ライフ サイクルの拡張性

2020 年 12 月 28 日

vRealize Automation 7.4

最新の技術ドキュメントは、 VMware の Web サイト （https://docs.vmware.com/jp/）

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

ヴイエムウェア株式会社
105-0013　東京都港区浜松町 1-30-5
浜松町スクエア 13F
www.vmware.com/jp

目次

ライフ サイクルの拡張性

1

vRealize Automation および vRealize Orchestrator を使用して、IaaS マシンのライフ サイクル管理方法を拡張することができます。

vRealize Automation を拡張するには、提供済みの vRealize Orchestrator ワークフローを使用し、カスタム ワークフローを作成する必要があります。

この章には、次のトピックが含まれています。

マシンの拡張性の概要

特にミッション クリティカルなシステムの場合、新しいマシンのプロビジョニングまたは廃止には通常、DNS サーバ、ロード バランサー、CMDB、IP アドレス管理システム、その他のシステムなど、複数の異なる管理システムの操作が必要です。

マシン ライフサイクルの拡張性

ワークフロー スタブと呼ばれる IaaS 状態変更ワークフローを活用することで、さまざまな事前に設定された IaaS ライフサイクル ステージでカスタム ロジックを挿入することができます。ワークフロー スタブを使用して、外部管理システムとの双方向統合のために vRealize Orchestrator を呼び出すことができます。

状態変更ワークフローを作成することで、IaaS メイン ワークフローが特定の状態になる前にワークフローの実行をトリガできます。たとえば、カスタム ワークフローを作成して外部データベースと統合し、マシン ライフサイクルのさまざまなステージで情報を記録することができます。

- メイン ワークフローが MachineProvisioned 状態になる前に実行され、マシンの所有者や承認者などの情報を記録するカスタム ワークフローを作成します。
- マシンが MachineDisposing 状態になる前に実行され、マシンが破棄された時間と、最後のデータ収集時や最後のログオン時などのリソース使用率などのデータを記録するカスタム ワークフローを作成します。

このメイン ワークフローの図は、プライマリ ワークフローの主な状態を示しています。黄色で強調表示されている状態が IaaS ワークフロー スタブを使用してカスタマイズできる状態です。[カスタマイズ可能な状態変更ワークフロー]の表には、使用可能なワークフロー スタブ、対応するメイン ワークフローの状態、各状態で使用してマシン ライフサイクルを拡張できるカスタム ロジックの例が一覧表示されています。

図 1-1. マシンのプロビジョニングのためのメイン ワークフローの状態

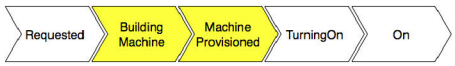


図 1-2. マシンのインポートのためのメイン ワークフローの状態

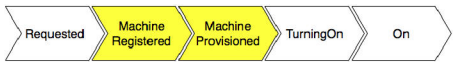


図 1-3. マシン リースの有効期限のためのメイン ワークフローの状態

図 1-4. マシンの削除のためのメイン ワークフローの状態

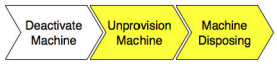


表 1-1. カスタマイズ可能な状態変更ワークフロー

メイン ワークフローの状態	カスタマイズ可能なワークフロー名	拡張性の例
BuildingMachine	WFStubBuildingMachine	ハイパーバイザーで作成されるマシンを準備します。構成管理データベース (CMDB) レコードを作成し、外部システムに呼び出してマシンに IP アドレスを割り当ててから、マシンの処分中に WFStubMachineDisposing を使用してその IP アドレスをプールに返します。
RegisterMachine	WFStubMachineRegistered	インポートされたマシンをアプリケーション プロビジョニング ツールに追加して、更新を受け取り、コンプライアンス チェックを実行します。
MachineProvisioned	WFStubMachineProvisioned	マシンはハイパーバイザーに存在し、ゲスト エージェントのカスタマイズなど、追加のカスタマイズはこの時点で完了します。このワークフロー スタブを使用して、構成管理データベース (CMDB) レコードを DCHP IP アドレスとストレージ情報で更新します。WFStubMachineProvisioned を使用して行われたカスタマイズは通常、WFStubUnprovisionMachine を使用することで元に戻ります。

表 1-1. カスタマイズ可能な状態変更ワークフロー （続き）

メイン ワークフローの状態	カスタマイズ可能なワークフロー名	拡張性の例
Expired	WFStubMachineExpired	期限切れのマシンを低コストのストレージに移動してアーカイブ コストを低減し、CMDB レコードと課金システムを更新してストレージとコストの変更を反映します。
UnprovisionMachine	WFStubUnprovisionMachine	Active Directory アカウントからマシンを削除します。WFStubMachineProvisioned を使用して行われたカスタマイズは通常、WFStubUnprovisionMachine を使用することで元に戻ります。
Disposing	WFStubMachineDisposing	プールに IP アドレスを返します。

ライフサイクルの拡張シナリオの選択

vRealize Orchestrator または vRealize Automation Designer を使用してマシン ライフサイクルを拡張できます。

vRealize Automation Designer を使用して vRealize Orchestrator を呼び出すか、vRealize Orchestrator を直接使用して、マシン ライフサイクルを拡張できます。どちらの方法でも、カスタムの vRealize Orchestrator ワークフローを作成して状態変更ワークフロー スタブに挿入することで、IaaS マシン ライフサイクルの事前定義されているステージにカスタム ロジックを追加できます。ただし、カスタムの状態変更ロジックを特定のブループリントに制限できるのは vRealize Orchestrator を直接を使用している場合に限られ、ワークフローの実行を特定の Distributed Execution Manager (DEM) に制限できるのは vRealize Automation Designer のみです。

注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。

表 1-2. ライフサイクルの拡張シナリオの選択

シナリオ	手順
事前定義されている IaaS マシン ライフサイクル ステージにカスタム ロジックを追加し、そのカスタム ロジックを特定のブループリントに適用します。	vRealize Orchestrator チェックリストを使用してマシンのライフサイクルを延長する
事前定義された IaaS マシン ライフサイクル ステージにカスタム ロジックを追加し、そのカスタム ロジックをすべてのブループリントにグローバルに適用します。	vRealize Automation Designer チェックリストを使用してマシンのライフサイクルを延長する
vRealize Automation Designer のスキルを使用して、ワークフローの実行を特定の Distributed Execution Manager に制限します。スキルはタグのようなもので、ワークフローと DEM ワーカー インスタンスの両方に適用できます。 たとえば、クラウド プロビジョニング ワークフローを、Amazon の URL へのネットワーク アクセスが可能なホストで実行されている特定の DEM に制限できます。	スキルを使用したワークフローと DEM ワーカーの関連付け

vRealize Orchestrator の使用によるマシン ライフサイクルの拡張

カスタム vRealize Orchestrator ワークフローを作成してから、vRealize Orchestrator を使用してカスタム ワークフローを固有のブループリントからビルドされたマシンのライフサイクルに挿入することで、IaaS マシン ライフサイクルの事前に設定されたステージにカスタム ロジックを挿入できます。

vRealize Orchestrator チェックリストを使用してマシンのライフサイクルを延長する

「vRealize Orchestrator チェックリストを使用してマシンのライフサイクルを延長する」では、IaaS マシンのライフサイクルをカスタマイズするために vRealize Orchestrator をインストールおよび構成するのに必要な手順についての概要を提供します。

表 1-3. vRealize Orchestrator チェックリストを使用してマシンのライフサイクルを延長する

タスク	詳細
❑ vRealize Orchestrator に vRealize Automation ホストを構成します。	vRealize Automation ホストの追加
❑ vRealize Orchestrator に IaaS ホストを構成します。	IaaS ホストの追加
❑ IaaS マシンのライフサイクルを延長するために vRealize Orchestrator カスタマイズをインストールします。	vRealize Orchestrator カスタマイズのインストール
❑ vRealize Orchestrator インスタンスに vRealize Automation エンドポイントを作成します。	vRealize Orchestrator エンドポイントの作成
❑ vRealize Automation プラグイン ライブラリの拡張性サブディレクトリ内に提供されたワークフロー テンプレートを使用して、マシンのライフサイクル中に実行するカスタム vRealize Orchestrator ワークフローを作成します。単一のラッパー ワークフローでネストしている場合は、同一のブループリントに対して、複数のワークフローを同じ状態で実行することができます。	vRealize Orchestrator でワークフローを開発する場合の詳細については、vRealize Orchestrator のドキュメントを参照してください。vRealize Automation 連携のための vRealize Orchestrator 開発のトレーニングについては、VMware Education で利用可能なトレーニング コースおよび VMware ラーニングで提供されている教材を参照してください。
❑ 提供されるワークフローを実行して、IaaS ワークフロー スタブにカスタム ワークフローを挿入し、IaaS ワークフロー スタブを呼び出すブループリントを構成します。 **注：** ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。	ブループリントとその仮想マシンへの状態変更ワークフローの割り当て

マシンを拡張するための vRealize Automation プラグインの設定

vRealize Automation および IaaS ホストを設定し、マシンを拡張するためにカスタマイズを行い、vRealize Orchestrator インスタンス用に vRealize Automation エンドポイントを作成します。

vRealize Automation ホストの追加

vRealize Automation ホストを追加するワークフローを実行し、ホストの接続パラメータを設定することができます。

手順

1 Orchestrator クライアントのドロップダウン メニューから、[実行] または [設計] を選択します。

2 [ワークフロー] ビューをクリックします。

3 [ライブラリ] - [vRealize Automation] - [構成] の順に展開します。

4 [vRA ホストの追加] ワークフローを右クリックして [ワークフローの開始] の順に選択します。

5 [ホスト名] テキスト ボックスに一意のホストの名前を入力します。

6 [ホスト URL] テキスト ボックスにホストの URL アドレスを入力します。

例： *https://hostname*。

7 （必須） [テナント] テキスト ボックスにテナントの名前を入力します。

テナント用のプラグインの全機能を使用するには、テナントごとに専用の vRealize Automation ホストを作成します。

8 ユーザーへの確認なしで自動的に SSL 証明書をインストールするかどうかを選択します。

9 （オプション）vRealize Orchestrator が vRealize Automation からの接続または応答を待機する時間を設定するには、[接続タイムアウト（秒）] および [操作タイムアウト（秒）] テキスト ボックスにタイムアウト間隔を入力します。

10 ホストへの接続の種類を [セッション モード] ドロップダウン メニューから選択します。

オプション	アクション
共有セッション	vRealize Automation ユーザーの認証情報を、[認証ユーザー名] と [認証パスワード] のテキスト ボックスに入力します。
ユーザー セッション単位	現在ログインしているユーザーの認証情報を使用して接続します。vRealize Automation システム管理者の認証情報を使用して Orchestrator クライアントにログインしている必要があります。 このオプションを外部の vRealize Orchestrator サーバで使用するには、Orchestrator サーバを vRealize Automation コンポーネント レジストリに登録する必要があります。 **注：** 外部 vRealize Orchestrator サーバをコンポーネント レジストリに登録するには、Orchestrator が vRealize Automation を認証プロバイダとして使用するように設定する必要があります。詳細については、『VMware vRealize Orchestrator のインストールおよび構成』を参照してください。

11 [送信] をクリックします。

次のステップ

vRealize Automation のインフラストラクチャ管理ホストを追加します。

IaaS ホストの追加

vRealize Automation ホストの IaaS ホストを追加するワークフローを実行し、接続パラメータを設定できます。

手順

1. Orchestrator クライアントのドロップダウン メニューから、[実行] または [設計] を選択します。
2. [ワークフロー] ビューをクリックします。
3. [ライブラリ] - [vRealize Automation] - [インフラストラクチャ管理] - [構成] の順に展開します。
4. [IaaS ホストの追加] を右クリックして [ワークフローの開始] の順に選択します。
5. [vCAC ホスト] ドロップダウン メニューから、IaaS ホストを構成する vRealize Automation ホストを選択します。
6. [ホスト名] テキスト ボックスに一意のホストの名前を入力します。
7. Model Manager がインストールされているマシンの URL を入力します。

   例：https://*model_manager_machine.com*
8. SSL 証明書をインストールするには、[はい] を選択します。
9. Model Manager マシンへのアクセスにプロキシを使用するには、[はい] を選択します。

   このオプションを選択した場合は、次のページでプロキシ ホストとプロキシ ポートを入力する必要があります。
10. [次へ] をクリックします。
11. 明示的プロキシを設定している場合は、プロキシ ホストの URL とポートを入力します。
12. [次へ] をクリックします。
13. 独自のタイムアウト値を設定するには、[いいえ] をクリックします。
14. （オプション）vRealize Orchestrator が vRealize Automation からの接続または応答を待機する時間を設定するには、[接続タイムアウト（秒）] および [操作タイムアウト（秒）] テキスト ボックスにタイムアウト間隔を入力します。
15. [次へ] をクリックします。
16. ホストの認証タイプを選択します。

オプション	説明
SSO	vCenter Single Sign-On を使用する場合に選択します。
NTLM	Active Directory インフラストラクチャが NTLM 認証に依存している場合にのみ、このオプションを選択して NT LAN Manager (NTLM) プロトコルベースの認証を有効にします。 このオプションには、NTLM 認証情報と認証オプションが必要です。

17 NTLM を選択した場合は、[次へ] をクリックし、ワークステーション　マシン名前と NetBIOS ドメイン名を入力します。

18 [送信] をクリックします。

vRealize Orchestrator カスタマイズのインストール

ワークフローを実行して、カスタマイズされた状態変更ワークフロー スタブと Orchestrator メニュー操作ワークフローをインストールできます。

注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。

手順

1 Orchestrator クライアントのドロップダウン メニューから、[実行] または [設計] を選択します。

2 [ワークフロー] ビューをクリックします。

3 [ライブラリ] - [vCloud Automation Center] - [インフラストラクチャ管理] - [拡張性] - [インストール] の順に選択します。

4 [vCO カスタマイズのインストール] ワークフローを右クリックして [ワークフローの開始] を選択します。

5 IaaS ホストを選択します。

6 [次へ] をクリックします。

7 インストールする 1 つ以上の状態変更ワークフロー スタブを選択して、カスタム ロジックを追加するライフサイクル ステージを選択します。

8 [送信] をクリックします。

vRealize Orchestrator エンドポイントの作成

vRealize Orchestrator サーバに接続する vRealize Orchestrator エンドポイントを作成できます。

複数のエンドポイントを設定して、別々の vRealize Orchestrator サーバに接続できますが、各エンドポイントに優先度を設定する必要があります。

vRealize Orchestrator ワークフローを実行するとき、vRealize Automation は、最初に最も優先度の高い vRealize Orchestrator エンドポイントの使用を試みます。 そのエンドポイントにアクセスできない場合は、vRealize Orchestrator サーバがワークフローを実行できるようになるまで、次に優先度の高いエンドポイントの使用を試みます。

前提条件

- IaaS 管理者として vRealize Automation にログインします。

手順

1 [インフラストラクチャ] - [エンドポイント] - [エンドポイント] を選択します。

2 [新規] - [オーケストレーション] - [vRealize Orchestrator] の順に選択します。

3 名前と説明（説明は任意）を入力します。

4 vRealize Orchestrator サーバの完全修飾名または IP アドレスおよび vRealize Orchestrator ポート番号を含む URL を入力します。

   転送プロトコルは HTTPS にする必要があります。 ポートが指定されない場合は、デフォルト ポート 443 が使用されます。

   vRealize Automation アプライアンスに組み込まれたデフォルトの vRealize Orchestrator インスタンスを使用するには、**https://*vrealize-automation-appliance-hostname*:443/vco** と入力します。

5 vRealize Orchestrator の認証情報を [ユーザー名] と [パスワード] テキスト ボックスに入力し、vRealize Orchestrator エンドポイントに接続します。

   使用する認証情報には、IaaS から呼び出す vRealize Orchestrator ワークフローに対する実行権限が必要です。

   vRealize Automation アプライアンスに組み込まれたデフォルトの vRealize Orchestrator インスタンスを使用する場合、ユーザー名は **administrator@vsphere.local**、パスワードは、SSO の構成時に指定した管理者パスワードになります。

6 [優先度] テキスト ボックスに 1 以上の整数を入力します。

   値が小さいほど、優先度が高くなります。

7 （オプション）[プロパティ] をクリックし、提供されたカスタム プロパティ、プロパティ グループ、またはエンドポイント用の独自のプロパティ定義を追加します。

8 [OK] をクリックします。

vRealize Orchestrator を使用した IaaS ワークフローのカスタマイズ

vRealize Orchestrator で 1 つのワークフローを使用して、カスタム ロジックを IaaS ワークフロー スタブに挿入し、カスタマイズされたライフサイクルをマシン ブループリントに割り当てます。

注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。

カスタムの vRealize Orchestrator ワークフローは、文字列の入力を受け入れるように設計する必要があります。カスタム ワークフローで複合データ タイプが求められる場合は、この複合値を検索して文字列に変換するラッパー ワークフローを作成します。ラッピング ワークフローの例については、[ライブラリ] - [vRealize Automation] - [インフラストラクチャ] - [拡張性] で提供されるサンプル ワークフロー テンプレートを参照してください。

ブループリントとその仮想マシンへの状態変更ワークフローの割り当て

カスタム ワークフローを状態変更ワークフロー スタブに関連付け、これをブループリントに割り当てることによって、メイン マシン ワークフローの特定のステージで、カスタムの vRealize Orchestrator ワークフローが実行されるようにします。

注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。

前提条件

マシンのライフサイクルで実行するカスタム ワークフローを作成するには、vRealize Automation プラグイン ライブラリの Extensibility サブディレクトリにあるワークフロー　テンプレートを使用します。

