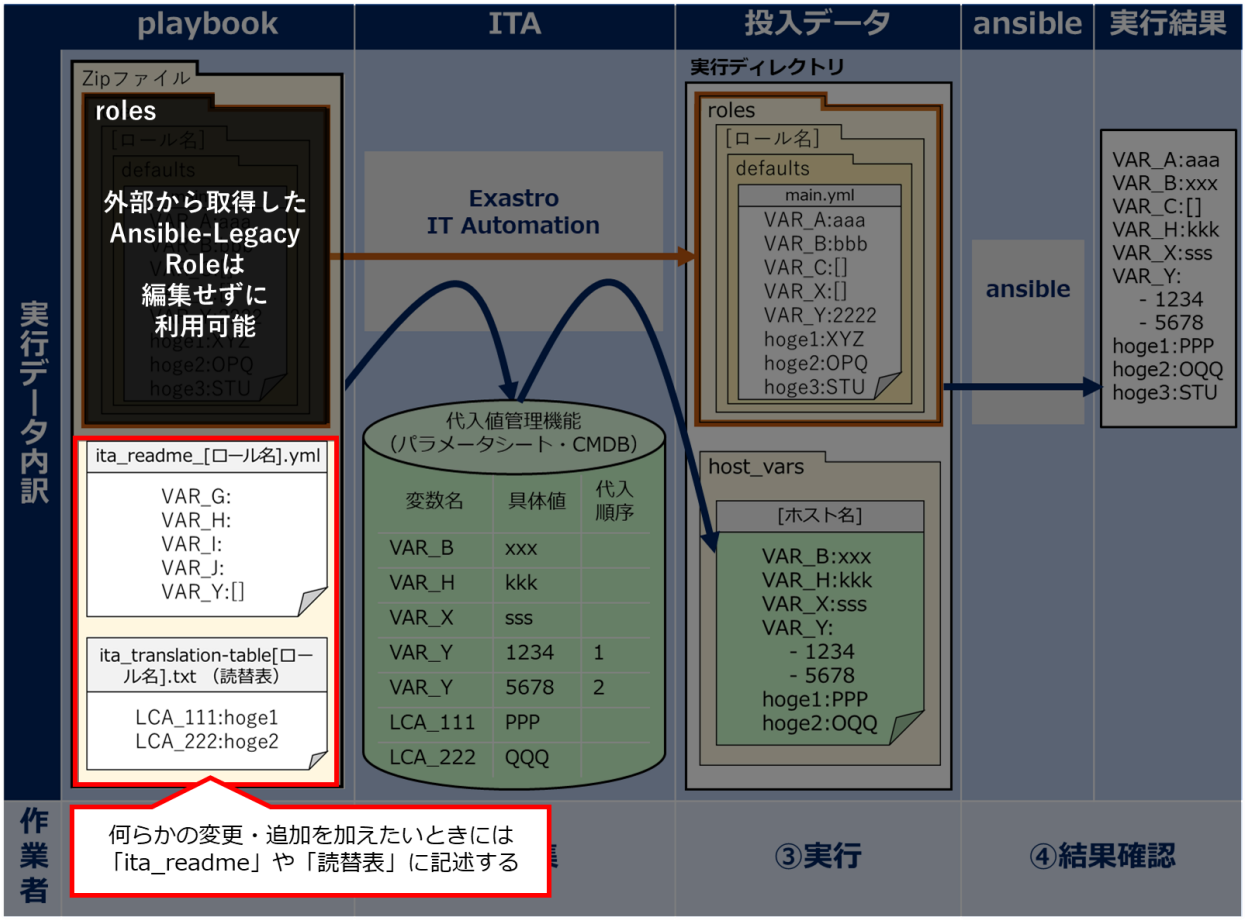


図 6.6-2 観点1のイメージ図

- **観点2:「ita_readme」と「読替表」の役割について**

「ita_readme」、「読替表」は変数名および変数の型を ITA に伝えるための機能です。
言い換えれば、「ita_readme」および「読替表」は変数の具体値（パラメータ）を定義するための機能ではありません（具体値を記載しても ITA で認識しません）。

具体値を与える方法を以降の観点で説明します。

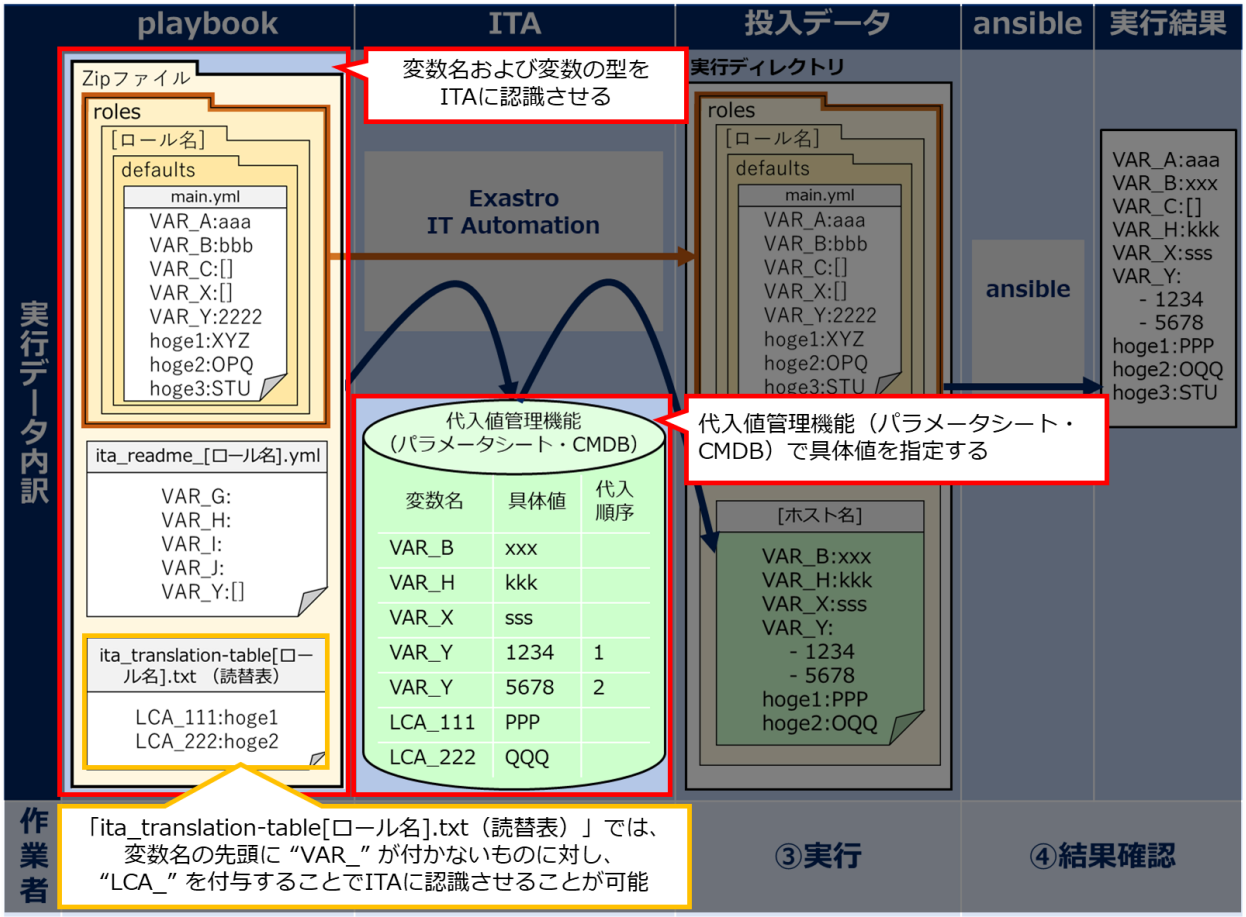


図 6.6-3 観点2のイメージ図

- **観点3:「defaults/main.yml」に記載の変数定義およびデフォルト値について**

「roles」配下の「defaults/main.yml」はそのまま変更なく ansible に渡されます。
変数定義およびデフォルト値は host_vars で定義されない限り有効となります。(例:『VAR_A:aaa』)