手順

1. Orchestrator クライアントのドロップダウン メニューから、[実行] または [設計] を選択します。
2. [ワークフロー] ビューをクリックします。
3. [ライブラリ] - [vRealize Automation] - [インフラストラクチャ] - [拡張性] の順に選択します。
4. [ブループリントとその仮想マシンへの状態変更ワークフローの割り当て] ワークフローを右クリックして、[ワークフローの開始] を選択します。
5. [有効にする vCAC ワークフロー スタブ] のドロップダウン メニューからスタブを 1 つ選択することで、ワークフローを実行するライフサイクル ステージを選択します。
6. IaaS ホストを選択します。
7. [次へ] をクリックします。
8. ワークフローを割り当てるブループリントを選択します。
9. このブルー プリントを使用してプロビジョニングされた既存のマシンに、これらのワークフローを適用するかどうかを選択します。
10. マシンのライフサイクルで実行するワークフローを選択します。
11. どのワークフローの入力値をカスタム プロパティとしてマシンに追加するかを構成します。
    a　vCO ワークフローの入力値をブループリントのプロパティとして追加します。
    b　最後の vCO ワークフローの実行入力値をブループリントのプロパティとして追加します。
12. [送信] をクリックします。

vRealize Automation を拡張するためのワークフロー サブスクリプションの設定

イベント ブローカ サービスを使用するワークフロー サブスクリプションを作成して vRealize Automation のイベント メッセージ用に登録されているサービスを監視し、サブスクリプションの条件が一致した場合に、指定された vRealize Orchestrator ワークフローが実行されるようにします。サブスクリプションを設定するには、イベント トピック、トリガ条件およびトリガされたときに実行するワークフローを指定します。

テナント管理者は、自分のテナントに固有のワークフロー サブスクリプションを作成および管理できます。

システム管理者は、システム ワークフロー サブスクリプションを作成および管理できます。作成したシステム ワークフロー サブスクリプションは、任意のテナントのイベントとシステム イベントでアクティブになります。

vRealize Automation で提供されるイベント トピック

イベント トピックは、他のサービスによってイベント ブローカ サービスに送信されるイベント メッセージのタイプを説明します。イベント トピックを選択し、そのトピックに基づいてワークフロー サブスクリプションを構成します。

表 1-4. イベント トピック

イベント トピック名	説明	サービス
ブループリント コンポーネントの完了	複合ブループリントに含まれるブループリント コンポーネントがプロビジョニングを完了します。このコンポーネントは複合ブループリントに含まれるブループリントです。	composition-service
ブループリント コンポーネントの申請	複合ブループリントに含まれるブループリント コンポーネントが申請されます。このコンポーネントは複合ブループリントに含まれるブループリントです。	composition-service
ブループリント構成	ブループリントを作成、更新、または削除します。	composition-service
申請が完了したブループリント	複合ブループリントがプロビジョニングを完了しました。このイベント トピックにはすべてのブループリント コンポーネントが含まれています。スタンドアロン XaaS ブループリントは含まれてません。	composition-service
申請済みブループリント	複合ブループリントが申請されます。このイベント トピックに XaaS ブループリントは含まれてません。	composition-service
ビジネス グループ構成	ビジネス グループを作成、更新、または削除します。	identity
コンポーネントのアクションの完了	展開アクションの申請時に、展開されたブループリント コンポーネントでアクションが実行されました。	composition-service
コンポーネントのアクションの申請	展開アクションの申請時に、展開されたブループリント コンポーネントで実行されるアクションが申請されます。	composition-service

表 1-4. イベント トピック （続き）

イベント トピック名	説明	サービス
展開アクションの完了	展開されたブループリントでのアクションの実行（すべてのコンポーネント アクションの実行など）が完了しました。	composition-service
展開アクションの申請	展開されたブループリントでのアクションが申請されます。	composition-service
EventLog デフォルト イベント	標準エントリをイベント ログに追加します。ログ エントリは、サブスクライバには配布されません。	eventlog-service
IP アドレス管理の IP ライフサイクル イベントの完了	IP 割り当てまたは割り当て解除申請が完了します。	ipam-service
マシンのライフサイクル	指定した IaaS アクションを、プロビジョニング対象マシンで実行します。	iaas-service
マシン プロビジョニング	IaaS マシンのプロビジョニング処理が進行中です。	iaas-service
オーケストレーション サーバ構成	vRealize Orchestrator サーバ設定を作成、更新、削除したり、異なるデフォルト インスタンスを使用するように変更します。	o11n-gateway-service
オーケストレーション サーバ構成 (XaaS): 廃止	vRealize Orchestrator サーバ設定を作成、更新、削除したり、異なるデフォルト インスタンスを使用するように変更します。	advanced-designer-service
事後承認	事後承認ポリシーのレベルを、イベント サブスクリプション オプションを使用するように設定します。	approval-service
事前承認	事前承認ポリシーのレベルを、イベント サブスクリプション オプションを使用するように設定します。	approval-service
リソース回収完了イベント	リース期限が切れたリソースを回収します。	management-service

ワークフロー サブスクリプションおよびイベント ブローカの用語集

ワークフロー サブスクリプションおよびイベント ブローカ サービスを扱っていると、サブスクリプションおよびイベント ブローカ サービスに固有のいくつかの用語に出会うことがあります。

表 1-5. ワークフロー サブスクリプションおよびイベント ブローカの用語集

用語	説明
イベント トピック	同じ論理的な意図と構造を持つ一連のイベントを表します。各イベントは、イベント トピックの 1 つのインスタンスです。
イベント	プロデューサまたはプロデューサによって管理されるエンティティのいずれかで、状態が変更されたことを示します。このイベントは、イベントの発生に関する情報を記録するエンティティです。

表 1-5. ワークフロー サブスクリプションおよびイベント ブローカの用語集 （続き）

用語	説明
メッセージ	さまざまなサービスおよびコンポーネント間のイベントに関する情報を伝達します。たとえば、プロデューサからイベント ブローカ サービスに、またはイベント ブローカ サービスからサブスクライバに情報を伝えます。
イベント ブローカ サービス	プロデューサが公開したメッセージを、登録した利用者に送るサービスです。
ペイロード	イベント データを表します。
サブスクリプション	イベントに関する通知を受け取る意思があるサブスクライバが、イベント トピックに登録し、通知をトリガする基準を定義します。
サブスクライバ	サブスクリプションの定義に基づいて、イベント ブローカ サービスに公開されたイベントを利用します。サブスクライバは、利用者とも呼ばれます。
プロバイダ	イベント トピックをイベント ブローカ サービスに登録します。
プロデューサ	イベント ブローカ サービスにイベントを公開します。
システム管理者	API または vRealize Automation プラグインを使用して、テナント ワークフロー サブスクリプションとシステム ワークフロー サブスクリプションの作成、読み取り、更新、および削除を行うための権限を持ったユーザーです。vRealize Automation にはシステム管理者用のユーザー インターフェイスはありません。
テナント管理者	自分のテナントのテナント ワークフロー サブスクリプションの作成、読み取り、更新、および削除を行うための権限を持ったユーザーです。
ワークフロー サブスクリプション	vRealize Orchestrator ワークフローをトリガするイベント トピックと条件を指定します。
システム ワークフロー サブスクリプション	システム イベントと全テナントのイベントに対処する特別なワークフロー サブスクリプションです。
テナント ワークフロー サブスクリプション	同じテナント内のイベント用に vRealize Orchestrator ワークフローをトリガする条件を指定する、特別なワークフロー サブスクリプションです。

ブロック可能および応答可能なイベント トピック

イベント トピックでは、ブロック可能なイベントと応答可能なイベントがサポートされる場合があります。ワークフロー サブスクリプションの動作は、トピックでこれらのイベント タイプがサポートされるかどうか、またワークフロー サブスクリプションの設定方法によって異なります。

ブロック可能でないイベント トピック

ブロック可能でないイベント トピックでは、非ブロック サブスクリプションの作成のみが許可されます。非ブロック サブスクリプションは非同期的にトリガされるため、サブスクリプションのトリガが、設定された順序どおりに処理されないことがあります。ただし、トリガ イベントは必ず発生し、サブスクリプションに関連付けられている vRealize Orchestrator ワークフローは実行されます。非ブロック サブスクリプションは、トピックが応答可能な場合にのみ応答を返します。

ブロック可能なイベント トピック

一部のイベント トピックはブロックをサポートします。ワークフロー サブスクリプションがブロックとしてマークされている場合、最初のワークフローが完了するまで、設定された条件を満たすすべてのメッセージは、条件が一致する他のワークフロー サブスクリプションでは受信されません。同じイベント トピックに複数のブロック ワークフロー サブスクリプションがある場合は、各サブスクリプションに優先順位を設定します。

ブロック サブスクリプションは優先順位に従って実行されます。最も高い優先順位の値は 0（ゼロ）です。1 つのイベント トピックに優先順位レベルが同じ 2 つ以上のブロック サブスクリプションがある場合、各サブスクリプションの名前に基づいてアルファベット順に実行されます。すべてのブロック サブスクリプションが処理されたら、メッセージは、すべての非ブロック サブスクリプションに同時に送信されます。ブロック ワークフロー サブスクリプションは同期的に実行されるため、後続のワークフロー サブスクリプションが通知される際には、変更されたイベント ペイロードに更新済みのイベントが含まれます。

選択したワークフローや目標に応じて、1 つまたは複数のワークフロー サブスクリプションにブロックを適用します。

たとえば、2 つのプロビジョニング ワークフロー サブスクリプションがあり、2 つ目のワークフローが 1 つ目の結果を使用する場合について考えます。1 つ目は、プロビジョニング時にプロパティを変更し、2 つ目はファイル システム内の新しいプロパティ（仮想マシン名など）を記録します。ChangeProperty サブスクリプションの優先順位は 0 で、RecordProperty は ChangeProperty サブスクリプションの結果を使用するため、優先順位が 1 になります。仮想マシンのプロビジョニングが開始されると、ChangeProperty サブスクリプションが実行されます。RecordProperty サブスクリプションの条件はプロビジョニング後の条件に基づくため、RecordProperty サブスクリプションはメッセージによってトリガされます。ただし、ChangeProperty ワークフローはブロック ワークフローであるため、完了するまでメッセージは受信されません。名前が変更されて最初のワークフローが終了したら、2 つ目のワークフローが実行されてファイル システムの名前が記録されます。

イベント トピックがブロックをサポートする場合でも、ワークフロー サブスクリプションに、前のワークフローの結果を使用する後続のワークフローがない場合は、非ブロック ワークフロー サブスクリプションを作成できます。ワークフロー サブスクリプションがトリガされ、vRealize Automation や外部システムから操作を行わなくても、vRealize Orchestrator ワークフローが実行されます。

応答可能なイベント トピック

一部のイベント トピックは、登録済みサービスからの応答をサポートします。応答可能なイベント トピックを登録したサービスは、通常は、システムまたはユーザーからの操作の結果として、ワークフロー出力を提供する応答イベントを受け入れることができます。応答の出力パラメータは、元の応答可能なイベントを発行した vRealize Automation サービスによる処理を可能にするために、応答スキーマで定義された条件を満たす必要があります。たとえば、事前承認および事後承認のワークフロー サブスクリプションは応答可能です。外部システムに承認申請を送信するワークフローを作成する場合、vRealize Automation は（承認済みまたは却下済みの）応答を処理し、カタログ アイテムがプロビジョニングされるか、申請が却下されたことがユーザーに通知されます 。

応答は vRealize Orchestrator ワークフローの出力になるか、ワークフローがタイムアウトまたは失敗した場合には、エラーになることがあります。応答がワークフローの出力パラメータから行われる場合、応答は正しい応答スキーマ形式である必要があります。

ワークフロー サブスクリプション用に vRealize Orchestrator ワークフローを作成するためのベスト プラクティス

ワークフロー サブスクリプションは、特定のトピック スキーマに基づきます。サブスクリプションが vRealize Orchestrator ワークフローを確実に開始できるようにするために、正しい入力パラメータを使用してサブスクリプションを設定し、イベント データと連携できるようにする必要があります。

ワークフローの入力パラメータ

作成するカスタム ワークフローには、すべてのパラメータまたはペイロード内のすべてのデータを消費する 1 つのパラメータを含めることができます。

- 個々のパラメータを含めるには、1 つ以上のパラメータを設定します。名前およびタイプが、スキーマ内に指定された名前およびタイプと一致することを確認します。スキーマの複合タイプは、ワークフローで「プロパティ」として定義されています。
- 単一のパラメータを使用するには、1 つのパラメータを `Properties` のタイプで設定します。使いやすい任意の名前を指定できます。たとえば、パラメータ名として `payload` を使用することができます。

ワークフローの出力パラメータ

作成したカスタム ワークフローには、応答イベント トピック タイプに必要な後続のイベントに関連する出力パラメータを含めることができます。

イベント トピックで応答が見込まれている場合、ワークフローの出力パラメータは、応答スキーマと一致する必要があります。

ワークフロー サブスクリプションの設定

サブスクリプション オプションによって、vRealize Automation のイベント メッセージに基づいてワークフローを実行するタイミングが決まります。各オプションを使用してサブスクリプションを管理します。

サブスクリプションとは、ユーザーがイベント トピックのイベントに登録し、定義した条件と一致するトピックのイベントを受け取ったときにワークフローを実行するものです。

ワークフロー サブスクリプションを作成するには、テナント管理者である必要があります。すべてのワークフロー サブスクリプションはテナントに固有です。

ワークフロー サブスクリプションを管理するには、[管理] - [イベント] - [サブスクリプション] の順に選択します。

表 1-6. ワークフロー サブスクリプションのオプション

オプション	説明
新規	新しいサブスクリプションを作成します。
編集	選択したサブスクリプションを変更します。 サブスクリプションが公開済みの場合、保存された変更はすぐにアクティブになります。 公開済みまたは未公開のサブスクリプションでは、イベント トピックを編集したり、ブロック オプションを変更したりすることはできません。

表 1-6. ワークフロー サブスクリプションのオプション （続き）

オプション	説明
公開	サブスクリプションをアクティブにします。 イベント ブローカ サービスからのイベントが処理され、サブスクリプションの条件が評価されます。設定済みの条件が true の場合、ワークフローがトリガされます。
公開解除	サブスクリプションをドラフト状態に戻します。 サブスクリプションは環境内でアクティブではなくなり、イベントを受け取らなくなります。 サブスクリプションを再発行すると、サブスクリプションで新しいイベントが受信されるようになります。過去のイベントは受信されません。
削除	選択したサブスクリプションを削除します。

サブスクリプションへのイベント トピックの割り当て

イベント トピックは、vRealize Automation で提供されるイベントのクラスです。サブスクリプションを定義するイベント トピックを選択します。

イベント トピックは、同種のイベントをグループ化するためのカテゴリです。イベント トピックをサブスクリプションに割り当てると、そのサブスクリプションをトリガするイベントが定義されます。

手順

1 [管理] - [イベント] - [サブスクリプション] の順に選択します。

2 [新規] をクリックし、[イベント トピック] を選択します。

表 1-7. イベント トピックの詳細

イベント トピックの詳細	説明
トピック ID	イベント トピックの識別子。
名前	イベント トピックの名前。
説明	イベント トピックの説明。
公開者	このイベント トピックが登録されているサービスの名前。
ブロック可能	このイベント トピックのブロック サブスクリプションを作成するかどうかを示します。 ブロック サブスクリプションは、イベントのペイロードの変更、または同一のイベントにおいて 2 つ目のワークフローの結果が 1 つ目のワークフローの結果を使用する際に、カスタム ロジックを実行する場合に使用します。
応答可能	イベント トピック サブスクリプションが元のイベントを生成したサービスに返信イベントを発行できるかどうかを示します。値が [はい] の場合は、ワークフローが完了すると、元のイベントを公開したサービスに返信が送信されます。その返信には、vRealize Orchestrator ワークフローおよびエラー詳細の出力内容が含まれます。
スキーマ	イベントのペイロードの構造について記述します。 スキーマを使用すると、ペイロード情報を使用できるワークフローを作成することができます。

サブスクリプションへのワークフロー条件の割り当て

サブスクリプションに設定する条件によって、イベント データに基づいてワークフローの実行がトリガされるかどうかが決定されます。

ワークフロー条件を定義することにより、ワークフローの開始方法を制御できます。[条件に基づいて実行] を選択すると、次のようなタイプが使用可能になります。

- データ

  これには、選択したイベント トピックに固有の、イベント メッセージ内の情報が含まれます。たとえば、仮想マシンのライフサイクル イベント トピックの条件を作成する場合、データ フィールドはブループリントと仮想マシンに関連付けられます。事前承認イベント トピックを選択すると、データ フィールドは承認ポリシーに関連付けられます。

  また、ツリーの上部にあるテキスト ボックスにパスを入力することで、スキーマに含まれないフィールドに条件を追加することもできます。形式は **${PATH}** を使用します。PATH はスキーマ内のパスです。**~** を使用してノードを区切ります。たとえば、**${data~machine~properties~SomeCustomProperty}** のように指定します。