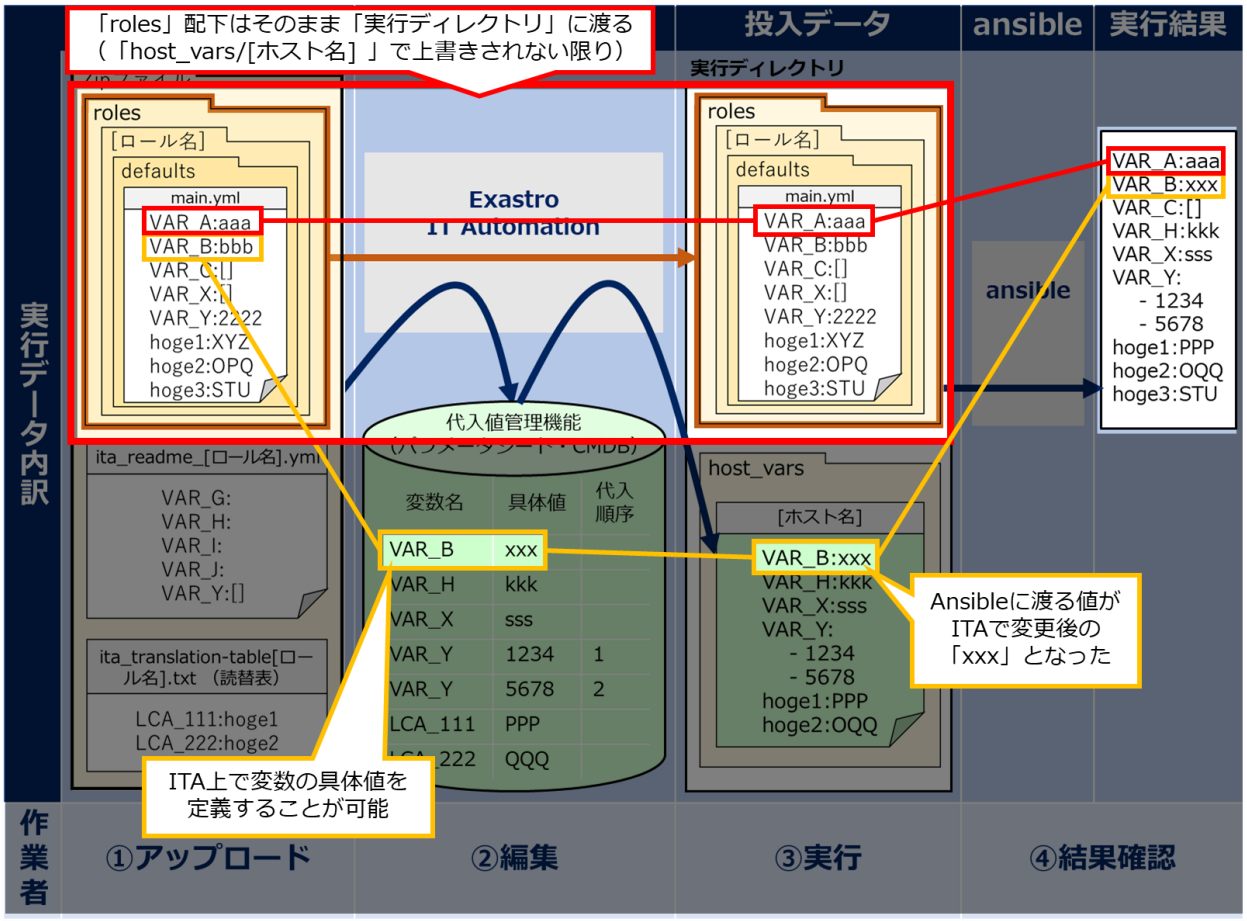


図 6.6-4 観点3のイメージ図

- **観点4:「host_vars ファイル」と「ITA の CMDB」について**

host_vars ファイルは ITA の CMDB（パラメータシート）から実行ごとに自動作成されます。

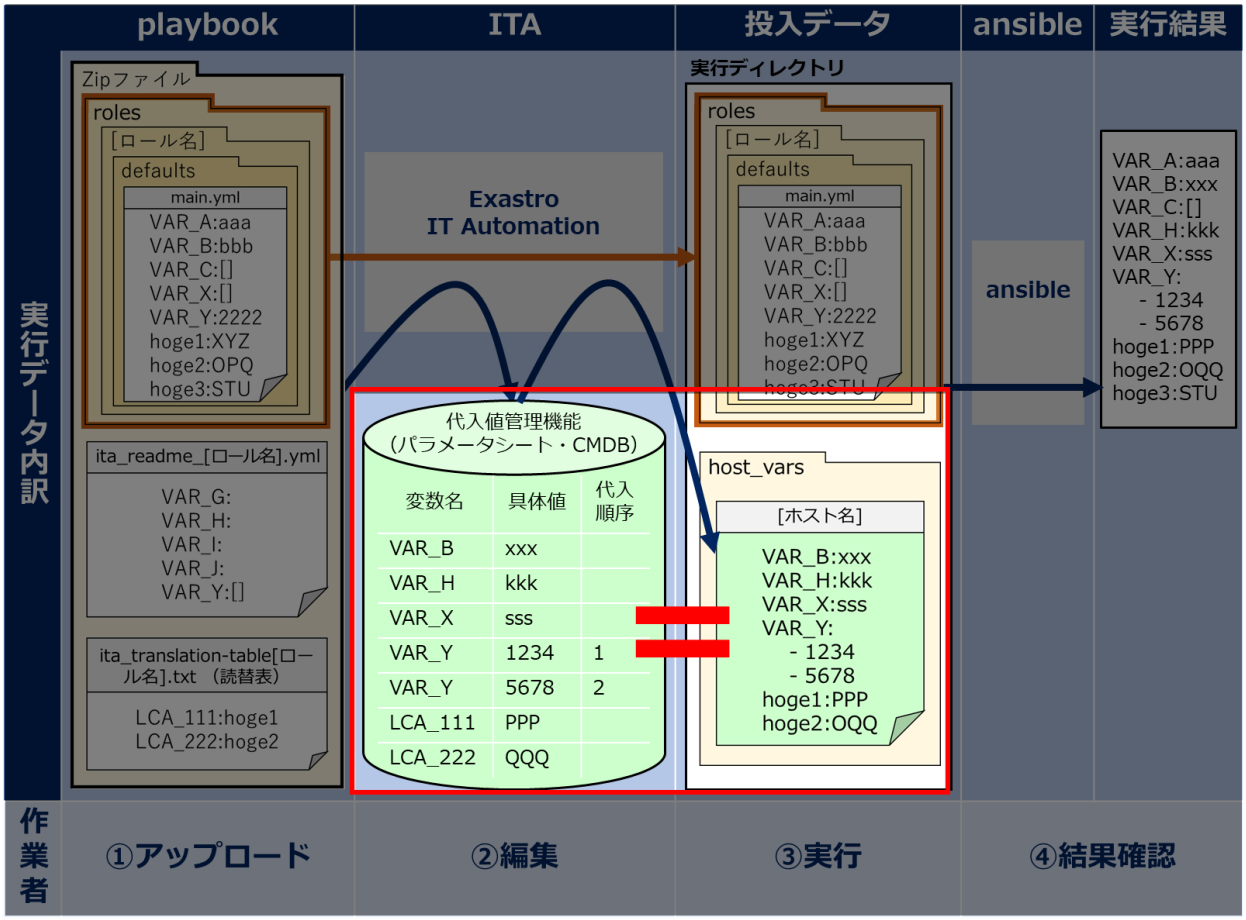


図 6.6-5 観点4のイメージ図

- **観点５:「defaults/main.yml」に追記したい場合の救済処置**

Ansible-Legacy Role（「roles」ディレクトリ）に変更を加えたい場合、救済処置として「ita_readme」に変数名および型を記述することが可能です。

既に「defaults/main.yml」に記載がある変数を、改めて「ita_readme」に定義する必要はありません。
もし二つのファイルで同じ変数が定義されている場合は、「ita_readme」側が優位になります。

※下図のとおり、変数「VAR_H」を「ita_readme」に記述することで変数の追加が可能