- コア イベント メッセージの値

  これには、イベント メッセージに関する一般的な情報が含まれます。たとえば、イベント タイプ、タイム スタンプ、ユーザー名などがあります。

前提条件

手順

1 [管理] - [イベント] - [サブスクリプション] の順に選択します。

2 [新規] をクリックし、[イベント トピック] を選択します。

3 [次へ] をクリックし、[ワークフロー条件] を定義します。

表 1-8. 条件タイプ

条件	説明
すべてのイベントで実行	このイベント トピックのメッセージが受信されると、選択したワークフローが実行されます。
条件に基づいて実行	イベント メッセージが検出され、設定された条件にイベントが一致する場合に、選択したワークフローが実行されます。 このオプションを選択した場合は、このサブスクリプション用に選択したワークフローをトリガするためのイベント データに基づいて、条件を定義する必要があります。 ■ [単一条件]。設定された条件節が true の場合にワークフローがトリガされます。 ■ [次のすべて]。すべての節が true で少なくとも 2 つの条件を指定した場合に、ワークフローがトリガされます。 ■ [次のいずれか]。少なくとも 1 つの節が true で少なくとも 2 つの条件を指定した場合に、ワークフローがトリガされます。 ■ [次を含まない]。いずれの節も true にならない場合に、ワークフローがトリガされます。 定数値に基づいて条件を作成する場合、値は大文字と小文字が区別されずに処理されます。たとえば、「ブループリント名に UNIX が含まれる」という条件に対し、ブループリント名に Unix が使用されている場合、この条件は正しく処理されます。 ブループリント名に一致するように条件名を変更するには、最初に同じ文字列を含まない値に値を変更する必要があります。たとえば、条件 UNIX を編集するには、値を xxxx に変更して保存した後、xxxx を Unix に変更して保存します。

サブスクリプションへのワークフローの割り当て

選択した vRealize Orchestrator ワークフローは、サブスクリプション条件が true として評価されると実行されます。

ワークフローは、特定の順で実行されたときに、仮想環境において特定のタスクまたは特定のプロセスを完了する ABX アクション、決定、および結果を組み合わせます。ワークフローでは、仮想マシンのプロビジョニング、バックアップ、通常のメンテナンスの実行、メール送信、SSH 操作、物理インフラストラクチャの管理、および他の汎用ユーティリティ操作などのタスクを実行します。ワークフローは、入力を機能に応じて受け付けます。ワークフローは他のワークフローを呼び出すこともできます。たとえば、いくつかの異なるワークフローにおいて、仮想マシンを開始するワークフローを再利用できます。

サブスクリプション内でワークフローをリンクすることで、発生したイベントに応答する手順を自動化することができます。これにより、ユーザーの介入なしでワークフローを実行し、結果を得ることができます。具体的には、仮想マシンのプロビジョニング ライフサイクル イベントに対してワークフローを実行する機能が追加されます。サブスクリプションの出力を再利用して、同じ状態のワークフローの間でデータを共有するすることもできます。同じライフサイクル状態に登録されている複数のワークフローは、出力ペイロードをマージできます。

前提条件

ワークフローは、[管理] - [vRO 構成] - [サーバ設定] のリストに示されているとおりに、vRealize Orchestrator の中に置く必要があります。

手順

1 [管理] - [イベント] - [サブスクリプション] の順に選択します。

2 [新規] をクリックし、[イベント トピック] を選択します。

3 [次へ] をクリックし、[ワークフロー条件] を定義します。

4 [次へ] をクリックし、サブスクリプションに適用する [ワークフロー] を選択します。

表 1-9. [ワークフロー] タブ

[ワークフロー] タブ	説明
ワークフローの選択	ワークフローに移動します。
選択されたワークフロー	入力パラメータや出力パラメータを含む、ワークフローに関する情報を表示します。この情報を使用して、これが目的のワークフローであることを確認できます。

ワークフロー サブスクリプションの詳細の定義

サブスクリプションの詳細では、サブスクリプションの処理方法を定義します。

追加のサブスクリプションの詳細を定義することにより、サブスクリプションをさらに構成およびカスタマイズできます。

手順

1 [管理] - [イベント] - [サブスクリプション] を選択します。

2 [新規] をクリックし、[イベント トピック] を選択します。

3 [次へ] をクリックし、[ワークフロー条件] を割り当てます。

4 [次へ] をクリックし、サブスクリプションに割り当てる [ワークフロー] を選択します。

5 [次へ] をクリックし、[ワークフロー サブスクリプションの詳細] を定義します。

表 1-10. ワークフローの詳細

詳細	説明
名前	デフォルトでは、表示される名前は、選択したワークフローの名前です。 この名前は、サブスクリプションのリストに表示されます。名前は、テナント内で一意である必要があります。
優先度	ブロック サブスクリプションが実行される順序です。 最も高い優先順位は 0 です。イベント トピックに、同じ優先順位を持つブロック ワークフロー サブスクリプションが複数ある場合、サブスクリプションはサブスクリプション名のアルファベット順に処理されます。 このオプションは、ブロック ワークフロー サブスクリプションのみで使用することができます。

表 1-10. ワークフローの詳細 （続き）

詳細	説明
タイムアウト（分）	ワークフローが失敗と判断されるまでの時間（分）を入力します。 ワークフローが許可された時間内で終了することができない場合、ワークフローはキャンセルされ、[優先度] リストにある次のサブスクリプションにメッセージが送信されます。 値を指定しない場合、タイムアウトの制限はありません。 ブロック可能または応答可能イベントへの応答が必要なサービスは、デフォルトでタイムアウト値があります。たとえば、IaaS プロビジョニングおよびライフ サイクルのイベント トピックは、30 分でタイムアウトとなります。この値は、IaaS サーバ上で設定されます。承認トピックでは、デフォルト値は 24 時間です。この値は、システム上で設定されています。
説明	デフォルトでは、ワークフローの説明が表示されます。
ブロック	ワークフローが応答を待機中に、同じイベント トピックの後続ワークフローがイベント メッセージを受信するのをブロックするかどうかを指定します。 ブロックが有効なサブスクリプションは、優先順位に基づき、同じイベント トピックのブロックが設定されていないサブスクリプションよりも前にメッセージを受信します。ワークフローが完了すると、メッセージは次の優先順位の高いブロック サブスクリプションに送信されます。すべてのブロック サブスクリプションの処理が完了すると、メッセージがすべての非ブロック サブスクリプションに一斉送信されます。 ブロックのオプションは、イベント トピックがブロック可能の場合にのみ使用できます。この情報は、[イベント トピック] タブで提供されます。 [イベント トピック] タブで、ブロックの可否が示されます。 ■ チェック ボックスを選択しない場合、イベント ブローカは後続のワークフローをブロックしません。 ■ チェック ボックスを選択すると、イベント ブローカはどのワークフロー サブスクリプションがこのイベントに適格であるかを事前設定された条件に基づいて計算し、優先順位に従ってワークフローを実行します。イベント ブローカは、次のワークフローを実行する前に、各ワークフローからの応答を待機します。既存のワークフローの実行で変更されたすべてのパラメータは、キュー内にある次のワークフローに渡されます。 応答を待機している間、システム応答を使用中は他のワークフローにはイベントが通知されません。 ワークフロー サブスクリプションが作成された後で、このオプションを変更することはできません。
ワークフローが失敗した場合は、処理を停止します。	ワークフロー サブスクリプションのブロックが失敗した場合、エラーが解決されるまで後続のワークフローは実行されません。エラー メッセージがイベント ログに追加され、電子メールが要求側ユーザーに送信されます。

6 [完了] をクリックします。

プロビジョニングおよびライフサイクル ワークフロー サブスクリプションの使用

プロビジョニングおよびライフサイクル ワークフロー サブスクリプションを作成すると、vRealize Orchestrator を使用して IaaS マシンの管理を拡張できます。プロビジョニング サブスクリプションは、プロビジョニング プロセスの処理を拡張します。ライフサイクル サブスクリプションは、ユーザーがプロビジョニング対象アイテムを管理しているときに、ワークフロー開発者が行う処理を拡張します。

IaaS サービス統合

ワークフロー サブスクリプションは、IaaS サービスによって生成されるメッセージをベースにしたカスタム vRealize Orchestrator ワークフローを実行するプロビジョニングまたはライフサイクルのイベント トピックに基づいて作成します。vRealize Automation には、IaaS 統合に使用できる 2 つのイベント トピックが含まれます。

- マシン プロビジョニング。IaaS マシンのプロビジョニングおよび削除の間にワークフローを実行する、ワークフロー サブスクリプションを作成します。
- マシンのライフサイクル。プロビジョニング対象のマシンで、所有しているユーザーが実行する管理アクションに関連するワークフローを実行する、ワークフロー サブスクリプションを作成します。

プロビジョニングおよびライフサイクル ワークフロー用の vRealize Orchestrator ワークフローの設定

vRealize Orchestrator ワークフローは、IaaS サービス メッセージをサポートするように設定する必要があります。

プロビジョニングおよびライフサイクル イベント トピックのスキーマ

マシン プロビジョニングとマシン ライフサイクル イベント トピックでは、同じライフサイクル スキーマを使用します。違いは、トリガ状態にあります。マシン プロビジョニングでは、プロビジョニングの状態とイベントに基づいてメッセージを受信し、マシン ライフサイクルでは、アクティブな状態とイベントに基づいてメッセージを受信します。一部のプロビジョニングの状態には BuildingMachine および Disposing があります。一部のライフサイクルの状態には InstallTools および Off があります。

イベント メッセージは、イベント データ ペイロードです。イベント データ ペイロードの構造を次に示します。

```
{
  machine : {
      id                : STRING,      /* IaaS machine ID */
      name              : STRING,      /* machine name */
      externalReference : STRING,      /* machine ID on the hypervisor */
      owner             : STRING,      /* machine owner */
      type              : INTEGER,     /* machine type: 0 – virtual machine; 1 – physical machine; 2
– cloud machine */
      properties        : Properties   /* machine properties, see notes below how to expose virtual
machine properties */
  },
  blueprintName   : STRING,      /* blueprint name */
  componentId     : STRING,      /* component id */
  componentTypeId : STRING,      /* component type id */
  endpointId      : STRING,      /* endpoint id */
```

```
  requestId       : STRING,      /* request id */
  lifecycleState  : {                                          /* see Life Cycle State
Definitions*/
      state : STRING,
      phase : STRING,
      event : STRING
  },
  virtualMachineEvent                 : STRING,     /* fire an event on that machine – only processed
by Manager Service as consumer */
  workflowNextState                   : STRING,     /* force the workflow to a specific state – only
processed by Manager Service as consumer */
  virtualMachineAddOrUpdateProperties : Properties, /* properties on the machine to add/update – only
processed by Manager Service as consumer */
  virtualMachineDeleteProperties      : Properties  /* properties to remove from the machine – only
processed by Manager Service as consumer */
}
```

vRealize Orchestrator パラメータは、名前とタイプによってイベント ペイロードにマップされます。

`virtualMachineEvent` と `workflowNextState` を出力パラメータとして使用する場合、入力する値は、イベントをトリガし現在の vRealize Orchestrator ワークフローを開始したワークフローの状態またはイベントを表す必要があります。入力可能なライフサイクルの状態とイベントを確認するには、VMPS メイン ワークフローのライフサイクルの状態およびマシン タイプ別のプロビジョニング ライフサイクルの状態を参照してください。

拡張性のカスタム プロパティの使用

仮想マシンのカスタム プロパティは、ライフサイクルの状態に対する拡張性カスタム プロパティとして指定しない限り、イベント ペイロードには含まれません。これらのプロパティは、IaaS のエンドポイント、予約、ブループリント、申請、およびカスタム プロパティをサポートするその他のオブジェクトに追加できます。

オブジェクトに追加するカスタム プロパティの形式は `Extensibility.Lifecycle.Properties.{workflowName}.{stateName}` です。

たとえば、仮想マシンの状態が BuildingMachine の場合に、非表示のプロパティと「Virtual」で始まるすべてのプロパティを含める場合は、ブループリントのマシンにカスタム プロパティを追加します。この例のカスタム プロパティの名前は `Extensibility.Lifecycle.Properties.VMPSMasterWorkflow32.BuildingMachine` で、値は、コンマで区切られた `__*` および `Virtual*` です。

二重のアンダー スコア (__*) には、非表示のプロパティが含まれています。値 `Virtual*` には、virtual で始まるすべてのプロパティが含まれています。アスタリスク (*) はワイルドカードで、値として単独で使用することができますが、この方法でワイルドカードを使用すると大量のデータが転送されることになります。

カスタム プロパティを含む連続でトリガされる複数のワークフロー サブスクリプションがある場合は、ペイロード チェックにカスタム プロパティが保持されていることを確認するためのワークフローに適切なエントリを含める必要があります。

表 1-11. カスタム プロパティを保持するタスクのエントリ

状態	タスクのエントリ
追加または更新されたカスタム プロパティ	`virtualMachineAddOrUpdateProperties = payload.virtualMachineAddOrUpdateProperties || new Properties();`
削除されたカスタム プロパティ	`virtualMachineDeleteProperties = payload.virtualMachineDeleteProperties || new Properties();`

ライフサイクルまたはプロビジョニング スキーマに基づいた vRealize Orchestrator ワークフローの作成

作成するカスタム ワークフローには、タイプが Properties の payload 入力パラメータが必要です。ワークフローが vRealize Orchestrator で実行されると、プロビジョニングまたはライフサイクル イベント データのペイロードがこのパラメータに配置されます。また、イベント ペイロードのフィールドに指定された名前とタイプに一致する個別の入力パラメータを含めることもできます。

ワークフロー サブスクリプションのライフサイクルの状態の定義

ライフサイクルの状態に基づいてワークフロー サブスクリプションの条件を設定する際、値の指定に次の定義が役立つ場合があります。

各メッセージには、IaaS のマシン状態の変更に基づいた lifecycleState 要素が含まれます。

この要素は、メッセージ内で次の構造になります。

```
lifecycleState : {
    state : STRING,
    phase : STRING,
    event : STRING
}
```

表 1-12. LifecycleState 要素

プロパティ	説明	形式および値	例
state	ワークフロー名と状態名が含まれます。	{workflowName}.{stateName}	■ VMPSMasterWorkflow32.Requested ■ VMPSMasterWorkflow32.MachineActivated ■ BasicVmWorkflow.BuildComplete
phase	メッセージをトリガした段階が含まれます。	PRE、POST、EVENT	■ PRE。この状態になるとイベントが公開されます。 ■ POST。この状態が終了するとイベントが公開されます。 ■ EVENT。この状態で IaaS イベントを受信するとイベントが公開されます。
event	イベントが含まれます。このプロパティはオプションで、段階が EVENT のときにのみ存在します。	{workflowName}.{stateName}.EVENT.{eventName}	■ VMPSMasterWorkflow32.Requested.EVENT.OnProvisionMachine ■ VMPSMasterWorkflow32.VMPSMasterWorkflow32.EVENT.OnBuildSuccess ■ BasicVmWorkflow.CreatingMachine.EVENT.OnCreatingMachineComplete

VMPS メイン ワークフローのライフサイクルの状態

VMPS メイン ワークフローのライフサイクルの状態は、申請から破棄までの、IaaS 仮想マシンのライフサイクルを表します。VMPS メイン ワークフローの状態とイベントは、ライフサイクル状態のイベントとライフサイクル状態の名前に基づいて、トリガ条件を作成する際に使用できます。

各仮想マシンは、4 つの基本ステージを進みます。

- 申請。承認が含まれます。
- プロビジョニング。作成、クローン作成、キックスタート、または WIM といったプロビジョニングのさまざまなタイプが含まれます。
- 管理。パワーオン、パワーオフ、またはスナップショット作成などのアクションが含まれます。
- 破棄。無効化、プロビジョニング解除、およびマシンの削除が含まれます。

これらの基本的なステージはメイン ワークフローに含まれています。次のイベント トピックの条件を作成する場合は、VMPSMasterWorkflow32 の状態を使用できます。

- マシン ライフサイクル
- マシン プロビジョニング

グローバル イベントの状態は、VMPS メイン ワークフローによってイベント ブローカに送信されるメッセージです。グローバル イベントは、いつでもトリガできます。

クライアントを登録してイベントをリッスンできますが、テーブル エントリにトリガー文字列値がない限り、イベントはスローされません。たとえば、イベント [トリガ文字列]（トピック）。

表 1-13. グローバル イベント

状態（トピック）	イベント [トリガ文字列]（トピック）
Global	■ onBuildFailure (Provision) ■ OnBuildSuccess (Provision) ■ OnFinalizeMachine [Destroy] (Provision) ■ OnForceUnregisterEvent [ForceUnregister] (Provision) ■ ReconfigureVM.Pending [ReconfigureVM.Pending] (Active) ■ ReconfigureVM.ExecutionUpdated (Active) ■ ReconfigureVM.RetryRequestMade (Active) ■ ReconfigureVM.Failed (Active) ■ ReconfigureVM.Successful (Active) ■ ReconfigureVM.Complete (Active) ■ ReconfigureVM.Canceled (Active)

アクティブなグローバル状態は、プロビジョニング対象マシンで実行できるアクションです。

表 1-14. アクティブ イベント

状態	イベント [トリガ文字列]（トピック）
Active	■ OnExpireLease [Expire] (Active) ■ OnForceExpire [ForceExpire] (Active) ■ onReprovision [Reprovision] (Active) ■ onResetBuildSuccess [ResetBuildSuccess] (Active)

メイン ワークフローでは、プロビジョニング イベントがマシン プロビジョニング ライフサイクル中に発生します。アクティブ イベントは、プロビジョニング対象マシンで実行できるアクションです。メイン ワークフローの図については、VMPS メイン ワークフローの例を参照してください。

マシンのタイプごとに独自のプロビジョニング ワークフローがあります。個々のマシン タイプに関する詳細については、マシン タイプ別のプロビジョニング ライフサイクルの状態を参照してください。

表 1-15. VMPSMasterWorkflow32 の状態およびイベント

状態（トピック）	イベント [トリガ文字列]（トピック）
BuildingMachine ■ Pre(Provision) ■ Post(Provision)	
DeactivateMachine ■ Pre(Provision) ■ Post(Provision)	
Disposing ■ Pre(Provision) ■ Post(Provision)	■ OnDisposeComplete(Provision) ■ OnDisposeTimeout(Provision) ■ OnUnregisterMachine [Unregister] (Provision)
Expired ■ Pre(Active) ■ Post(Active)	■ OnActiveExpiredMachine [ActivateExpiredMachine] (Active) ■ TurnOffFromExpired [TurnOffExpiredMachine] (Active)