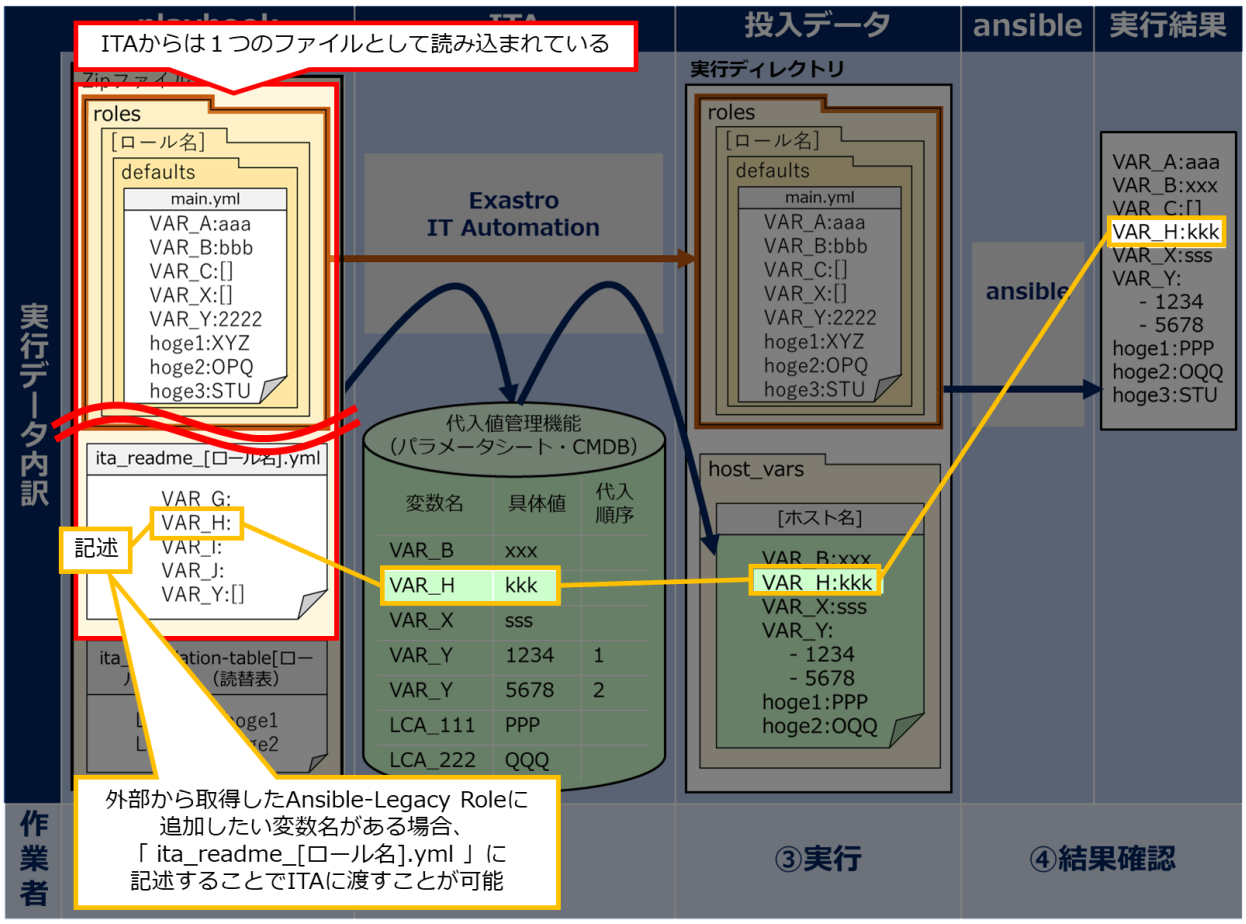


図 6.6-6 観点5のイメージ図

- **観点6：変数名の先頭“VAR_”の有無について**

「defaults/main.yml」の変数のうち、 ITA は“VAR_”という接頭文字から始まるものを変数マスタに管理します。

「defaults/main.yml」の変数のうち、“VAR_”という接頭文字から始まっていないものを ITA にて変数管理させる場合は「読替表」を活用します。

「読替表」では“VAR_”という接頭文字から始まっていない変数を“LCA_” という接頭文字から始まる変数名と紐づけることで ITA にて変数管理させることが可能となります。

なお、本機能の応用例として、「defaults/main.yml」の変数（“VAR_”という接頭文字から始まっていない）に ITA からパラメータを与えずに実行したい場合は、あえて「読替表」での変数紐づけを避けることも可能です。
※以下、図の変数名「hoge3」のパターン