表 1-15. VMPSMasterWorkflow32 の状態およびイベント （続き）

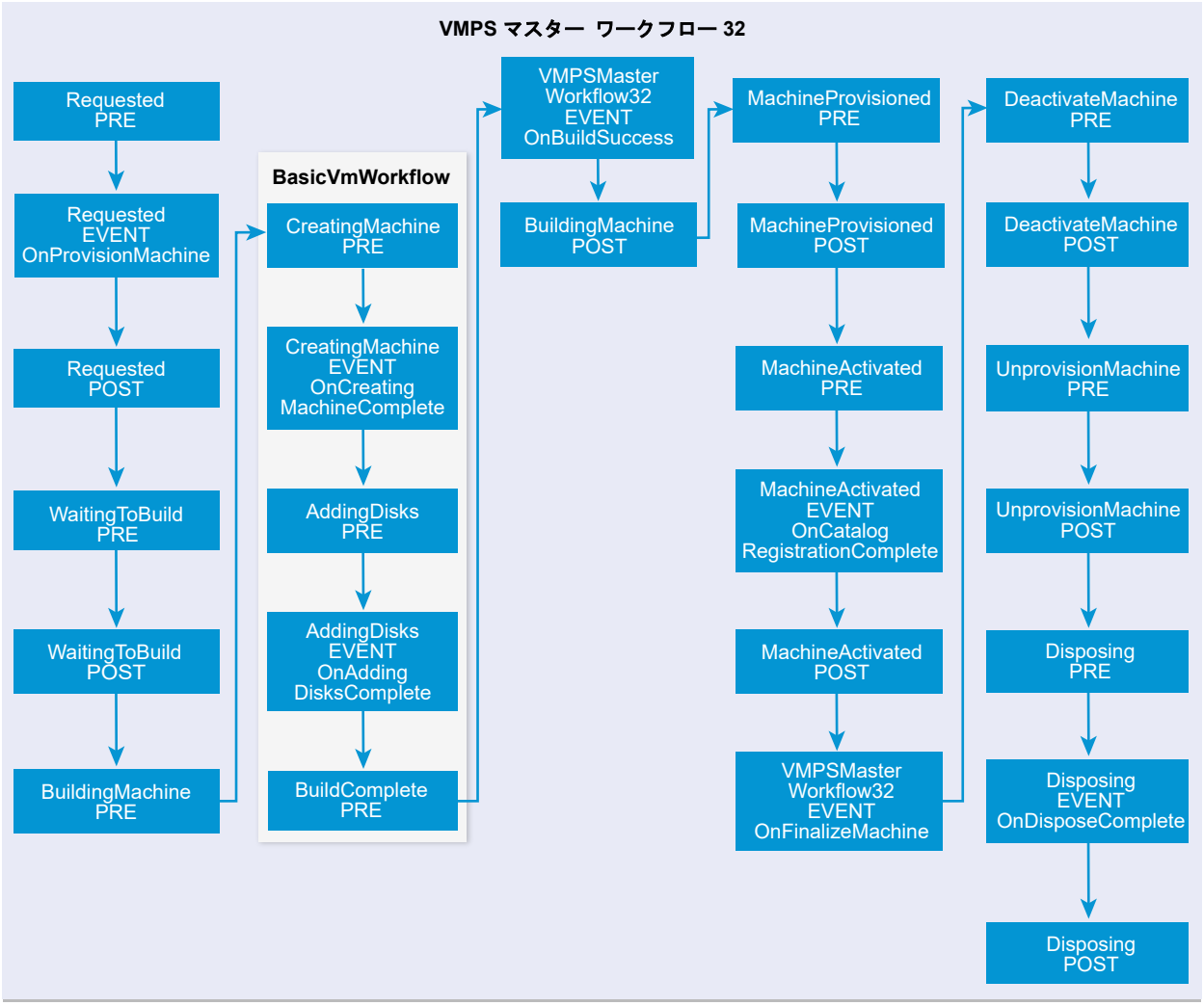
状態（トピック）	イベント [トリガ文字列]（トピック）
InstallTools ■ Pre(Active) ■ Post(Active)	■ InstallToolsComplete(Active) ■ TimeoutInstallTools(Active)
Leased	■ OnChangeLease (Active) ■ OnUpdateDescription (Active) ■ OnUpdateOwner (Active)
MachineActivated ■ Pre(Provision) ■ Post(Provision)	■ OnCatalogRegistrationComplete (Provision)
MachineProvisioned ■ Pre(Provision) ■ Post(Provision)	
Off ■ Pre(Active) ■ Post(Active)	■ OnForceOn [ForceOn] (Active) ■ OnResetOff [Turn Off] (Active) ■ OnTurnOn [Turn On] (Active)
On ■ Pre(Active) ■ Post(Active)	■ OnForceOff [ForceOff] (Active) ■ onInstallTools [InstallTools] (Active) ■ OnReboot [Reboot] (Active) ■ OnReset [Reset] (Active) ■ OnResetOn [Turn On] (Active) ■ OnShutdown [Shutdown] (Active) ■ OnSuspend [Suspend] (Active) ■ OnTurnOff [Turn Off] (Active)
Rebooting ■ Pre(Active) ■ Post(Active)	■ OnRebootComplete(Active) ■ TimoutFromReboot(Active)
RegisterMachine ■ Pre(Provision) ■ Post(Provision)	■ onRegisterComplete(Provision) ■ RegisterTimeout(Provision)
Requested ■ Pre(Provision) ■ Post(Provision)	■ OnProvisionMachine [Provision] (Provision)
Resetting ■ Pre(Active) ■ Post(Active)	■ OnResetComplete(Active) ■ TimoutFromReset(Active)
ShuttingDown ■ Pre(Active) ■ Post(Active)	■ OnShutdownComplete(Active) ■ TimoutFromShutdown(Active)
Suspending ■ Pre(Active) ■ Post(Active)	■ OnSuspendComplete(Active) ■ TimoutFromSuspend(Active)

表 1-15. VMPSMasterWorkflow32 の状態およびイベント （続き）

状態（トピック）	イベント [トリガ文字列]（トピック）
TurningOff ■ Pre(Active) ■ Post(Active)	■ OnTurningOffComplete(Active) ■ TimoutFromPowerOff(Active)
TurningOn ■ Pre(Active) ■ Post(Active)	■ OnTurningOnComplete(Active) ■ TimeoutPowerOn(Active)
UnprovisionMachine ■ Pre(Provision) ■ Post(Provision)	
WaitingToBuild ■ Pre(Provision) ■ Post(Provision)	

VMPS メイン ワークフローの例

VMPS ワークフローは、他のプロビジョニング ワークフローが埋め込まれているメイン ワークフローです。この例は、仮想マシンのライフサイクルを示す基本的な仮想マシン ワークフローを示しています。環境内の特定のワークフローを表すものではありません。



マシン タイプ別のプロビジョニング ライフサイクルの状態

マシン タイプ別のライフサイクルの状態は、特定の仮想マシン タイプに固有です。ワークフロー サブスクリプション用にトリガ条件を作成する際には、マスター ワークフローに加えて、プロビジョニング ワークフローの状態およびイベントを使用できます。

クライアントを登録してイベントをリッスンできますが、テーブル エントリにトリガー文字列値がない限り、イベントはスローされません。たとえば、イベント [トリガ文字列]（トピック）。

ブレード論理ベア メタル

状態（トピック）	イベント（トピック）
BuildFinished ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	

Opsware ベア メタル

状態（トピック）	イベント（トピック）
BuildFinished ■ Pre(Provision)	
OpswareRegister ■ Pre(Provision)	■ OnOpswareRegister(Provision)

クラウド プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
BuildComplete ■ Pre(Provision)	
CloudProvisioning ■ Pre(Provision)	■ OnCloudProvisioningTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	

アプリケーション サービス プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
AppServiceProvisioning ■ Pre(Provision)	■ OnAppServiceProvisioningTimeout(Provision)
BuildComplete ■ Pre(Provision)	
FailedProvisioning ■ Pre(Provision)	

基本的な仮想マシン ワークフロー

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	

Opsware 仮想

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildFinished ■ Pre(Provision)	
CreatingVM ■ Pre(Provision)	■ OnCreateVMComplete(Provision) ■ OnCreateVMTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
OpswareRegister ■ Pre(Provision)	■ OnOpswareRegister(Provision)

クラウド Linux キックスタート ワークフロー

状態（トピック）	イベント（トピック）
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
CustomizeOS ■ Pre(Provision)	■ OnCustomizeOSComplete(Provision) ■ OnCustomizeOSTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
InstallingOS ■ Pre(Provision)	■ OnInstallingOSComplete(Provision) ■ OnInstallingOSTimeout(Provision)

クローン ワークフロー

状態（トピック）	イベント（トピック）
BuildComplete ■ Pre(Provision)	
CloneMachine ■ Pre(Provision)	■ OnCloneMachineComplete(Provision) ■ OnCloneMachineTimeout(Provision)
CustomizeMachine ■ Pre(Provision)	■ OnCustomizeMachineComplete(Provision) ■ OnCustomizeMachineTimeout(Provision)
CustomizeOS	■ OnCustomizeOS(Provision) ■ OnCustomizeOSComplete(Provision) ■ OnCustomizeOSTimeout(Provision)
EjectCD ■ Pre(Provision)	■ OnEjectCDComplete(Provision) ■ OnEjectCDTimeout(Provision)

状態（トピック）	イベント（トピック）
FailedProvisioning ■ Pre(Provision)	
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeComplete(Provision) ■ OnFinalizeTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
InstallSoftware ■ Pre(Provision)	■ OnInstallSoftwareComplete(Provision) ■ OnInstallSoftwareTimeout(Provision)
MountCD ■ Pre(Provision)	■ OnMountCDComplete(Provision) ■ OnMountCDTimeout(Provision)
PostInstallSoftwareChecks ■ Pre(Provision)	
PrepareInstallSoftware ■ Pre(Provision)	

クラウド WIM イメージ ワークフロー

状態（トピック）	イベント（トピック）
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
InstallOS ■ Pre(Provision)	■ onInstallOSComplete(Provision) ■ OnInstallOSTimeout(Provision)
Reboot ■ Pre(Provision)	■ OnRebootComplete(Provision) ■ OnRebootTimeout(Provision)
SetupOS ■ Pre(Provision)	■ OnSetupOSComplete(Provision) ■ OnSetupOSTimeout(Provision)

外部プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)

状態（トピック）	イベント（トピック）
EpiRegister ■ Pre(Provision)	■ OnEpiRegisterComplete(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)

Linux キックスタート ワークフロー

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
CustomizeOS ■ Pre(Provision)	■ OnCustomizeOSComplete(Provision) ■ OnCustomizeOSTimeout(Provision)
EjectingCD ■ Pre(Provision)	■ OnEjectingCDComplete(Provision) ■ OnEjectingCDTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
InstallingOS ■ Pre(Provision)	■ OnInstallingOSComplete(Provision) ■ OnInstallingOSTimeout(Provision)

物理プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
FailedProvisioning ■ Pre(Provision)	
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeProvisioningTimeout(Provision)
InitializeProvisioning ■ Pre(Provision)	■ OnInitializeProvisioningTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnTimeout(Provision)
InstallOS ■ Pre(Provision)	■ OnInstallOSComplete(Provision) ■ OnInstallOSTimeout(Provision)

状態（トピック）	イベント（トピック）
Reboot ■ Pre(Provision)	■ OnRebootComplete(Provision) ■ OnRebootTimeout(Provision)
SetupOS ■ Pre(Provision)	■ OnSetupOSComplete(Provision) ■ OnSetupOSTimeout(Provision)

物理 PXE プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
CheckHardwareType ■ Pre(Provision)	
CleanPxe ■ Pre(Provision)	■ OnCleanPxeTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeProvisioningTimeout(Provision)
InitializeProvisioning ■ Pre(Provision)	■ OnInitializeProvisioningTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnTimeout(Provision)
InstallOS ■ Pre(Provision)	■ OnInstallOSComplete(Provision) ■ OnInstallOSTimeout(Provision)
Reboot ■ Pre(Provision)	■ OnRebootComplete(Provision) ■ OnRebootTimeout(Provision)
SetupOS ■ Pre(Provision)	■ OnSetupOSComplete(Provision) ■ OnSetupOSTimeout(Provision)
SetupPxe ■ Pre(Provision)	■ OnSetupPxeTimeout(Provision)

物理 SCCM プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
CheckHardwareType ■ Pre(Provision)	
Complete ■ Pre(Provision)	■ OnCompleteProvisioningComplete(Provision) ■ OnCompleteProvisioningTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	■ OnFailedProvisioningTimeout(Provision)
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeProvisioningTimeout(Provision)
InitializeProvisioning ■ Pre(Provision)	■ OnInitializeProvisioningTimeout(Provision)

状態（トピック）	イベント（トピック）
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnTimeout(Provision)
SccmRegistration ■ Pre(Provision)	■ OnSccmRegistrationTimeout(Provision)

物理 SCCM PXE プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
CheckHardwareType ■ Pre(Provision)	
CleanPxe ■ Pre(Provision)	■ OnCleanPxeTimeout(Provision)
Complete ■ Pre(Provision)	■ OnCompleteProvisioningComplete(Provision) ■ OnCompleteProvisioningTimeout(Provision)
Disposing ■ Pre(Provision)	
FailedProvisioning ■ Pre(Provision)	■ OnFailedProvisioningTimeout(Provision)
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeProvisioningTimeout(Provision)
InitializeProvisioning ■ Pre(Provision)	■ OnInitializeProvisioningTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnTimeout(Provision)
SccmRegistration ■ Pre(Provision)	■ OnSccmRegistrationTimeout(Provision)
SetupPxe ■ Pre(Provision)	■ OnSetupPxeTimeout(Provision)

vApp クローン ワークフロー

状態（トピック）	イベント [トリガ文字列]（トピック）
Global	■ OnFailProvisioning(Provision) ■ OnMasterProvisioned(Provision)
BuildComplete ■ Pre(Provision)	
CloneMachine ■ Pre(Provision)	■ OnCloneMachineComplete(Provision) ■ OnCloneMachineTimeout(Provision)
CustomizeMachine ■ Pre(Provision)	■ OnCustomizeMachineComplete(Provision) ■ OnCustomizeMachineTimeout(Provision)

状態（トピック）	イベント [トリガ文字列]（トピック）
CustomizeOS	■ OnCustomizeOS(Provision) ■ OnCustomizeOSComplete(Provision) ■ OnCustomizeOSTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
FinalizeProvisioning ■ Pre(Provision)	■ OnFinalizeComplete(Provision) ■ OnFinalizeTimeout(Provision)
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
WaitingForMaster ■ Pre(Provision)	■ OnWaitingForMasterTimeout(Provision)

仮想 SCCM プロビジョニング ワークフロー

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildComplete ■ Pre(Provision)	
CreatingMachine ■ Pre(Provision)	■ CreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
Disposing ■ Pre(Provision)	
EjectingCD ■ Pre(Provision)	■ OnEjectingCDComplete(Provision) ■ OnEjectingCDTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnPowerOnTimeout(Provision)
InstallingOS ■ Pre(Provision)	■ OnInstallingOSComplete(Provision) ■ OnInstallingOSTimeout(Provision)
SccmRegistration ■ Pre(Provision)	■ OnSccmRegistrationTimeout(Provision)

WIM イメージ ワークフロー

状態（トピック）	イベント（トピック）
AddingDisks ■ Pre(Provision)	■ OnAddingDisksComplete(Provision) ■ OnAddingDisksTimeout(Provision)
BuildComplete ■ Pre(Provision)	

状態（トピック）	イベント（トピック）
CreatingMachine ■ Pre(Provision)	■ OnCreatingMachineComplete(Provision) ■ OnCreatingMachineTimeout(Provision)
EjectingCD ■ Pre(Provision)	■ OnEjectingCDComplete(Provision) ■ OnEjectingCDTimeout(Provision)
FailedProvisioning ■ Pre(Provision)	
InitialPowerOn ■ Pre(Provision)	■ OnInitialPowerOnComplete(Provision) ■ OnInitialPowerOnTimeout(Provision)
InstallOS ■ Pre(Provision)	■ onInstallOSComplete(Provision) ■ OnInstallOSTimeout(Provision)
Reboot ■ Pre(Provision)	■ OnRebootComplete(Provision) ■ OnRebootTimeout(Provision)
SetupOS ■ Pre(Provision)	■ OnSetupOSComplete(Provision) ■ OnSetupOSTimeout(Provision)

状態およびイベントのタイムアウト値の設定

すべての状態およびイベントのデフォルト タイムアウト値は 30 分で、vRealize Automation のグローバル設定が適用されます。一部のワークフローでは、正常な実行でもこれより長い時間がかかる場合があります。環境内のさまざまなワークフローに合わせて、個々のワークフローまたは状態用にタイムアウトのオーバーライド値を追加できます。

デフォルトのタイムアウト値を変更するには、[インフラストラクチャ] - [管理] - [グローバル設定] を選択し、[拡張性ライフサイクル メッセージのタイムアウト] の値を編集します。グローバル設定を変更した場合は、マネージャ サービスを再起動する必要があります。

個々のタイムアウト値を設定するには、IaaS サーバ上にある `ManagerService.exe.config` ファイルの `appSetting` セクションに、ワークフローまたはイベントのプロパティを追加します。ファイルは通常、`%SystemDrive%\Program Files x86\VMware\vCAC\Server` ディレクトリにあります。このファイルを編集する前に必ずコピーをとっておいてください。個々の設定を変更した場合は、マネージャ サービスを再起動する必要があります。

キーの基本的な形式は、次の例のようになります。

- ワークフローの場合：`Extensibility.{workflow}.Timeout`
- イベントの場合：`Extensibility.{workflow}.{state}.EVENT.{event}.Timeout`
- 状態の場合：`Extensibility.{workflow}.{state}.(PRE/POST).Timeout`

`appSetting` セクションにキーを追加する際には、次の内容を例として使用します。タイムアウト値の形式は、D.HH:mm:ss.ms です。D は日で ms はミリ秒です。日とミリ秒はオプションです。時間、分、および秒は必須です。

- BasicVmWorkflow ワークフロー全体のタイムアウトを 30 分に設定するには、`<add key="Extensibility.BasicVmWorkflow.Timeout" value="00:30:00"/>` を追加します。

- VMPSMasterWorkflow32 の OnFinalizeMachine グローバル イベントのタイムアウトを 2 時間に設定するには、<add key="Extensibility.VMPSMasterWorkflow32.VMPSMasterWorkflow32.EVENT.OnFinalizeMachine.Timeout" value="02:00:00"/> を追加します。
- VMPSMasterWorkflow32 の事前申請状態のタイムアウトを 2 日に設定するには、<add key="Extensibility.VMPSMasterWorkflow32.Requested.PRE.Timeout" value="2.00:00:00"/> を追加します。

状態およびイベントのエラー動作の設定

ワークフロー サブスクリプションのタイムアウトとエラー処理にはデフォルトの動作があります。環境内のマシンに適用する動作をカスタマイズすることができます。

IaaS は、イベント タイムアウトおよびエラー処理をイベント ブローカ サービスから処理します。

状態が移行するたびに、SendEBSMessage はイベント ブローカ サービスにイベントを送信し、応答を待機します。デフォルトでは、タイムアウトまたはエラーがイベント ブローカ サービスによって報告されると、それが記録されてからワークフローが再開されます。

マスター ワークフローで次の状態が発生している間にタイムアウトまたはエラーが発生すると、ワークフローは再開せず、強制的にエラー状態になります。

表 1-16. ワークフローが再開しない例外

エラーが発生する状態	エラー状態
PRE MachineProvisioned	UnprovisionMachine
PRE BuildingMachine	Disposing
PRE RegisterMachine	Finalized

タイムアウトまたはエラーの動作をカスタマイズするには、イベントをトリガするまたは状態の変更を強制する任意のイベントまたは状態に対するカスタム プロパティをマシンに追加することができます。次の例を使用して、カスタム プロパティを設定します。

- `Extensibility.Lifecycle.Error.Event.{Workflow}.{State}`。プロパティの値は、タイムアウトまたはエラーが発生した場合にワークフロー内でトリガされるイベントの名前です。
- `Extensibility.Lifecycle.Error.State.{Workflow}.{State}`。プロパティの値は、タイムアウトまたはエラーが発生した場合にワークフローが強制的に移行する状態の名前です。

シナリオ：仮想マシンのプロビジョニング後のスナップショットの作成

テナント管理者は、仮想マシンのプロビジョニング後、サービス カタログ ユーザーが新しいマシンを申請することなく、スナップショットを使用して仮想マシンを初期状態に戻せるように設定できます。

手順

1 シナリオ：プロビジョニング後スナップショット アクション用の vRealize Orchestrator ワークフローの作成

必要な入力パラメータを受け取る vRealize Orchestrator ワークフローを作成します。プロビジョニング後の目標を達成するようにワークフローを設計します。

2 シナリオ：プロビジョニング後スナップショットのワークフロー サブスクリプションを作成する

テナント管理者は、各仮想マシンの作成後にスナップショットを作成することができます。マシン プロビジョニング イベント トピックに基づいてワークフロー サブスクリプションを設定し、公開してアクティブにします。

シナリオ：プロビジョニング後スナップショット アクション用の vRealize Orchestrator ワークフローの作成

必要な入力パラメータを受け取る vRealize Orchestrator ワークフローを作成します。プロビジョニング後の目標を達成するようにワークフローを設計します。

vRealize Orchestrator フォルダとワークフローの作成の詳細については、「VMware vRealize Orchestrator を使用した開発」を参照してください。

前提条件

ワークフローを作成できる権限のある vRealize Automation 用に設定されたインスタンスである vRealize Orchestrator にログインします。

手順

1 ワークフロー ライブラリに、ワークフロー サブスクリプションのワークフロー用のフォルダを作成します。

2 新しいワークフローを作成します。

このシナリオでは、ワークフローに **`Automation Post-Provisioning Snapshot`** という名前を付けます。

3 次の入力パラメータを追加します。

名前	タイプ
payload	プロパティ

4 入力パラメータを受け取り、仮想マシン スナップショットを作成するスクリプト可能なタスクを追加します。

5 ワークフローを保存します。

次のステップ

Automation Post-Provisioning Snapshot ワークフローを実行するワークフロー サブスクリプションを作成します。シナリオ：プロビジョニング後スナップショットのワークフロー サブスクリプションを作成する

シナリオ：プロビジョニング後スナップショットのワークフロー サブスクリプションを作成する

テナント管理者は、各仮想マシンの作成後にスナップショットを作成することができます。マシン プロビジョニング イベント トピックに基づいてワークフロー サブスクリプションを設定し、公開してアクティブにします。

仮想マシンがプロビジョニングされており、検出されたイベント メッセージが有効化された状態である場合にスナップショット作成ワークフローを実行する、ワークフロー サブスクリプションを構成します。

前提条件

- テナント管理者として vRealize Automation にログインします。
- vCenter Server プラグインを vRealize Orchestrator エンドポイントとして設定します。『vRealize Automation の構成』を参照してください。

- vSphere 仮想マシンのブループリントがあることを確認します。
- 仮想マシンのスナップショットを作成する vRealize Orchestrator ワークフローがあることを確認します。vRealize Automation プラグインで提供される、スナップショット作成ワークフローは使用できません。このスナップショット ワークフローは、XaaS 統合に固有のものです。プロビジョニングおよびライフサイクル ワークフロー用の vRealize Orchestrator ワークフローの設定を参照してください。

手順

1 [管理] - [イベント] - [サブスクリプション] を選択します。

2 [新規] アイコン（  ）をクリックします。

3 [マシン プロビジョニング] を選択します。

4 [次へ] をクリックします。

5 [条件] タブでトリガ条件を構成します。

a [条件に基づいて実行] を選択します。

b [条件節] ドロップダウン メニューから [次のすべて] を選択します。

c 次の条件を構成します。

プロパティ	演算子	値
[データ] > [マシン] > [マシン タイプ]	等しい	[定数] > [仮想マシン]
[データ] > [ライフサイクルの状態] > [ライフサイクル状態名]	等しい	[定数] > [VMPSMasterWorkflow32.MachineActivated]
[データ] > [ライフサイクルの状態] > [状態の段階]	等しい	[定数] > [POST]

d [次へ] をクリックします。

6 [ワークフロー] タブの [Orchestrator] ツリーを参照し、[Automation Post-Provisioning Snapshot] ワークフローを選択します。

7 [次へ] をクリックします。

8 [詳細] タブの [名前] と [説明] に値を指定します。

このシナリオでは、名前を **`Post-Provisioning Virtual Machine Snapshot`** とし、説明を **`Create a snapshot when new virtual machine is provisioned and activated`** とします。

9 [終了] をクリックします。

10 [Post-Provisioning Virtual Machine Snapshot] の行を選択して [公開] をクリックします。

結果

ワークフロー サブスクリプションが有効になり、申請された仮想マシンがプロビジョニングおよび有効化されたことがイベント メッセージに示されると、スナップショット ワークフローがトリガされます。

次のステップ

ワークフロー サブスクリプションをテストするには、サービス カタログの仮想マシンを申請します。プロビジョニングが成功したことが申請に示されたら、スナップショットが作成されたことを確認します。

承認ワークフロー サブスクリプションの使用

事前承認および事後承認のワークフロー サブスクリプションを作成すると、外部システムでの処理のために承認申請を送信できます。承認または却下された応答は、vRealize Automation によって処理されます。

承認サービスの統合

vRealize Automation の外部システムで承認申請を処理するカスタムの vRealize Orchestrator ワークフローを実行する、事前承認または事後承認のワークフロー サブスクリプションを作成します。

承認ポリシーの承認レベルで、承認者として [イベント サブスクリプションを使用する] を選択できます。このレベルは、承認ポリシーに含まれる複数のレベルの 1 つにすることができます。[イベント サブスクリプションを使用する] 承認者を含む承認ポリシーが適用されたアイテムをサービス カタログ ユーザーが申請すると、承認サービスでは、次の結果が含まれるメッセージをイベント ブローカ サービスに送信します。

- 公開済みのワークフロー サブスクリプションがあり、基準が一致する場合、vRealize Orchestrator は承認または却下のワークフローを実行します。
- 公開済みのワークフロー サブスクリプションがあっても基準が一致しない場合は、ワークフロー サブスクリプションの公開を解除します。または、公開済みのサブスクリプションがない場合は、承認レベルは承認され、承認プロセスは次の承認レベルに移行します。

承認ワークフロー サブスクリプションが承認サービスからメッセージを受け取り、そのメッセージが承認サブスクリプション用に設定された基準と比較されます。基準に一致するメッセージが見つかった場合、選択した vRealize Orchestrator ワークフローの実行が開始されます。公開されたイベント データは、ワークフローに入力として渡され、ワークフローに指定された方法で処理されます。ワークフローの結果が vRealize Automation に返され、申請が処理されます。承認されると、次の承認レベルが評価されます。却下されると、申請は却下されます。承認サービスのデフォルトのタイムアウトである 24 時間以内に、承認サービスが応答を受信しない場合、申請は却下として処理されます。

承認イベント トピック用の vRealize Orchestrator ワークフローの設定

カスタム vRealize Orchestrator ワークフローを設定して、承認メッセージをサポートし、vRealize Automation が処理できる適切な形式の情報で応答できるようにする必要があります。

承認イベント トピック スキーマ

事前承認および事後承認のイベント メッセージ スキーマには、フィールドの名前と値、申請に含まれる情報、および申請元に関する情報が含まれます。

イベント データ ペイロードの構造を次に示します。

```
{
    fieldNames : Properties,            // Property names

    fieldValues : Properties,           // Property values
```

```
    // Information about the request
    requestInfo : {
        requestRef : STRING,            // Identifier for the source request
        itemName : STRING,              // Name of the requested item
        itemDescription : STRING,       // Description of the requested item
        reason : STRING,                // Justification provided by the user specifying why the
request is required
        description : STRING,           // Description entered by the user specifying the purpose of
the request
        approvalLevel:ExternalReference,// Approval level ID. This is a searchable field
        approvalLevelName : STRING,     // Approval level name
        createDate : DATE_TIME,         // Time the approval request is created
        requestedFor : STRING,          // Principal id of the user for whom the source request is
initiated
        subtenantId : STRING,           // Business group id
        requestedBy : STRING            // Principal id of the user who actually submits the request
    },

    // Information about the source of the request
    sourceInfo : {
        externalInstanceId : STRING,    // Identifier of the source object, as defined by the
intiator service
        serviceId : STRING,             // Identifier of the service which initiated the approval
        externalClassId : STRING        // Identifier of the class to which the source object belongs
    }
}
```

プロパティ名とプロパティ値には、承認ポリシー レベルで設定するカスタム プロパティまたはシステム プロパティを指定できます。これらのプロパティは、承認者が承認プロセス中に値を変更できるようにするために承認ポリシー内で設定されます。たとえば、CPU が含まれる場合、承認者は承認申請フォームで CPU 数を減らすことができます。

応答イベント データ ペイロードは、ワークフローによって vRealize Automation に返される情報です。応答ペイロードのコンテンツによって、申請が承認されるか却下されるかが決まります。

```
{
    approved : BOOLEAN,

    // Property values
    fieldValues : Properties
}
```

応答イベント ペイロードに含まれる approved パラメータは、true（承認された申請）または false（拒否された申請）のいずれかです。プロパティの値は、vRealize Orchestrator ワークフローによる変更後に vRealize Automation に返され、承認プロセスに含められたカスタム プロパティまたはシステム プロパティです。

ベスト プラクティスとして、vRealize Orchestrator ワークフローを `businessJustification` の出力パラメータを使用して設定するようにします。このパラメータは、外部システムの承認者によって提供されたコメントを vRealize Automation 承認プロセスに渡すために使用できます。これらのコメントは、承認または却下のためのものである可能性があります。

承認スキーマに基づいた vRealize Orchestrator ワークフローの作成

作成するカスタム承認ワークフローには、タイプが Properties として設定され、任意のわかりやすい名前が付いた入力パラメータが必要です。ワークフロー サブスクリプションの実行がトリガされると、このパラメータに承認イベント データ ペイロードが入力されます。

vRealize Automation に送り返されるワークフローの出力パラメータは、`approved : Boolean` および `fieldValues : Properties` です。返される `approved : Boolean` パラメータによって、承認レベルが承認されるか却下されるかが決まります。`fieldValues : Properties` パラメータには、外部システムで変更された値が含まれます。

シナリオ：承認のための外部システムへのソフトウェア申請の送信

テナント管理者は、サービス カタログ ユーザーがソフトウェアを含むマシンを申請した場合に、vRealize Automation の外部のユーザーがソフトウェア コンポーネントを承認するように設定できます。すべてのソフトウェアのプロビジョニングに承認が必要な承認ポリシーと、定義された条件に一致する事前承認メッセージを受け取ったときに実行されるワークフロー サブスクリプションを設定します。

手順

1 シナリオ：承認ワークフロー サブスクリプション用の vRealize Orchestrator ワークフローの作成

vRealize Automation から必要な承認入力パラメータを受け取り、必要な出力パラメータを返して承認プロセスを完了させる vRealize Orchestrator ワークフローを作成します。

2 シナリオ：外部承認用の承認ポリシーの作成

テナント管理者は、承認サービスによって公開されるイベント メッセージを生成する承認ポリシーを作成します。イベント メッセージと一致する条件でワークフロー サブスクリプションを構成した場合は、選択したワークフローを vRealize Orchestrator が実行します。

3 シナリオ：事前承認ワークフロー サブスクリプションを作成する

テナント管理者は、サービス カタログ申請が、設定済みの条件に一致する承認要求を生成する場合は、vRealize Orchestrator ワークフローを実行している事前承認ワークフロー サブスクリプションを作成します。

シナリオ：承認ワークフロー サブスクリプション用の vRealize Orchestrator ワークフローの作成

vRealize Automation から必要な承認入力パラメータを受け取り、必要な出力パラメータを返して承認プロセスを完了させる vRealize Orchestrator ワークフローを作成します。

承認の目標を達成するようにワークフローを設計する必要があります。vRealize Orchestrator フォルダとワークフローの作成の詳細については、「VMware vRealize Orchestrator を使用した開発」を参照してください。

前提条件

ワークフローを作成できる権限のある vRealize Automation 用に設定されたインスタンスである vRealize Orchestrator にログインします。

手順

1 ワークフロー ライブラリに、ワークフロー サブスクリプションのワークフロー用のフォルダを作成します。

2 新しいワークフローを作成します。

このシナリオでは、ワークフローに **Automation Approval Request** という名前を付けます。

a 次の入力パラメータを追加します。

名前	タイプ
input	プロパティ

b 次の出力パラメータを追加します。

名前	タイプ
approved	真偽値
fieldValues	プロパティ

3 入力および出力パラメータを処理するスクリプト可能なタスクを作成します。

4 ワークフローを保存します。

次のステップ

承認者としてワークフロー サブスクリプションを使用する承認ポリシーを作成します。シナリオ：外部承認用の承認ポリシーの作成

シナリオ：外部承認用の承認ポリシーの作成

テナント管理者は、承認サービスによって公開されるイベント メッセージを生成する承認ポリシーを作成します。イベント メッセージと一致する条件でワークフロー サブスクリプションを構成した場合は、選択したワークフローを vRealize Orchestrator が実行します。

前提条件

- テナント管理者または承認管理者として vRealize Automation にログインします。

手順

1 [管理] - [承認ポリシー] を選択します。

2 ソフトウェア コンポーネント用の承認ポリシーを作成します。

a [新規] アイコン（  ）をクリックします。

b [承認ポリシーのタイプを選択] を選択します。

c リストで、[サービス カタログ - カタログ アイテム申請 - ソフトウェア コンポーネント] を選択します。

d [OK] をクリックします。

e 次のオプションを構成します。

オプション	構成
名前	**Software external approval** と入力します。
説明	**Approval request sent to external approval system** と入力します。
ステータス	[Active] を選択します。

3 [事前承認] タブで [追加] アイコン（  ）をクリックします。

4 トリガ基準と承認アクションを使用して [レベル情報] を構成します。

a [名前] テキスト ボックスに、**External level for software** と入力します。

b [説明] テキスト ボックスに、
Software approval request sent to external approval system と入力します。

c [常に必要] を選択します。

d [イベント サブスクリプションを使用する] を選択します。

5 [OK] をクリックします。

次のステップ

- 構成された承認レベルに基づいてイベント メッセージを受け取る事前承認ワークフロー サブスクリプションを作成します。シナリオ：事前承認ワークフロー サブスクリプションを作成するを参照してください。
- 資格内のソフトウェア コンポーネントに承認ポリシーを適用します。『vRealize Automation の構成』を参照してください。

シナリオ：事前承認ワークフロー サブスクリプションを作成する

テナント管理者は、サービス カタログ申請が、設定済みの条件に一致する承認要求を生成する場合は、vRealize Orchestrator ワークフローを実行している事前承認ワークフロー サブスクリプションを作成します。

前提条件

- テナント管理者として vRealize Automation にログインします。
- 「External level for software」という名前の承認ポリシー レベルを設定します。シナリオ：外部承認用の承認ポリシーの作成を参照してください。
- 外部システムに申請を送信するカスタム vRealize Orchestrator ワークフローを作成します。このシナリオでは、自動承認申請ワークフローを使用します。