※「読替表」には“LCA_”から始まる変数定義のみ有効

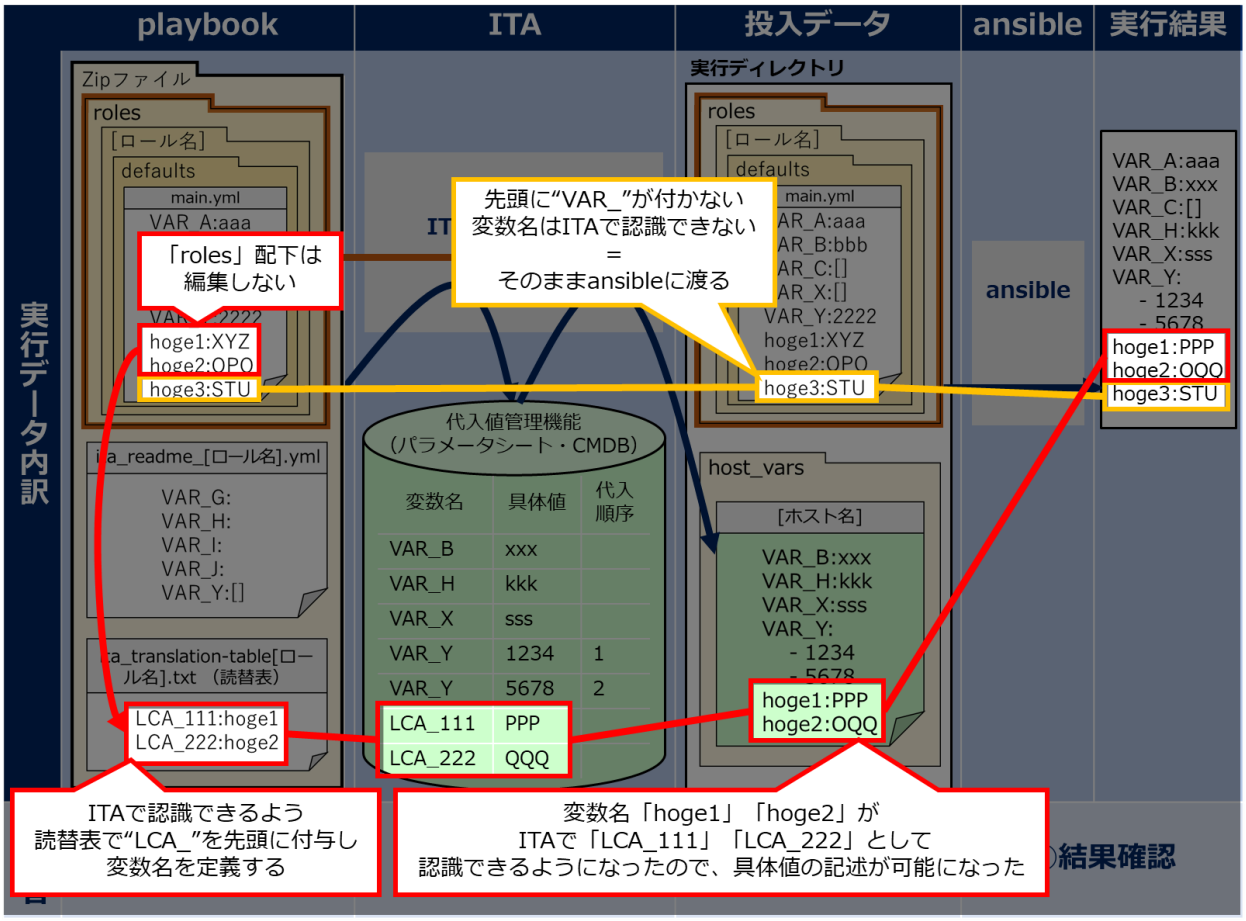


図 6.6-7 観点6のイメージ図

- **観点7:「ita_readme」と「読替表」の連携活用**

「tasks/main.yml」（playbook 本体）において“VAR_”から始まっていない変数が使われており、また、その変数が「default/main.yml」に定義されていない場合、「ita_readme」と「読替表」を連携活用することで、ITA からパラメータを与えることが可能です。

例えば、下図の様に「roles」配下の「tasks/main.yml」に変数「hoge」が使われている場合、以下の手順によって ITA に変数を伝えることが可能です。

① 「ita_readme」に、変数名「hoge」を追記する
② 「読替表」に、変数名「hoge」を ITA の CMDB（パラメータシート）で「LCA_xxx」という変数名で扱うことを追記する

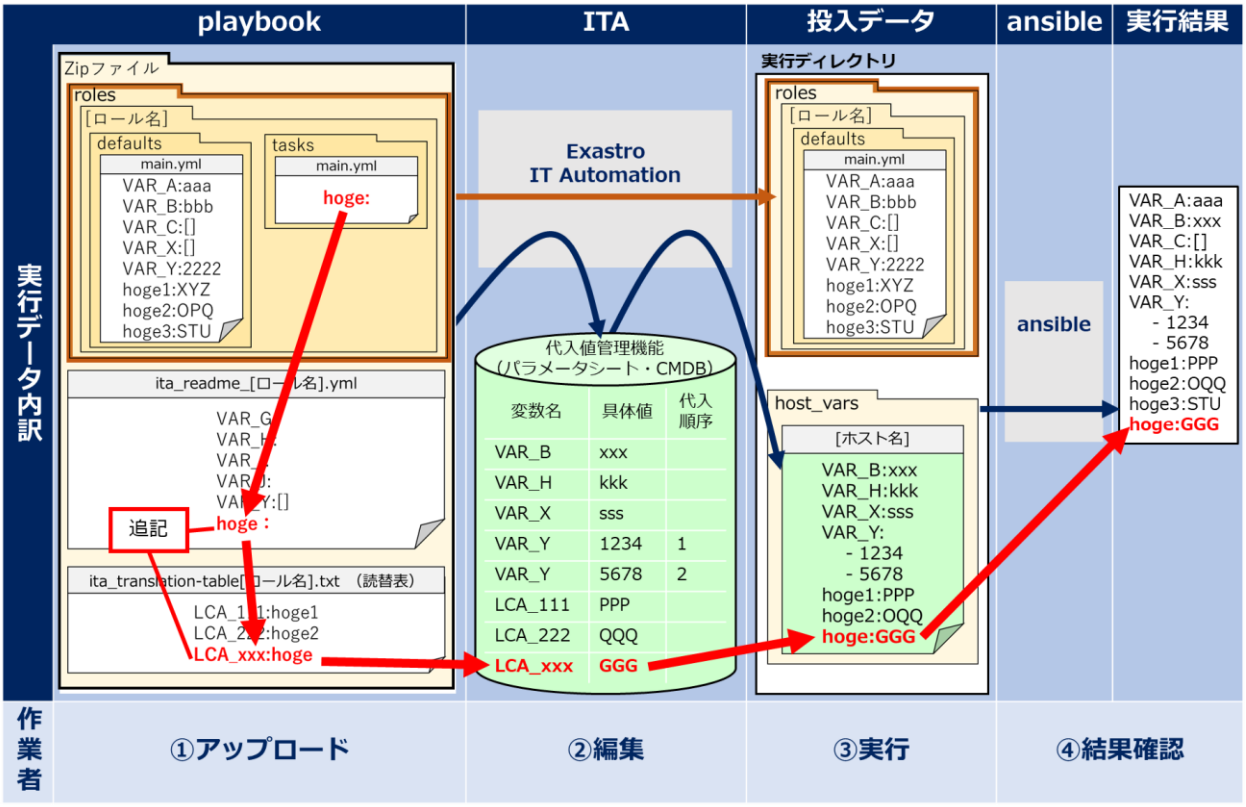


図 6.6-8 観点7のイメージ図

- **観点8：playbook における length 評価への応用**

変数に対し具体値があるか否かによって、length 評価における条件分岐に活用することが可能です。

例えば、「defaults/main.yml」に『VAR_C：[]』がある状態で、 変数「VAR_C」に具体値を与えずに実行した場合 length＝0 となります。
反対に、何らかの具体値を与えて実行した場合 length＞0 となります。（例：『VAR_X：sss』）