手順

1 [管理] - [イベント] - [サブスクリプション] を選択します。

2 [新規] アイコン（  ）をクリックします。

3 [事前承認] をクリックします。

4 [次へ] をクリックします。

5 [条件] タブでトリガ条件を設定します。

a [条件に基づいて実行] を選択します。

b [条件節] ドロップダウン メニューで、次の条件を設定します。

プロパティ	演算子	値
データ > 申請に関する情報 > 承認レベル名	等しい	External level for software

c [次へ] をクリックします。

6 [ワークフロー] タブで、Orchestrator ツリーを参照し、[自動承認申請] ワークフローを選択します。

7 [次へ] をクリックします。

8 [詳細] タブで、名前と説明を入力します。

このシナリオでは、名前に **Software External**、説明に [Sends approval request to external system] と入力します。

9 [タイムアウト（分）] テキスト ボックスに「120」と入力します。

ワークフロー サブスクリプションのタイムアウトに指定する時間は、ターゲット システムに依存します。vRealize Automation が指定した時間（分）以内にターゲット システムからの応答を処理しない場合、リクエストは自動的に却下されます。

値を指定しない場合、デフォルトのタイムアウトは 24 時間です。

10 [終了] をクリックします。

11 [Software External] の行を選択し、[公開] をクリックします。

結果

これで、Software External Pre-Approval イベント サブスクリプションは事前承認イベント メッセージを受信します。

次のステップ

- 承認ポリシーをアクティブな資格のソフトウェア コンポーネントに適用した場合は、サービス カタログのアイテムを申請し、承認ポリシーとワークフロー サブスクリプションが設計のとおりに動作することを確認します。

ワークフロー サブスクリプションのトラブルシューティング

ワークフロー サブスクリプションのトラブルシューティングには、いくつかの共通した問題があります。各種のログの確認が必要になる場合もあります。

- 開始されない vRealize Orchestrator ワークフローのトラブルシューティング

  イベント メッセージの受信時にカスタム ワークフローを実行するようにワークフロー サブスクリプションを設定しても、ワークフローが実行されません。

- 時間がかかりすぎるプロビジョニング申請のトラブルシューティング

  IaaS マシンのプロビジョニングに 1 台あたり 10 時間以上かかります。

- 承認申請に対して実行されない vRealize Orchestrator ワークフローのトラブルシューティング

  vRealize Orchestrator ワークフローを実行するための事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、定義された条件に一致するマシンがサービス カタログで申請されても、ワークフローが実行されません。

- 承認申請が却下された場合のトラブルシューティング

  特定の vRealize Orchestrator ワークフローを実行する事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、承認されているにも関わらず申請が却下されます。

- 却下された承認申請のトラブルシューティング

  指定した vRealize Orchestrator ワークフローを実行する事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、申請が予期せずに却下されました。

開始されない vRealize Orchestrator ワークフローのトラブルシューティング

イベント メッセージの受信時にカスタム ワークフローを実行するようにワークフロー サブスクリプションを設定しても、ワークフローが実行されません。

解決方法

1. ワークフロー サブスクリプションが公開されていることを確認します。
2. ワークフロー サブスクリプションの条件が正しく設定されていることを確認します。
3. vRealize Orchestrator サーバに、指定されたワークフローがあることを確認します。

時間がかかりすぎるプロビジョニング申請のトラブルシューティング

IaaS マシンのプロビジョニングに 1 台あたり 10 時間以上かかります。

原因

プロビジョニング状態でワークフロー サブスクリプションをトリガするように設定すると、環境内で 2 つの IaaS マネージャ サービス インスタンスが実行されることがあります。

解決方法

- IaaS マネージャ サービスのインスタンスが 1 つだけアクティブであることを確認します。複数のインスタンスをアクティブにしている場合、ログには次のようなエラーも記録されます。

```
[EventBrokerService] Failed resuming workflow b6e9276a-f20f-40f1-99ad-6d9524560cc2 on queue
3679fa71-ac2a-42d5-8626-f98ea096f0d3. System.Workflow.Runtime.QueueException: Event Queue
operation failed with MessageQueueErrorCode QueueNotFound for queue '3679fa71-ac2a-42d5-8626-
f98ea096f0d3'. at System.Workflow.Runtime.WorkflowQueuingService.EnqueueEvent(IComparable
queueName, Object item) at System.Workflow.Runtime.WorkflowExecutor.EnqueueItem(IComparable
queueName, Object item, IPendingWork pendingWork, Object workItem) at
System.Workflow.Runtime.WorkflowInstance.EnqueueItem(IComparable queueName, Object item,
IPendingWork pendingWork, Object workItem) at
DynamicOps.VMPS.Service.Workflow.Services.EventBrokerService.OnMessage(EventObject obj)
[UTC:2015-11-14 07:14:25 Local:2015-11-13 23:14:25] [Error]: Thread-Id="15" - context="HKBsp6Tt"
token="JeuTG7ru" [EventBrokerClient] Invoking subscription callback failed: Event Queue operation
failed with MessageQueueErrorCode QueueNotFound for queue '3679fa71-ac2a-42d5-8626-f98ea096f0d3'.
```

承認申請に対して実行されない vRealize Orchestrator ワークフローのトラブルシューティング

vRealize Orchestrator ワークフローを実行するための事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、定義された条件に一致するマシンがサービス カタログで申請されても、ワークフローが実行されません。

原因

承認用のワークフロー サブスクリプションを正常に実行するには、すべてのコンポーネントが正しく設定されていることを確認する必要があります。

解決方法

1. 承認ポリシーがアクティブで、ポリシーの承認レベルに [イベント サブスクリプションを使用する] を選択したことを確認します。
2. 付与された資格に承認ポリシーが正しく適用されていることを確認します。
3. ワークフロー サブスクリプションが正しく設定および公開されていることを確認します。
4. 承認に関連するメッセージのイベント ログを参照します。

承認申請が却下された場合のトラブルシューティング

特定の vRealize Orchestrator ワークフローを実行する事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、承認されているにも関わらず申請が却下されます。

解決方法

1. vRealize Orchestrator でワークフローを確認します。
   a. 管理者権限を使用して vRealize Orchestrator にログインします。
   b. ワークフローがエラーなく実行されたことを確認します。
   c. `approval` および `fieldValues` パラメータに、期待される値が返されたことを確認します。
2. vRealize Automation で申請を確認します。
   a. 却下された項目を申請したユーザーとして vRealize Automation にログインします。
   b. [申請] タブをクリックします。
   c. 却下された申請を開きます。
   d. [承認ステータス] をクリックし、[申請理由] 列で詳細を確認します。

      エラーが発生した場合は、エラーに関する情報が [申請理由] のデータとして表示されます。

却下された承認申請のトラブルシューティング

指定した vRealize Orchestrator ワークフローを実行する事前承認または事後承認のワークフロー サブスクリプションを設定しましたが、申請が予期せずに却下されました。

問題

この外部承認レベル以前のすべての承認レベルは承認されており、このレベルも承認されているはずですが、却下として処理されました。

原因

考えられる原因の 1 つとして、vRealize Orchestrator がワークフローを実行しようとしたときに内部エラーが発生した可能性があります。たとえば、ワークフローが見つからなかったり、vRealize Orchestrator サーバが実行されていないことが考えられます。

解決方法

1 [管理] - [イベント] - [イベント ログ] の順に選択します。

2 承認に関連するメッセージのログを参照します。

vRealize Automation Designer の使用によるマシン ライフサイクルの拡張

vRealize Automation Designer を使用して状態変更ワークフロー スタブを直接編集し、必要に応じてカスタム vRealize Orchestrator ワークフローに呼び出すことで、IaaS マシン ライフサイクルの事前に設定されたステージにカスタム ロジックを挿入できます。

注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。

vRealize Automation Designer チェックリストを使用してマシンのライフサイクルを延長する

「vRealize Automation Designer チェックリストを使用してマシンのライフサイクルを延長する」では、IaaS マシンのライフサイクルをカスタマイズするために vRealize Automation Designer をインストールおよび構成するのに必要な手順についての概要を提供します。

表 1-17. vRealize Automation Designer チェックリストを使用してマシンのライフサイクルを延長する

タスク	詳細
❑ vRealize Automation Designer をダウンロードしてインストールします。	vRealize Automation Designer のインストール
❑ vRealize Orchestrator インスタンスに vRealize Automation エンドポイントを作成します。	vRealize Orchestrator エンドポイントの作成
❑ vRealize Orchestrator エンドポイントをマシンのブループリントに関連付けます。	vRealize Orchestrator エンドポイントとブループリントの関連付け

表 1-17. vRealize Automation Designer チェックリストを使用してマシンのライフサイクルを延長する（続き）

タスク	詳細
❑ vRealize Automation Designer アクティビティを使用して、IaaS ワークフロー スタブをカスタマイズします。	IaaS ワークフローのカスタマイズ
注： ワークフロー スタブは、イベント ブローカ ワークフロー サブスクリプションに置き換えられます。ワークフロー スタブは現在ではまだ提供されサポートされており、使用することも可能ですが、今後の vRealize Automation のバージョンでは削除される予定です。将来的な製品の互換性を確保するためには、ワークフロー サブスクリプションを使用して状態変更に基づくカスタム ワークフローを実行する必要があります。 vRealize Automation を拡張するためのワークフロー サブスクリプションの設定を参照してください。	
オプションで、vRealize Orchestrator ワークフロー アクティビティを使用してカスタムの vRealize Orchestrator ワークフローを呼び出すことができます。	
❑ カスタム状態変更ワークフローを作成した後、テナント管理者またはビジネス グループ マネージャは、カスタム プロパティを追加することによって特定のブループリントに対するワークフローを有効にする必要があります。	状態変更ワークフローを呼び出すためのブループリントの設定

vRealize Automation Designer のインストールと構成

Windows マシンに vRealize Automation Designer をインストールして、リモート Model Manager インスタンスと通信するように構成できます。vRealize Orchestrator ワークフローの呼び出しに IaaS ワークフローを使用する場合には、IaaS の vRealize Orchestrator インスタンスも構成する必要があります。

vRealize Automation Designer のインストール

Windows マシンに vRealize Automation Designer をインストールして、リモート Model Manager インスタンスと通信するように構成できます。

vRealize Automation Designer の前提条件

vRealize Automation Designer は通常、サーバではなく開発マシンにインストールされます。

サポートされているオペレーティング システム

vRealize Automation Designer でサポートされているオペレーティング システムは、VMwarevRealize Automation のドキュメント ページの vRealize Automation のサポート マトリックスに記載されています。

システム構成要件

この情報が潜在的に更新されているかどうかを確認するには、使用している vRealize Automation のバージョンの vRealize Automation のサポート マトリックスを参照してください。

- .NET Framework 4.5 をインストールする必要があります。
- vRealize Automation Designer ホストには、IaaS Web サイト コンポーネント（具体的には、Model Manager Web コンポーネント）に対するネットワーク アクセスが必要です。

- Model Manager がリモートでインストールされている場合には、Model Manager Web コンポーネントに使用されている証明書が vRealize Automation Designer ホストで信頼されている必要があります。

vRealize Automation Designer インストーラのダウンロード

vRealize Automation Designer インストーラは、vRealize Automation アプライアンスからダウンロードできます。

前提条件

- ローカル管理者として Windows マシンにログインします。
- Internet Explorer を使用している場合は、セキュリティ強化の設定が有効になっていないことを確認します。`res://iesetup.dll/SoftAdmin.htm` を参照してください。

手順

1. ブラウザを開きます。
2. https://*vra-va-hostname.domain.name*:5480/installer/ のホスト名を使用して、Windows インストーラのダウンロード ページに移動します。
3. [vRealize Automation Designer] をクリックします。
4. メッセージが表示されたら、インストーラを保存します。

次のステップ

vRealize Automation Designer のインストール。

vRealize Automation Designer のインストール

vRealize Automation Designer インストーラは、Windows インストール ウィザードとしてパッケージされています。

前提条件

vRealize Automation Designer インストーラのダウンロード。

手順

1. インストーラをダウンロードしたディレクトリに移動します。
2. `DesignCenter Setup.exe` を右クリックし、[管理者として実行] を選択します。
3. [ようこそ] ページで、[次へ] をクリックします。
4. 使用許諾契約書を読み、[使用許諾契約書に同意します] を選択し、[次へ] をクリックします。
5. [カスタム セットアップ] ページで、[次へ] をクリックします。
6. Model Manager Web インスタンスの完全修飾ドメイン名およびポートを *Hostname:port* 形式で指定します。

   デフォルト ポートは 443 です。
7. Model Manager サービスのユーザー認証情報を指定します。

8 [次へ] をクリックします。

インストーラは、Model Manager へのアクセスを試みることで Model Manager ホストと認証情報の組み合わせを検証します。エラーが返された場合は、続行する前に Model Manager ホストと認証情報の正しい組み合わせを指定する必要があります。

9 [インストール] をクリックします。

10 [終了] をクリックします。

次のステップ

インストール ディレクトリに移動して Windows の [スタート] メニューから vRealize Automation Designer を起動することができます。

vRealize Orchestrator エンドポイントの設定

vRealize Automation ワークフローを使用して vRealize Orchestrator ワークフローを呼び出す場合は、vRealize Orchestrator インスタンスまたはサーバをエンドポイントとして設定する必要があります。

vRealize Orchestrator エンドポイントをマシン ブループリントに関連付けて、そのブループリントからプロビジョニングされたマシンの vRealize Orchestrator ワークフローのすべてが、そのエンドポイントを使用して実行されるようにすることができます。

vRealize Automation には、デフォルトで、組み込みの vRealize Orchestrator インスタンスが含まれています。 本番環境またはテスト環境で vRealize Automation ワークフローを実行するときや、事前検証（POC）を作成するときに、vRealize Orchestrator エンドポイントとしてこの組み込みのインスタンスを使用することをお勧めします。

プラグインを外部 vRealize Orchestrator サーバにインストールすることもできますが、この方法は本番環境では推奨されません。

vRealize Orchestrator の統合の前提条件

vRealize Automation ワークフローを使用して、`VC:VirtualMachine` のタイプの入力または出力パラメータを持つ vRealize Orchestrator ワークフローを実行する場合、vRealize Orchestrator と IaaS 間で仮想マシンのタイプを変換するための vRealize Orchestrator ワークフローがあることを確認します。

vRealize Orchestrator 5.5 以降では、必要なワークフローはデフォルトで vCenter Server プラグインの一部に含まれています。

vRealize Orchestrator 5.1 を使用している場合は、vRealize Orchestrator の vRealize Automation 統合パッケージをインストールします。パッケージ `com.vmware.library.vcenter.vcac-integration.package` を vRealize Orchestrator コミュニティ サイト (https://communities.vmware.com/t5/vRealize-Orchestrator-Documents/vCloud-Automation-Center-integration-package/ta-p/2777982) からダウンロードします。IaaS のエンドポイントとして設定した各 vRealize Orchestrator サーバにパッケージをインポートします。

パッケージを vRealize Orchestrator にインポートする方法の詳細については、vRealize Orchestrator ドキュメントを参照してください。

vRealize Orchestrator エンドポイントの作成

vRealize Orchestrator サーバに接続する vRealize Orchestrator エンドポイントを作成できます。

複数のエンドポイントを設定して、別々の vRealize Orchestrator サーバに接続できますが、各エンドポイントに優先度を設定する必要があります。

vRealize Orchestrator ワークフローを実行するとき、vRealize Automation は、最初に最も優先度の高い vRealize Orchestrator エンドポイントの使用を試みます。 そのエンドポイントにアクセスできない場合は、vRealize Orchestrator サーバがワークフローを実行できるようになるまで、次に優先度の高いエンドポイントの使用を試みます。

前提条件

- IaaS 管理者として vRealize Automation にログインします。

手順

1. [インフラストラクチャ] - [エンドポイント] - [エンドポイント] を選択します。
2. [新規] - [オーケストレーション] - [vRealize Orchestrator] の順に選択します。
3. 名前と説明（説明は任意）を入力します。
4. vRealize Orchestrator サーバの完全修飾名または IP アドレスおよび vRealize Orchestrator ポート番号を含む URL を入力します。

   転送プロトコルは HTTPS にする必要があります。 ポートが指定されない場合は、デフォルト ポート 443 が使用されます。

   vRealize Automation アプライアンスに組み込まれたデフォルトの vRealize Orchestrator インスタンスを使用するには、**https://*vrealize-automation-appliance-hostname*:443/vco** と入力します。
5. vRealize Orchestrator の認証情報を [ユーザー名] と [パスワード] テキスト ボックスに入力し、vRealize Orchestrator エンドポイントに接続します。

   使用する認証情報には、IaaS から呼び出す vRealize Orchestrator ワークフローに対する実行権限が必要です。

   vRealize Automation アプライアンスに組み込まれたデフォルトの vRealize Orchestrator インスタンスを使用する場合、ユーザー名は **administrator@vsphere.local**、パスワードは、SSO の構成時に指定した管理者パスワードになります。
6. [優先度] テキスト ボックスに 1 以上の整数を入力します。

   値が小さいほど、優先度が高くなります。
7. （オプション）[プロパティ] をクリックし、提供されたカスタム プロパティ、プロパティ グループ、またはエンドポイント用の独自のプロパティ定義を追加します。
8. [OK] をクリックします。

vRealize Orchestrator エンドポイントとブループリントの関連付け

特定の vRealize Orchestrator エンドポイントをブループリントと連携するように指定できます。

このブループリントからプロビジョニングされたマシンに対して IaaS が vRealize Orchestrator ワークフローを実行する場合は、常に、関連付けられているエンドポイントが使用されます。エンドポイントにアクセスできない場合、ワークフローは失敗します。