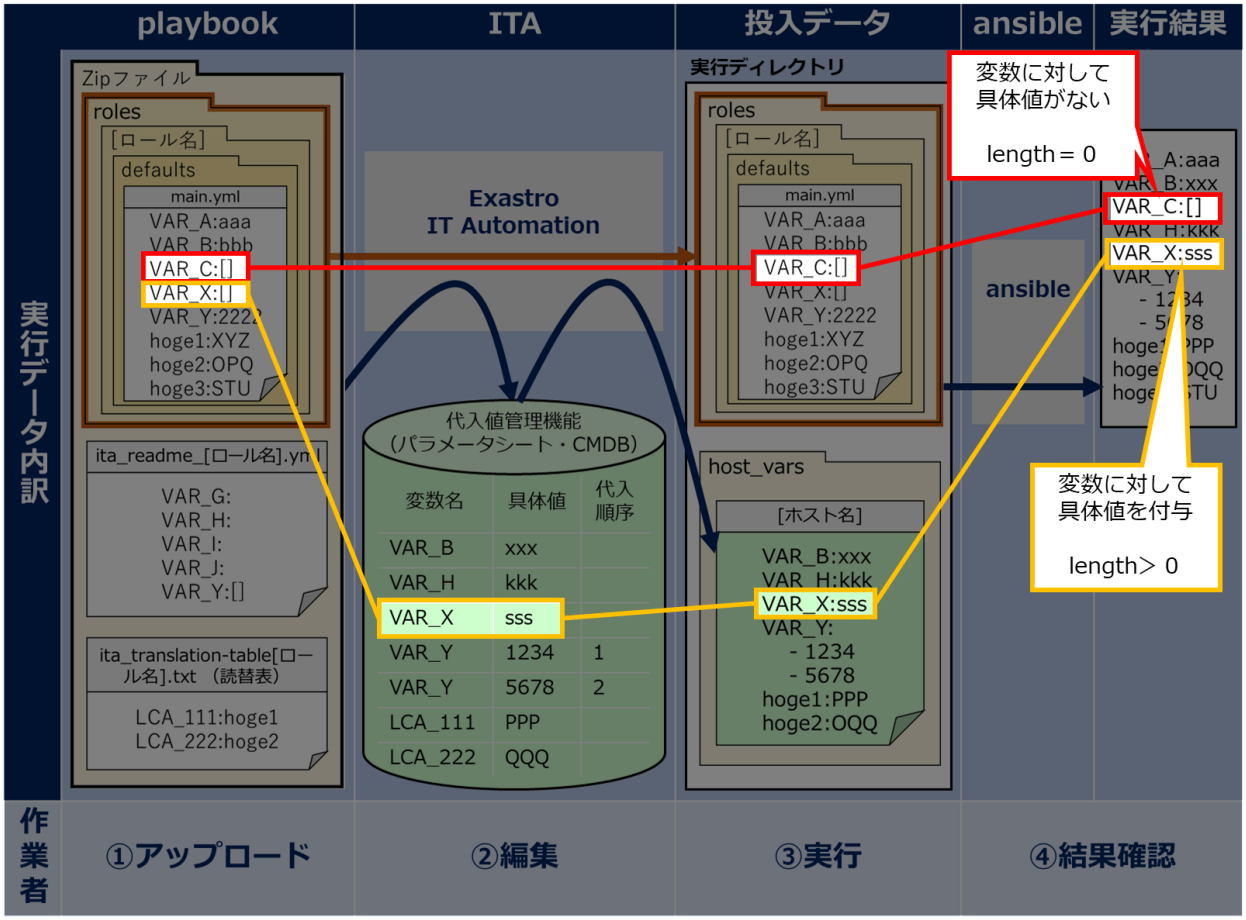


図 6.6-9 観点8のイメージ図

- **観点9：playbook における defined 評価への応用**

変数に対し具体値を定義しているか否かによって、defined 評価による条件分岐に活用することが可能です。

例えば、「defaults/main.yml」で定義のない変数「VAR_G」と「VAR_H」を、「ita_readme」で定義を記述します。「ita_readme」に記述することで、 ITA の CMDB（パラメータシート）で取り扱うことが可能となります。

変数「VAR_G」に具体値を付与せず実行すると、「defaults/main.yml」および「host_vars」に定義されずに動作するため defined→false となります。
反対に、変数「VAR_H」に具体値「kkk」を付与し実行すると、「host_vars」に定義されて動作するため defined→true となります。

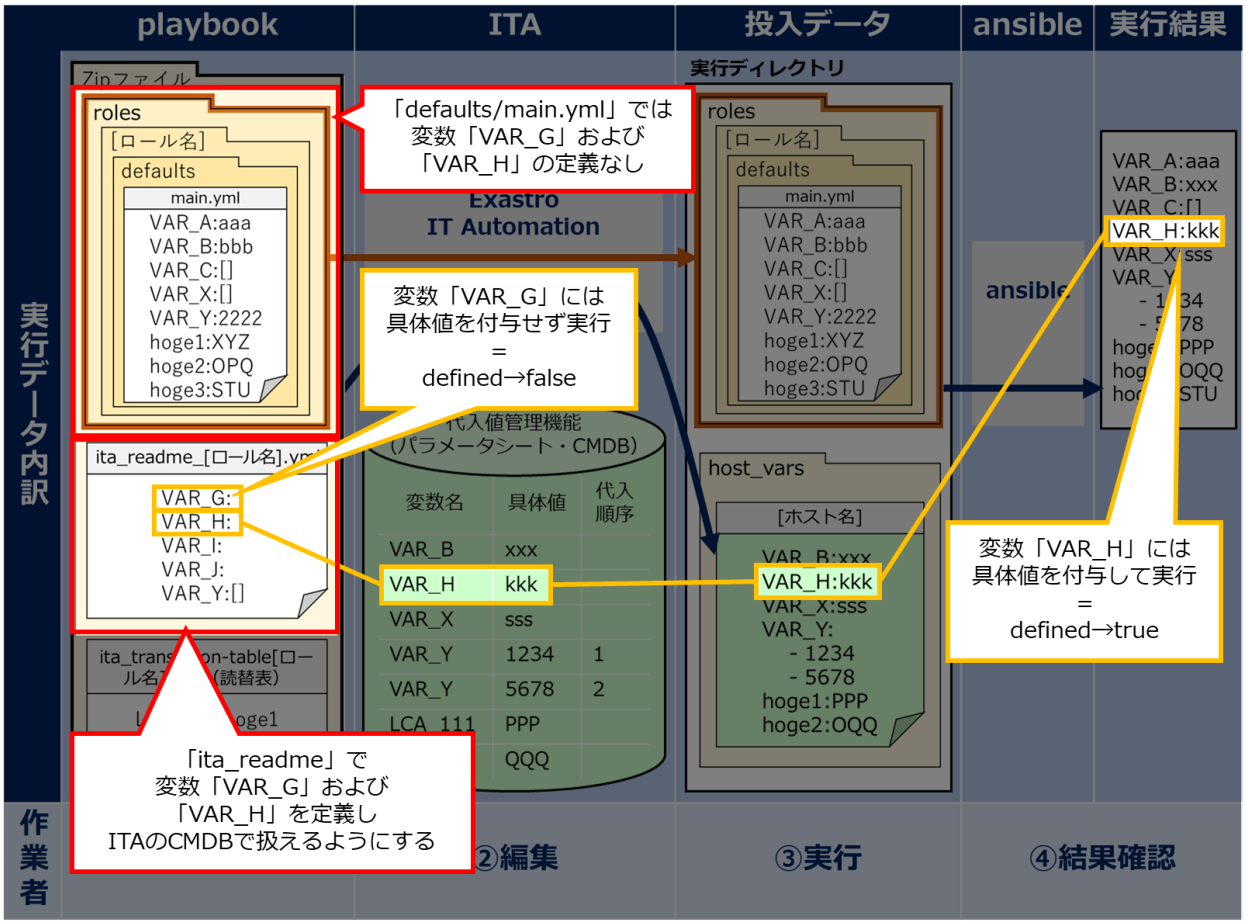


図 6.6-10 観点9のイメージ図

6.7 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
Playbook 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

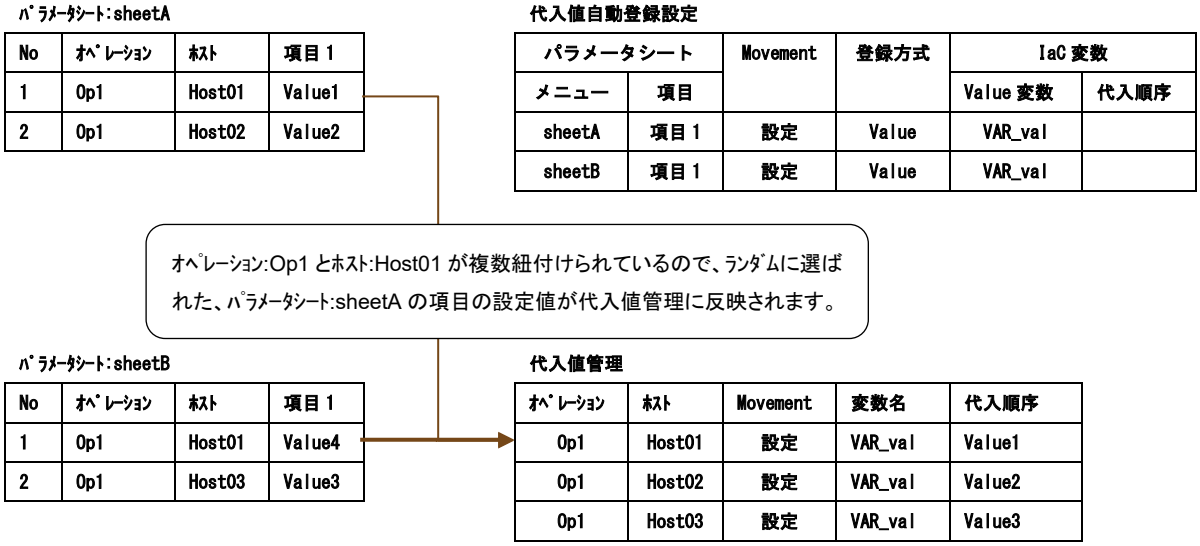
作業対象ホストの状態	該当レコードなし	該当レコードあり	該当レコード廃止中
作業対象ホストへの反映	新規レコード追加	ー	廃止レコード復活

※ 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。



6.8 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
Ansible Automation Controller データ同期 Ansible Automation Controller から各種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/confs/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/confs/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

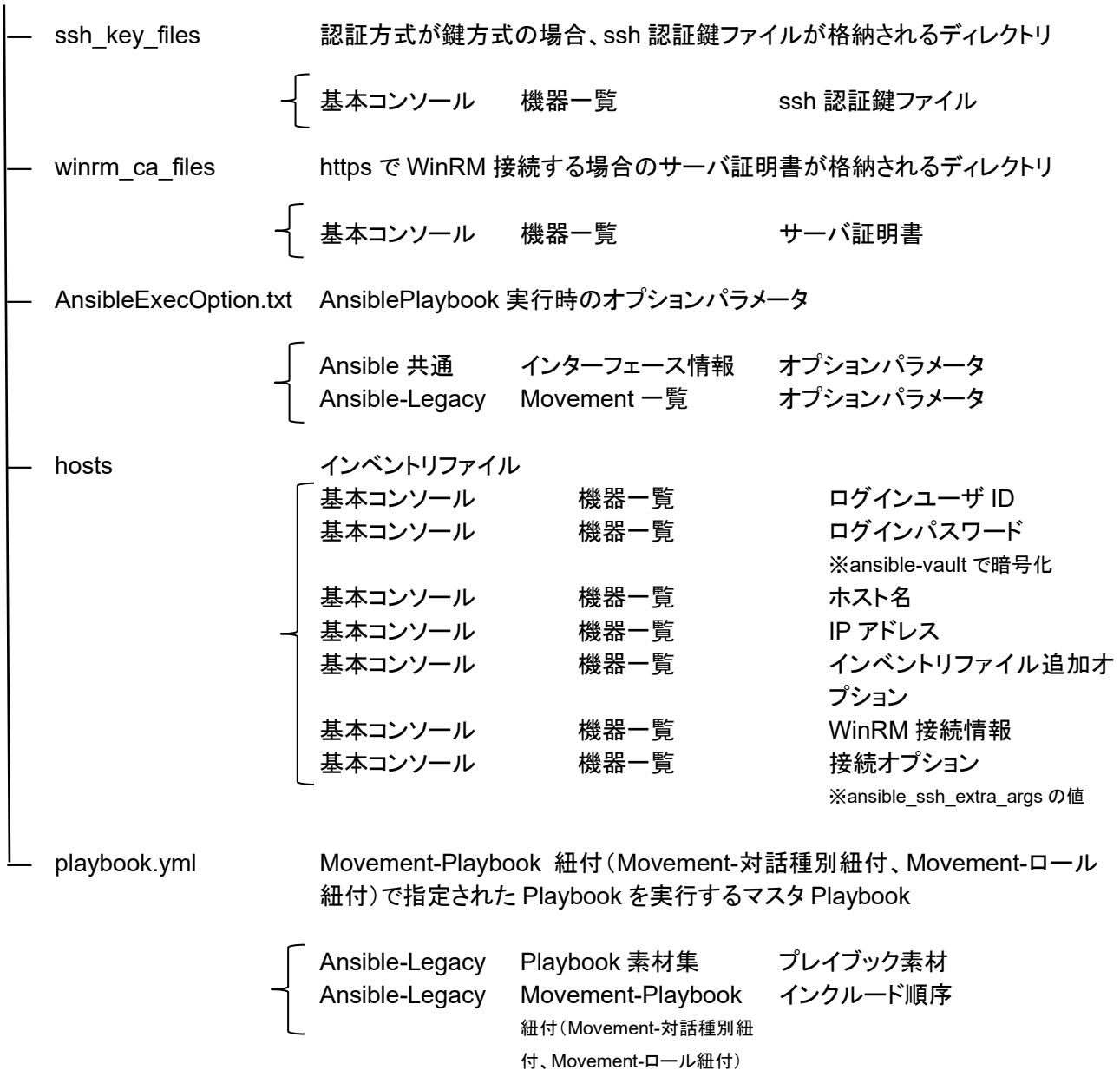
各 ITA メニューより情報を抜出して Ansible 実行に必要な投入データを作ります。この際、機器一覧のパスワードや代入値管理で Sensitive 設定を ON した変数の具体値が ansible-vault で暗号化されています。
投入データは ZIP 形式で「5.3.12 作業状態確認」よりダウンロードが可能です。
ダウンロードした投入データを所定のディレクトリに解凍することで、Ansible を直接実行することも可能です。