前提条件

インフラストラクチャ アーキテクトとして vRealize Automation にログインします。

手順

1 [設計] - [ブループリント] を選択します。

2 新しいブループリントを作成するか、既存のブループリントを編集します。

既存のブループリントを編集する場合、指定する vRealize Orchestrator エンドポイントは、更新後のブループリントから新たにプロビジョニングされるマシンにのみ適用されます。このブループリントからプロビジョニングされた既存のマシンでは、このプロパティが手動で追加されない限り、引き続き最も優先度が高いエンドポイントが使用されます。

3 [プロパティ] タブをクリックします。

a [新規プロパティ] をクリックします。

b [名前] テキスト ボックスに **VMware.VCenterOrchestrator.EndpointName** と入力します。

このプロパティ名は大文字と小文字が区別されます。

c vRealize Orchestrator エンドポイントの名前を [値] テキスト ボックスに入力します。

d [保存] アイコン（  ）をクリックします。

4 [OK] をクリックします。

vRealize Automation Designer を使用した IaaS ワークフローのカスタマイズ

VMware では、vRealize Automation Designer を使用してカスタマイズできるさまざまなワークフローを提供しています。これには、状態変更ワークフローとメニュー操作ワークフローが含まれます。

IaaS ワークフローは、.NET Framework 4 に含まれる Microsoft Windows Workflow Foundation 4 を使用して作成されます。Windows Workflow Foundation とワークフロー作成の詳細については、Microsoft のドキュメントを参照してください。vRealize Automation にも、vRealize Orchestrator ワークフローの実行および監視用にいくつかの vRealize Automation Designer アクティビティが提供されています。

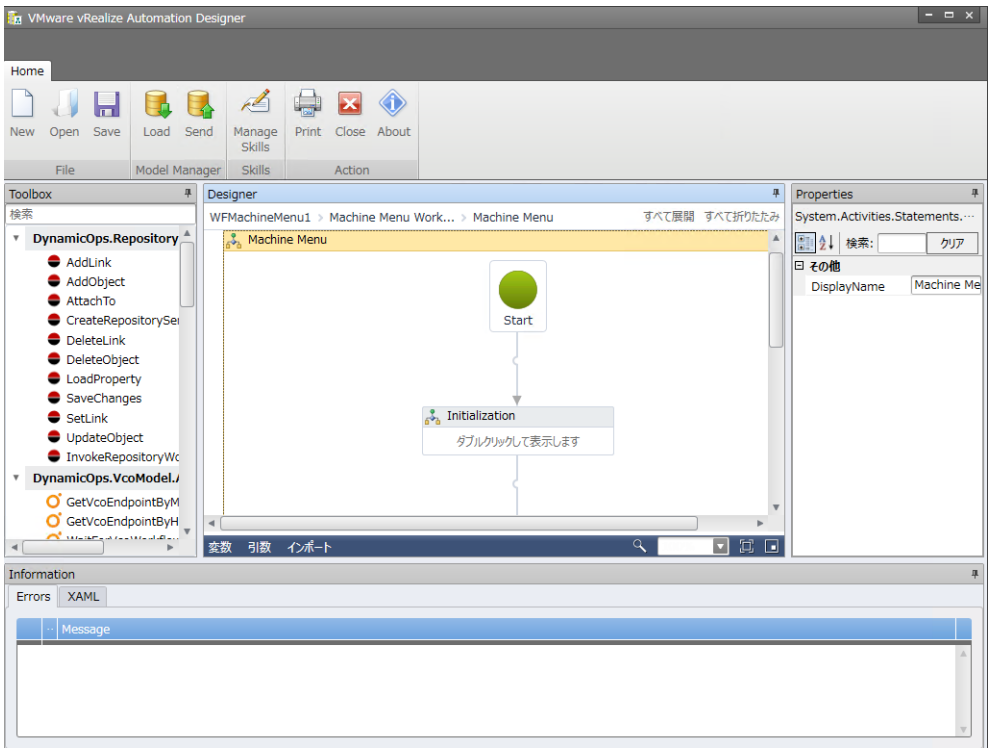
VMware が提供するカスタマイズ可能なワークフロー テンプレートでは、初期化、カスタム ロジックおよびファイナライズのそれぞれに向けたワークフロー構築のベスト プラクティスが手順付きで説明されています。ワークフロー全体がエラー処理のために TryCatch ブロックにラップされています。見逃された例外や再スローされた例外は、ワークフローを実行している Distributed Execution Manager によってログに記録されます。

カスタムの IaaS ワークフローを作成したら、ブループリントの作成者は特定のブループリントでワークフローを有効にする必要があります。

vRealize Automation Designer コンソール

vRealize Automation Designer コンソールは、IaaS ワークフローをカスタマイズするための視覚的なワークフロー エディタを提供します。

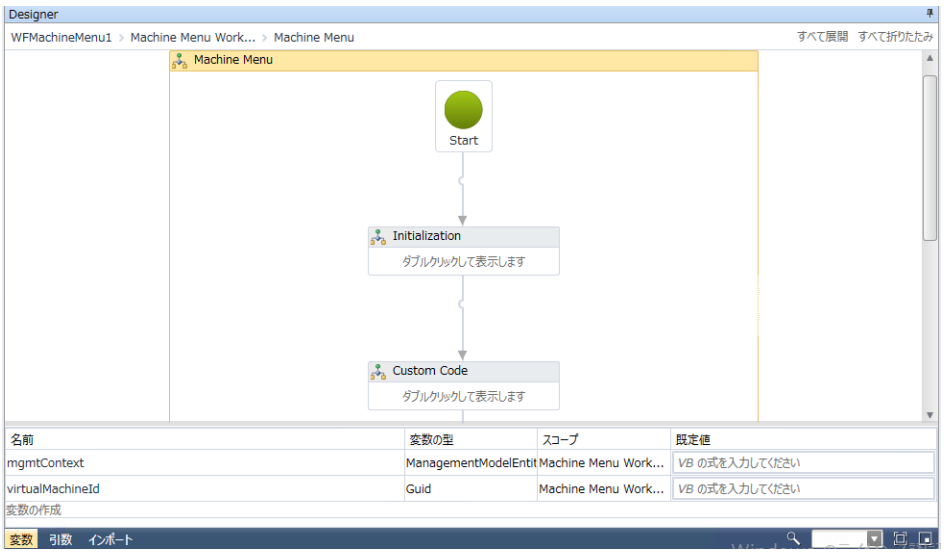
vRealize Automation Designer コンソールを起動するために、vRealize Automation Designer ホスト（通常は開発マシン）でのローカル管理者権限が必要です。



左側の [ツールボックス] ペインでは、vRealize Automation ワークフロー アクティビティ ライブラリにアクセスできます。ツールボックスから [Designer] ペインにアクティビティをドラッグして、ワークフローに追加できます。[プロパティ] ペインには、[Designer] ペインで現在選択されているアクティビティの設定可能なプロパティが表示されます。このインターフェイスは、Visual Studio のワークフロー デザイナと非常によく似ています。

[Designer] ペインの下部にある [詳細] タブでは、選択したアクティビティの範囲内の変数または選択したアクティビティに対する引数を表示および編集できます。

注： 変数と引数は、両方とも Visual Basic の式として指定します。ただし、変数名の大文字と小文字は区別されず、引数名は区別されます。IaaS ワークフロー アクティビティにおける有効な引数の詳細については、 vRealize Automation ワークフロー アクティビティ リファレンスを参照してください。



[インポート] タブにはインポートされた名前空間が表示され、ここからエンティティ タイプを選択してワークフローに追加できます。

コンソールの下部にある折りたたみ可能な [情報] ペインでは、アクティビティを設定する際のエラーが表示され、ワークフローの XAML 表現にアクセスできます。

IaaS ワークフローのタイプ

vRealize Automation Designer を使用して 2 つのタイプのワークフローをカスタマイズできます。すなわち、状態変更ワークフローとメニュー操作ワークフローです。

- 状態変更ワークフローは、たとえば新しいマシンのプロビジョニング プロセスにおける特定のステージなど、メイン ワークフローが状態間で移行する場合に実行されます。
- メニュー操作ワークフローは、ユーザーがサービス カタログの [アクション] メニュー、または [インフラストラクチャ] タブの [マシン] メニューからオプションを選択するときに実行されます。

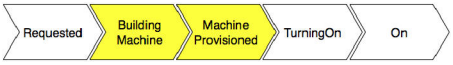
状態変更ワークフロー

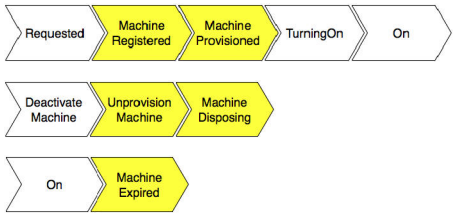
状態変更ワークフローを作成することで、IaaS メイン ワークフローが特定の状態になる前にワークフローを実行できます。

たとえば、カスタム ワークフローを作成して外部データベースと連携し、マシンのライフサイクルのさまざまなステージの情報を記録することができます。

- メイン ワークフローが MachineProvisioned 状態になる前に実行され、マシンの所有者や承認者などの情報を記録するカスタム ワークフローを作成します。
- マシンが MachineDisposing 状態になる前に実行され、マシンが破棄された時間と、最後のデータ収集時や最後のログイン時などのリソース使用率などのデータを記録するカスタム ワークフローを作成します。

次の図は、メイン ワークフローの主な状態を示しています。





vRealize Automation Designer は、これらの各状態のためにカスタマイズ可能なワークフローを提供します。

表 1-18. カスタマイズ可能な状態変更ワークフロー

メイン ワークフローの状態	カスタマイズ可能なワークフロー名
BuildingMachine	WFStubBuildingMachine
Disposing	WFStubMachineDisposing
Expired	WFStubMachineExpired
MachineProvisioned	WFStubMachineProvisioned
RegisterMachine	WFStubMachineRegistered
UnprovisionMachine	WFStubUnprovisionMachine

状態変更ワークフローの設定の概要

vRealize Automation Designer を使用して状態変更ワークフローをカスタマイズできます。その後、ブループリントの作成者が特定のブループリントに対してワークフローを有効にできます。

状態変更ワークフローを有効にする手順の概要は次のとおりです。

1 ワークフロー開発者は、vRealize Automation Designer を使用して状態変更ワークフローのテンプレートのいずれかをカスタマイズします。IaaS ワークフローのカスタマイズを参照してください。

任意の IaaS ワークフローから vRealize Orchestrator ワークフローを呼び出すことができます。詳細については、vRealize Orchestrator ワークフロー アクティビティの使用を参照してください。

2 テナント管理者またはビジネス グループ マネージャは、ブループリントからプロビジョニングされるマシン用にカスタム ワークフローを呼び出すよう、ブループリントを設定します。状態変更ワークフローを呼び出すためのブループリントの設定を参照してください。

メニュー操作ワークフロー

メニュー操作ワークフローは、ユーザーがサービス カタログの [アクション] メニュー、または [インフラストラクチャ] タブのマシン メニューからオプションを選択すると実行されます。

たとえば、ユーザーが [マシン] メニューから [サポート問題の提出] を選択することでマシンに関連するサポート チケットを作成できるカスタム ワークフローを作成することができます。

vRealize Automation Designer は、メニュー操作のワークフローをカスタマイズするためのテンプレートを提供します。

メニュー操作のワークフローは、ワークフローの定義のほか、操作構成ファイルにも依存します。このファイルは、表示テキスト、テキストにアクセスできるロール、および操作を利用できるマシン状態など、カスタム メニュー オプションの各項目を定義します。

注： XaaS アーキテクトは、XaaS を使用して任意のカタログ アイテムに対するカスタム アクションを定義することができます。vSphere または vCloud Director を使用してプロビジョニングされたもの以外の IaaS マシンのカスタム アクションを作成するには、vRealize Automation 6.1 以降が必要です。

メニュー操作ワークフローの設定の概要

vRealize Automation Designer と CloudUtil コマンドライン ユーティリティを使用して、メニュー操作ワークフローをカスタマイズできます。その後、ブループリントの作成者が特定のブループリントに対してワークフローを有効にできます。

メニュー操作ワークフローを有効にするための手順の概要は次のとおりです。

1 ワークフロー開発者は、vRealize Automation Designer を使用してメニュー操作ワークフローのテンプレートのいずれかをカスタマイズします。IaaS ワークフローのカスタマイズを参照してください。

  任意の IaaS ワークフローから vRealize Orchestrator ワークフローを呼び出すことができます。詳細については、vRealize Orchestrator ワークフロー アクティビティの使用を参照してください。

2 ワークフロー開発者は、Model Manager でメニュー操作を設定します。メニュー操作の設定を参照してください。

3 ワークフロー開発者は、サービス カタログを使用して新しいメニュー操作を登録します。サービス カタログを使用して新しいメニュー操作を登録するを参照してください。

4 テナント管理者またはビジネス グループ マネージャは、そのブループリントからプロビジョニングされるマシンのメニュー操作を有効にするためにブループリントを設定します。メニュー操作ワークフローを有効にするためのブループリントの設定を参照してください。

メニュー操作をサービス カタログで使用する場合は、ユーザーに対して資格を付与する必要があります。詳細については、『テナント管理』を参照してください。

IaaS ワークフローのカスタマイズ

vRealize Automation Designer を使用すると、カスタマイズ可能なワークフローを編集したり、Model Manager でワークフローを更新したりできます。

前提条件

vRealize Automation Designer を起動します。

手順

1 [ロード] をクリックします。

2 カスタマイズするワークフローを選択します。

オプション	説明
WFMachineMenu*N*	カスタマイズ可能なメニュー操作ワークフロー
WFStubBuildingMachine	マシンが BuildingMachine の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー
WFStubMachineDisposing	マシンが Disposing の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー
WFStubMachineExpired	マシンが Expired の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー
WFStubMachineProvisioned	マシンが MachineProvisioned の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー
WFStubMachineRegistered	マシンが RegisterMachine の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー
WFStubUnprovisionMachine	マシンが UnprovisionMachine の状態になる前に実行される、カスタマイズ可能な状態変更ワークフロー

3 [OK] をクリックします。

[Designer] ペインにワークフローが表示されます。

4 アクティビティをツールボックスから [Designer] ペインにドラッグし、その引数を設定して、ワークフローをカスタマイズします。

5 ワークフローの編集が終了したら、[送信] をクリックして、Model Manager でワークフローを更新します。

ワークフローが保存され、次にワークフローをロードしたときにリストに新しいバージョンとして表示されます。ワークフローの以前のバージョンにはいつでもアクセスできます。ワークフローの以前のリビジョンに戻すを参照してください。

vRealize Orchestrator ワークフロー アクティビティの使用

vRealize Automation Designer アクティビティを使用して、同期または非同期で vRealize Orchestrator ワークフローを呼び出すことができます。

vRealize Orchestrator エンドポイントは次のいずれかの方法で指定します。

- `VirtualMachineId` は、仮想マシン ID を表す変数の名前です。この ID を持つ仮想マシンが選択され、仮想マシンの `VMware.VCenterOrchestrator.EndpointName` カスタム プロパティから取得される値が vRealize Orchestrator エンドポイント名として使用されます。
- `GetVcoEndpointByManagementEndpoint` は指定された `ManagementEndpoint` オブジェクトのカスタム プロパティの値を返します。`CustomPropertyName` が指定されていない場合は、`VMware.VCenterOrchestrator.EndpointName` プロパティの値が使用されます。
- `GetVcoEndpointByHost` は指定されたホストのカスタム プロパティの値を返します。`CustomPropertyName` が指定されていない場合は、`VMware.VCenterOrchestrator.EndpointName` プロパティの値が使用されます。

同期実行

InvokeVcoWorkflow アクティビティは vRealize Orchestrator ワークフローを呼び出し、vRealize Orchestrator ワークフローが完了するまでその親 IaaS ワークフローのさらなる実行をブロックします。このアクティビティは vRealize Orchestrator ワークフローの出力パラメータを返します。

さらに、同期実行は次のプロパティをサポートします。

- WorkflowTimeout は秒単位のタイムアウト値です。vRealize Orchestrator ワークフローが指定された時間に完了しない場合は、応答が返されるまでワークフローをブロックするのではなく、例外が生成されます。値が定義されない場合、または値としてゼロが指定された場合、タイムアウトは有効になりません。VMware.VCenterOrchestrator.PollingInterval カスタム プロパティの値を指定してエンドポイントのポーリング時間が変更されない限り、ワークフローの状態がタイムアウト期間中 10 秒ごとに確認されます。

ワークフローの非同期実行

InvokeVcoWorkflowAsync アクティビティは、vRealize Orchestrator ワークフローを呼び出して、vRealize Orchestrator ワークフローの完了を待機することなく IaaS ワークフローのアクティビティを引き続き実行するアクティビティです。

このアクティビティは、ワークフローの監視に使用できる一意のワークフロー トークンを返すか、vRealize Orchestrator サーバに対する REST API 呼び出しが失敗した場合にはエラーを返します（サーバに到達できなかった場合など）。

このアクティビティは、以下の 2 つの追加アクティビティと併用できます。

- GetVcoWorkflowExecutionStatus を使用することで、vRealize Orchestrator ワークフローをそのステータスにポーリングできます。
- WaitForVcoWorkflowCompletion を使用することで、vRealize Orchestrator ワークフローが完了するかタイムアウトになるまで、IaaS ワークフローのさらなる実行をブロックできます。このアクティビティを使用して、非同期で実行した vRealize Orchestrator ワークフローの結果を取得することができます。

vRealize Orchestrator ワークフローの呼び出し

InvokeVcoWorkflow または InvokeVcoWorkflowAsync のいずれかのアクティビティを使用して、IaaS ワークフローから vRealize Orchestrator ワークフローを呼び出すことができます。