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 Playbook が格納されるディレクトリ		
	Ansible-Legacy	Playbook 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 共通	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 共通	ファイル管理	ファイル素材
upload_files	実行する Playbook 内で使用している変数の具体値に対して代入値管理でファイルをアップロードしている場合、該当ファイルを格納するディレクトリ		
	Ansible-Legacy	代入値管理	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(文字列)
	Ansible-Legacy	代入値管理	変数名/具体値(ファイル)
	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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）で指定された Playbook を実行するマスタ Playbook
Ansible-Legacy Playbook 素材集 プレイブック素材
Ansible-Legacy Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付） 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　Playbook 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 共通　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 共通　ファイル管理　ファイル素材
- upload_files　実行する Playbook 内で使用している変数の具体値に対して代入値管理でファイルをアップロードしている場合、該当ファイルを格納するディレクトリ
  - Ansible- Pioneer　代入値管理　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(文字列)
Ansible-Pioneer	代入値管理	変数名/具体値(ファイル)
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
  - 基本コンソール　機器一覧　ssh 認証鍵ファイル
- 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-Pioneer　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
※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　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　Pioneer 利用情報
  - 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）で指定された対話ファイルを実行するマスタ Playbook
  - Ansible-Pioneer　話ファイル素材集　話ファイル素材
  - Ansible-Pioneer　Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）　インクルード順序

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　Playbook 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 共通　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 共通　ファイル管理　ファイル素材
- upload_files　実行する Playbook 内で使用している変数の具体値に対して代入値管理でファイルをアップロードしている場合、該当ファイルを格納するディレクトリ
  - Ansible-LegacyRole　代入値管理　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(文字列)
  - Ansible-LegacyRole　代入値管理　変数名/具体値(ファイル)
  - 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-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）で指定された Role を実行するマスタ Playbook

Ansible LegacyRole	Playbook 素材集 Movement-Playbook 紐付（Movement-対話種別紐付、Movement-ロール紐付）	プレイブック素材 インクルード順序

8.1.4 投入データを直接実行する方法

(1)投入データの解凍先ディレクトリの作成

以下の 2 つのディレクトリを作成し、1.のディレクトリに投入データを解凍します。

1. /ベースディレクトリ/ドライバパス/作業番号/in
2. /ベースディレクトリ/ドライバパス/作業番号/out

ベースディレクトリ: インターフェース情報=>データリレイストレージパス(Ansible)

ドライバパス: legacy: legacy/ns Legacy-role: legacy/rl pioneer: pioneer/ns

作業番号: 作業実行時の作業番号を右寄せで不足分は 0 を埋めた 10 桁の文字列。

作業番号: 12345 => 0000012345

尚、投入データには、機器一覧にアップロードした秘密鍵ファイルが含まれていません。秘密鍵ファイルを必要といる認証方式の場合、投入データに含まれているインベントリファイル「hosts」を開き、ansible_ssh_private_key_file に設定されているパスに秘密鍵ファイルをコピーして下さい。

インベントリファイル「hosts」

```
all:
  children:
    hostgroups:
      hosts:
        target_host_1:
          ansible_user: keyauth_user
          ansible_ssh_private_key_file: /exastro/data_relay_storage/ansible_driver/legacy/ns/0000000060/in/ssh_key_files/0000000006-keyauth_user_id_rsa
```

(2) 投入データを直接実行するコマンド

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 --vault-password-file パスワードファイル 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/confs/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。

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 Core の場合	Ansible Automation Controller 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。	○	○

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 Core の場合	Ansible Automation Controller 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	○	
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-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。	○	○

(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 Core の場合	Ansible Automation Controller 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。	○	○

8.3 オプションパラメーター覧

インターフェース情報と Movement 一覧のオプションパラメータについて説明します。
ITA では、以下の順番で ansible-playbook のオプションパラメータを設定するので、単一の値しか許容していないパラメータを複数定義した場合、Movement 一覧=>オプションパラメータのパラメータが有効になります。

・Ansible 共通=>インターフェース情報=>オプションパラメータ
・Movement 一覧=>オプションパラメータ

実行エンジンが Ansible Core の場合

実行エンジンが Ansible Core 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。
ansible-playbook コマンドのオプションパラメータの仕様については、以下のコマンドを実行して表示されたヘルプを参照してください。
「 ansible-playbook –h 」

実行エンジンが Ansible Core 以外の場合

以下の表 8.3.1 は実行エンジンが Ansible Core 以外の場合に、指定可能なオプションパラメータです。

表 8.3.1 実行エンジンが Ansible Core 以外の場合に指定可能なオプションパラメーター覧

オプションパラメータ	指定方法	Ansible Automation Controller の設定箇所	備考
-v --verbose	-v -vv -vvv -vvvv -vvvvv --verbose	テンプレート画面の[詳細]に指定した v の数を設定	・v の合計値を適用 ・--verbose は、-v と同様 例：--verbose –vvv の場合、-vvvv と同様 ・v を 6 以上指定した場合、v は 5 の指定となる
-f --forks	-f FORKS --forks=FORKS	テンプレートの[フォーク]に指定した FORKS が設定される	・FORKS には数値を指定 ・複数定義した場合、最後に定義したパラメータが有効 例：-f 1 –forks=10 の場合、--forks=10 が有効となる ・数値以外が指定された場合、エラーとなる
-l --limit	-l SUBSET --limit=SUBSET	テンプレートの[制限]に指定した SUBMIT が設定される	・SUBSET：機器一覧にあるホスト名 ・複数定義した場合、最後に定義したパラメータが有効

オプションパラメータ	指定方法	Ansible Automation Controller の設定箇所	備考
-e --extra-vars	-e EXTRA_VARS --extra-vars=EXTRA_VARS	テンプレートの[追加変数]に変数名:具体値の形式で設定される	・EXTRA_VARS: 変数名、具体値を json 形式または yaml 形式 例: json 形式の場合 –extra-vars={"VAR_1":"directory","VAR_2":"0755"} yaml 形式の場合 –extra-vars=VAR_1: directory¥nVAR_2: 0755 ・複数定義した場合、最後に定義したパラメータが有効
-t --tags	-t TAGS --tags=TAGS	テンプレートの[ジョブタグ]に設定した TAGS が設定される	・TAGS:タグ名 ・複数個のパラメータを許容
-b --become	-b --become	テンプレートのオプション[権限昇格の有効化]がチェックされる	・少なくとも 1 つ指定すればパラメータが有効
-D --diff	-D --diff	テンプレートの[変更]の表示が有効化される	・少なくとも 1 つ指定すればパラメータが有効
--skip-tags	--skip-tags=SKIP_TAGS	テンプレートの[スキップタグ]に設定した SKIP_TAGS が設定される	・SKIP_TAGS:スキップタグ名 ・複数個のパラメータを許容
--start-at-task	--start-at-task=START_AT_TASK	※Ansible Automation Controller の Web UI には --start-at-task の表示はない。	・複数定義した場合、最後に定義したパラメータが有効
-ufc --use_fact_cache	-ufc --use_fact_cache	テンプレートのオプション[ファクトキャッシュの有効化]がチェックされる	・少なくとも 1 つ指定すればパラメータが有効
-as --allow_simultaneous	-as --allow_simultaneous	テンプレートのオプション[同時実行ジョブの有効化]がチェックされる	・少なくとも 1 つ指定すればパラメータが有効
-jsc --job_slice_count	-jsc ジョブスライス数 --job_slice_count=ジョブスライス数	テンプレートの[ジョブスライス数]に指定したジョブスライス数が設定される	・ジョブスライス数には数値を指定 ・複数定義した場合、最後に定義したパラメータが有効

※Ansible Automation Controller のオプションパラメータの仕様については、Ansible Automation Controller 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。

8.4 Ansible Automation Controller で ITA 独自変数を利用する場合の留意事項

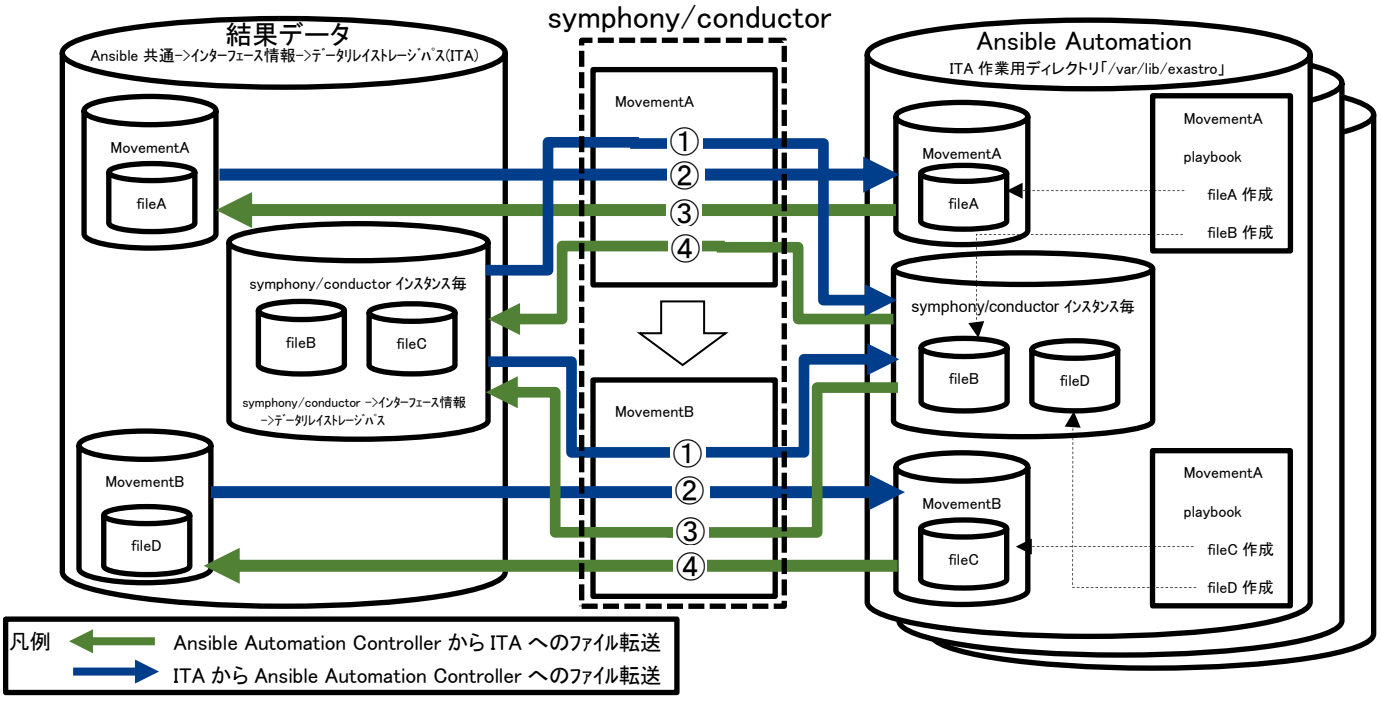
下記 ITA 独自変数を利用してファイル出力をする Plyabook を含む Movement をクラスタ構成の Ansible Automation Controller で作業実行した場合の留意事項について記載します。

対象の ITA 独自変数

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

(1) ITA 独自変数を利用して作成したファイルの取り扱い

ITA 独自変数を利用して作成したファイルの出力先ディレクトリパスは Ansible Automation Controller の ITA 作業用ディレクトリ配下「/var/lib/exastro」に設定されます。Movement 実行前に結果データを Ansible Automation Controller の ITA 作業用ディレクトリ配下「/var/lib/exastro」にファイル転送します。Movement 実行でここに作成されたファイルは、Movement 実行後に各 Ansible Automation Controller より結果データに上書きモードでファイル転送します。同一ファイル名でファイルを作成している場合、更新したファイルが結果データに正しく反映されない場合があります。



① Movement を symphony/conductor から実行している場合、Movement 実行前に該当 symphony/conductor インスタンス配下のファイルを Ansible Automation Controller の ITA 作業用ディレクトリ配下にファイル転送

② Movement 実行前に該当 Movement 配下のファイルを Ansible Automation Controller の ITA 作業用ディレクトリにファイル転送

③ Movement を symphony/conductor から実行している場合、Movement 実行後に Ansible Automation Controller の ITA 作業用ディレクトリの該当 Movement で作成したファイルを結果データにファイル転送

④ Movement 実行後に Ansible Automation Controller の ITA 作業用ディレクトリの該当 symphony/conductor インスタンス配下に作成したファイルを結果データにファイル転送

(2) 留意事項

① ファイル名は ansible「__loginhostname__」を含めるなどして、Movement に紐づいているターゲットホスト毎に同一ファイル名に出力しないように工夫して下さい。

② symphony/conductor から実行する場合、複数の Movement で同一ファイル名への出力しないよう工夫して下さい。

8.5 実行時データ削除で削除されるデータリソース

インターフェース情報の実行時データ削除で「削除する」を選択した場合に、削除されるデータリソースを以下に列挙します。

表 8.5-1 実行時データ削除で削除されるデータリソース一覧(Ansible Automation Controller 側)

データリソース	リソース タイプ	実行エンジン		備考
		Ansible Tower3.x	Ansible Automation Controller4.x	
ITA 作業用ディレクトリ /var/lib/exastro/ita_<区分>_executions_10 桁の作業番号	ディレクトリ	○	○	
SCM 管理ディレクトリ /var/lib/awx/projects/ita_<区分>_executions_10 桁の作業番号	ディレクトリ	○	※	※プロジェクトリソースを削除により削除される
インベントリ リソース名: ita_<区分>_executions_inventory_10 桁の作業番号_通番	リソース	○	○	
認証情報 リソース名: ita_<区分>_executions_credential_10 桁の作業番号_通番 ita_<区分>_executions_vault_credential_10 桁の作業番号 ita_<区分>_executions_git_credential_10 桁の作業番号	リソース	○	○	
プロジェクト リソース名: ita_<区分>_executions_project_10 桁の作業番号	リソース	○	○	
ジョブテンプレート リソース名: ita_<区分>_executions_jobtpl_10 桁の作業番号_通番	リソース	○	○	
ワークフロージョブテンプレート リソース名: ita_<区分>_executions_workflowtpl_10 桁の作業番号	リソース	○	○	
ジョブ リソース名: Job 番号―ita_<区分>_executions_workflowtpl_10 桁の作業番号 Job 番号―ita_<区分>_executions_jobtpl_10 桁の作業番号	リソース	○	○	

表 8.5-2 実行時データ削除で削除されるデータリソース一覧(ITA 側)

データリソース	リソース タイプ	実行エンジン		備考
		Ansible Tower3.x	Ansible Automation Controller4.x	
Git リポジトリ ITA インストールディレクトリ/ita-root/repositorys/ansible_driver/<区分>_10 桁の作業番号	Git リポジトリ	○	○	ディレクトリごと削除

区分: legacy/legacy_role/pioneer