一部の vRealize Orchestrator ワークフローでは、ユーザーの操作が必要です。これらのワークフローの場合、vRealize Automation コンソールではなく vRealize Orchestrator クライアントにユーザー プロンプトが表示されるため、vRealize Automation のエンド ユーザーには、ワークフローが入力待機中であることが提示されません。

ワークフローがユーザー入力によって停止されないようにするには、ユーザー操作を必要とする vRealize Orchestrator ワークフローを IaaS ワークフローから呼び出さないようにします。

手順

1. vRealize Automation Designer でワークフローを開き、vRealize Orchestrator ワークフローを呼び出すコンテキストに移動します。
2. InvokeVcoWorkflow または InvokeVcoWorkflowAsync のアクティビティを [Designer] ペインにドラッグします。

3 実行する vCenter Orchestrator ワークフローを選択します。

a [全般] の下で、ワークフローの横にある省略記号をクリックします。

b [vCO ワークフローの参照] ダイアログ ボックスでワークフローを選択します。

c [OK] をクリックします。

[入力] と [出力] セクションに、選択したワークフローの入力パラメータと出力パラメータが表示されます。

4 [プロパティ] ペインで、次のターゲット パラメータのいずれかを指定します。

- `VirtualMachineId` は、仮想マシン ID を表す変数の名前です。この ID を持つ仮想マシンが選択され、仮想マシンの `VMware.VCenterOrchestrator.EndpointName` カスタム プロパティから取得される値が vRealize Orchestrator エンドポイント名として使用されます。
- `VcoEndpointName` は、ワークフローの実行に使用されるエンドポイント名です。このパラメータを指定すると、vRealize Orchestrator エンドポイントを選択するときに `VirtualMachineId` の値がオーバーライドされます。
- `WorkflowTimeout` は秒単位のタイムアウト値です。vRealize Orchestrator ワークフローが指定された時間に完了しない場合は、応答が返されるまでワークフローをブロックするのではなく、例外が生成されます。値が定義されない場合、または値としてゼロが指定された場合、タイムアウトは有効になりません。`VMware.VCenterOrchestrator.PollingInterval` カスタム プロパティの値を指定してエンドポイントのポーリング時間が変更されない限り、ワークフローの状態がタイムアウト期間中 10 秒ごとに確認されます。

5 vRealize Orchestrator ワークフローのパラメータを指定します。

- [Designer] ペインのアクティビティに値を入力します。
- [プロパティ] ペインの [入力パラメータ] または [出力パラメータ] の横にある省略記号をクリックして、[パラメータ] ダイアログ ボックスを開きます。このダイアログ ボックスでは、各パラメータの IaaS タイプが表示されます。パラメータ タイプが太字で表示されるパラメータは必須です。

任意のパラメータのテキスト ボックスをポイントすると、vRealize Orchestrator タイプを示すツールチップが表示されます。

`InvokeVcoWorkflowAsync` アクティビティを使用している場合は、vRealize Orchestrator ワークフローの出力パラメータとそれに対応するタイプが情報として表示されますが、このアクティビティでパラメータの式を指定することはできません。

次のステップ

非同期的に実行したワークフローの結果を取得するには、`WaitForVcoWorkflowCompletion` アクティビティを使用します。

vRealize Orchestrator ワークフローのステータスを取得する

`GetVcoWorkflowExecutionStatus` アクティビティを使用して、`InvokeVcoWorkflowAsync` アクティビティで呼び出された vRealize Orchestrator ワークフローのステータスを確認することができます。

前提条件

`InvokeVcoWorkflowAsync` アクティビティを使用した vRealize Orchestrator ワークフローの呼び出し。

手順

1. vRealize Automation Designer で、InvokeVcoWorkflowAsync アクティビティを使用したワークフローを開きます。
2. vRealize Orchestrator ワークフローのステータスを確認するコンテキストに移動します。
3. GetVcoWorkflowExecutionStatus アクティビティを [Designer] ペインにドラッグします。
4. [プロパティ] ペインで VirtualMachineId. の仮想マシン ID を表す変数の名前を指定します。

   カスタマイズ可能なワークフローには、初期化中に設定される virtualMachineId という名前の変数がデフォルトで含まれています。
5. DynamicOps.VcoModel.Common.VcoWorkflowExecutionToken というタイプの変数を作成します。
6. InvokeVcoWorkflowAsync アクティビティの executionToken 出力パラメータとして、トークン変数の名前を指定します。
7. GetVcoWorkflowExecutionStatus アクティビティの WorkflowExecutionToken プロパティとして同じ変数名を指定します。
8. 文字列型の変数を作成します。
9. GetVcoWorkflowExecutionStatus アクティビティの VcoWorkflowExecutionStatus プロパティとしての文字列変数の名前を指定します。

結果

ワークフローを実行すると、VcoWorkflowExecutionStatus 変数の値は、vRealize Orchestrator ワークフローのステータスに設定されます。

vRealize Orchestrator ワークフローの結果を取得する

非同期的に vRealize Orchestrator ワークフローを呼び出し、完了したワークフローの結果を後で取得する場合は、WaitForVcoWorkflowCompletion アクティビティを使用できます。

vRealize Orchestrator ワークフローが完了するか、タイムアウトに到達するまで、WaitForVcoWorkflowCompletion アクティビティは IaaS ワークフローをブロックします。アクティビティは、vRealize Orchestrator ワークフローが正常に完了すると結果を返し、失敗するとエラーを返し、タイムアウトになると null を返します。

前提条件

InvokeVcoWorkflowAsync アクティビティを使用した vRealize Orchestrator ワークフローの呼び出し。

手順

1. vRealize Automation Designer で、InvokeVcoWorkflowAsync アクティビティを使用したワークフローを開きます。
2. vRealize Orchestrator ワークフローの結果を取得するコンテキストに移動します。
3. WaitForVcoWorkflowCompletion アクティビティを [Designer] ペインにドラッグします。

4 [プロパティ] ペインで VirtualMachineId. の仮想マシン ID を表す変数の名前を指定します。

カスタマイズ可能なワークフローには、初期化中に設定される virtualMachineId という名前の変数がデフォルトで含まれています。

5 DynamicOps.VcoModel.Common.VcoWorkflowExecutionToken というタイプの変数を作成します。

6 DynamicOps.VcoModel.Common.VcoWorkflowExecutionToken というタイプの変数を作成します。

7 InvokeVcoWorkflowAsync アクティビティの executionToken 出力パラメータとして、トークン変数の名前を指定します。

8 WaitForVcoWorkflowCompletion アクティビティの WorkflowExecutionToken プロパティとして同じ変数名を指定します。

9 vRealize Orchestrator ワークフローの出力を取得します。

a DynamicOps.VcoModel.Common.VcoWorkflowExecutionResult というタイプの変数を作成します。

b WaitForVcoWorkflowCompletion アクティビティの WorkflowOutput プロパティとして結果変数の名前を指定します。

ワークフローを実行すると、変数が存在する場合、その値が vRealize Orchestrator ワークフローの結果に設定されます。

vRealize Orchestrator および IaaS オブジェクト タイプ

vRealize Automation Designer で InvokeVcoWorkflow または InvokeVcoWorkflowAsync アクティビティを使用する場合、アクティビティの入力および出力プロパティは選択する vRealize Orchestrator ワークフローのパラメータに基づいて自動的に適用されます。

基本的な vRealize Orchestrator オブジェクト タイプは、次の IaaS オブジェクト タイプに変換されます。

表 1-19. vRealize Orchestrator および IaaS オブジェクト タイプ

vRealize Orchestrator タイプ	IaaS タイプ
string	string
boolean	bool
number	decimal
SecureString	string
Text	string
Array/T	Array<T>
Properties	Dictionary<string,object>
Date	DateTime
VC:VirtualMachine	VirtualMachine

注： vRealize Orchestrator 5.1 を使用している場合、VC:VirtualMachine オブジェクト タイプから VirtualMachine に変換するためには、vRealize Automation 統合パッケージをインストールしておく必要があります。

その他のすべての vRealize Orchestrator タイプは、IaaS タイプ `VcoSdkObject` に変換されます。

状態変更ワークフローを呼び出すためのブループリントの設定

カスタムの状態変更ワークフローを作成したら、テナント管理者またはビジネス グループ マネージャは、カスタム プロパティを追加して特定のブループリント用にワークフローを有効にする必要があります。

状態変更ワークフローは、それぞれが特定のカスタム プロパティに関連付けられます。対応する状態変更ワークフローによってマシンが特定の状態になる際に、IaaS は、対応するカスタム プロパティがそのマシンにあるかどうかを確認します。目的のプロパティが見つかると、関連付けられたワークフローが実行されます。たとえば、マシンにカスタム プロパティ `ExternalWFStubs.MachineProvisioned` がある場合、`WFStubMachineProvisioned` ワークフローはマスター ワークフローが MachineProvisioned の状態になる前に実行されます。

カスタム プロパティは、複数のソースから 1 台のマシンに適用できますが、通常、状態変更ワークフローのプロパティはブループリントで指定されるため、すべてのマシンのワークフローを特定のブループリントからプロビジョニングすることが可能になります。

前提条件

テナント管理者またはビジネス グループ マネージャとして vRealize Automation にログインします。

手順

1 [設計] - [ブループリント] を選択します。

2 ブループリントの名前をポイントし、[編集] をクリックします。

3 [ブループリントのプロパティ] のアイコン () を選択します。

4 [プロパティ] タブをクリックします。

5 [カスタム プロパティ] - [新規] の順にクリックします。

6 有効にするワークフローに関連付けられたカスタム プロパティの名前を [名前] テキスト ボックスに入力します。

カスタマイズ可能なワークフロー名	関連付けられたプロパティ名
WFStubMachineProvisioned	ExternalWFStubs.MachineProvisioned
WFStubBuildingMachine	ExternalWFStubs.BuildingMachine
WFStubMachineDisposing	ExternalWFStubs.MachineDisposing
WFStubUnprovisionMachine	ExternalWFStubs.UnprovisionMachine
WFStubMachineRegistered	ExternalWFStubs.MachineRegistered
WFStubMachineExpired	ExternalWFStubs.MachineExpired

7 [値] テキスト ボックスを空欄にします。

ワークフローは、特定の値ではなく、プロパティの有無に依存します。

8 [OK] をクリックしてプロパティを保存します。

9 [OK] をクリックします。

結果

ワークフローは、このブループリントからプロビジョニングされる新しいマシンに対して有効になります。

メニュー操作ワークフローの設定

メニュー操作ワークフローのカスタマイズ後、ユーザーが vRealize Automation コンソールで使用できるようにするには、追加の設定が必要です。

メニュー操作の設定

メニュー操作を設定するには、操作の設定ファイルを作成し、それを Model Manager にインストールします。

手順

1 操作構成ファイルの作成

操作構成ファイルは、メニュー操作ワークフローで必要になります。このファイルを使用して、vRealize Automation コンソールのカスタム メニュー オプションの項目を指定します。これには、表示テキスト、オプションへのアクセス権をどのロールに持たせるか、オプションが使用可能になるマシン状態などの設定があります。

2 Model Manager への操作のインストール

CloudUtil コマンドライン ユーティリティを使用して Model Manager に操作をインストールします。

次のステップ

メニュー操作をサービス カタログで使用する場合は、ユーザーに対して資格を付与できるように、サービス カタログに登録する必要があります。サービス カタログを使用して新しいメニュー操作を登録する。

操作構成ファイルの作成

操作構成ファイルは、メニュー操作ワークフローで必要になります。このファイルを使用して、vRealize Automation コンソールのカスタム メニュー オプションの項目を指定します。これには、表示テキスト、オプションへのアクセス権をどのロールに持たせるか、オプションが使用可能になるマシン状態などの設定があります。

手順

1 XML ファイルを新規作成します。

```
<?xml version="1.0" encoding="utf-8"?>
```

2 ルート要素 customOperations を作成します。

```
<customOperations xmlns="http://www.dynamicops.com/schemas/2009/OperationConfig/">
</customOperations>
```

この要素は、XML 名前空間 http://www.dynamicops.com/schemas/2009/OperationConfig/ を指定する必要があります。

3 定義する操作それぞれについて、customOperations 内に operation 要素を追加します。

```
<operation name="WFMachineMenu1" displayName="Execute Machine Menu task">
</operation>
```

operation 要素に指定できる属性は、次のとおりです。

属性	説明
name	この操作が実行するワークフローの名前。
displayName	マシン メニューのオプションに使用する説明ラベル。

4 メニューの捜査権限を付与するロールを指定します。

a authorizedTasks 要素を追加します。

```
<operation name="WFMachineMenu1" displayName="Execute Machine Menu task">
  <authorizedTasks>
  </authorizedTasks>
</operation>
```

b 操作権限を付与するそれぞれのロールに、次のように task 要素を追加します。

```
<authorizedTasks>
  <task>VRM User Custom Event</task>
  <task>VRM Support Custom Event</task>
  <task>Group Administrator Custom Event</task>
  <task>Enterprise Administrator Custom Event</task>
  <task>VRM Administrator Custom Event</task>
</authorizedTasks>
```

task 要素の有効なコンテンツは次のとおりです。

要素のコンテンツ	説明
VRM User Custom Event	すべてのユーザーに操作権限を付与します。
VRM Support Custom Event	サポート ユーザーに操作権限を付与します。
Group Administrator Custom Event	ビジネス グループ マネージャに操作権限を付与します。
Enterprise Administrator Custom Event	ファブリック管理者に操作権限を付与します。
VRM Administrator Custom Event	IaaS 管理者にのみ操作権限を付与します。

5 （オプション） 操作を可能とするマシンの状態を指定します。

a machineStates 要素を追加します。

```
<operation name="WFMachineMenu1" displayName="Execute Machine Menu task">
  <machineStates>
  </machineStates>
</operation>
```

b 操作を可能とする状態のそれぞれに state 要素を追加します。

```
<machineStates>
  <state>On</state>
  <state>Off</state>
</machineStates>
```

値には、任意のマシン状態を指定できます。マシン状態の完全なリストについては、『仮想プラットフォームの IaaS 構成』、『物理マシンの IaaS 構成』、または『IaaS Configuration for Cloud Platforms』を参照してください。

要素を省略すると、すべてのマシン状態で操作が可能となります。

例

次は、完全な操作構成ファイルの一例です。

```
<?xml version="1.0" encoding="utf-8" ?>
<customOperations xmlns="http://www.dynamicops.com/schemas/2009/OperationConfig/">
  <operation name="WFMachineMenu1" displayName="Execute Machine Menu task">
    <authorizedTasks>
      <task>VRM User Custom Event</task>
      <task>VRM Support Custom Event</task>
      <task>Group Administrator Custom Event</task>
      <task>Enterprise Administrator Custom Event</task>
      <task>VRM Administrator Custom Event</task>
    </authorizedTasks>
    <machineStates>
      <state>On</state>
      <state>Off</state>
    </machineStates>
  </operation>
</customOperations>
```

Model Manager への操作のインストール

CloudUtil コマンドライン ユーティリティを使用して Model Manager に操作をインストールします。

前提条件

操作構成ファイルの作成。

手順

1 管理者権限のコマンド プロンプトを開きます。

2 CloudUtil.exe コマンドを次の引数を使用して実行します。

- ```
  CloudUtil.exe Operation-Create -c <path to operation definition file>
  ```
- 必要に応じて、Model Manager ホストを指定して、エラーが発生した場合にスタック トレースを申請することができます。

  ```
  CloudUtil.exe Operation-Create -c <path to operation definition file>
  --repository <Model Manager Root URI> -v
  ```

次のステップ

メニュー操作をサービス カタログで使用する場合は、ユーザーに対して資格を付与できるように、サービス カタログに登録する必要があります。サービス カタログを使用して新しいメニュー操作を登録する。

サービス カタログを使用して新しいメニュー操作を登録する

新しいメニュー操作をインストールした後、ワークフロー開発者はその操作をサービス カタログに登録して、ユーザーが使用できるようにする必要があります。

前提条件

- メニュー操作の設定。
- IaaS Model Manager ホストで、管理者権限を持つローカル ユーザーとして Windows にログインします。

手順

1 管理者権限のコマンド プロンプトを開きます。

2 IaaS のルート インストール ディレクトリに移動します。

通常の配置では C:\Program Files (x86)\VMware\vCAC です。

3 Server\Model Manager Data\Cafe に移動します。

4 次のコマンドを入力します。

Vcac-Config.exe RegisterCatalogTypes -v

次のステップ

テナント管理者またはビジネス グループ マネージャは、新しいアクションがサービス カタログのユーザーに使用可能になる前に、そのアクションの使用資格を付与する必要があります。詳細については、『テナント管理』を参照してください。

メニュー操作ワークフローを有効にするためのブループリントの設定

ブループリントのセキュリティ設定を更新することにより、特定のブループリントからプロビジョニングされるマシンのメニュー操作ワークフローを有効にします。

前提条件

テナント管理者またはビジネス グループ マネージャとして vRealize Automation にログインします。
  ```

手順

1 [設計] - [ブループリント] を選択します。

2 ブループリントの名前をポイントし、[編集] をクリックします。

3 [アクション] タブをクリックします。

4 チェックボックスをクリックして、有効にする操作を選択します。

5 [OK] をクリックします。

結果

このブループリントからプロビジョニングされるマシンのメニュー操作が有効になり、操作の構成ファイルで指定されたすべてのユーザー ロールで使用可能になります。

次のステップ

メニュー操作をサービス カタログで使用する場合は、ユーザーに対して資格を付与する必要があります。詳細については、『テナント管理』を参照してください。

ワークフローの以前のリビジョンに戻す

[[ワークフローのロード]] ダイアログには Model Manager のワークフローのすべてのリビジョンが表示されるので、ワークフローの全バージョンの履歴にアクセスできます。

Model Manager にワークフローを送信するたびに、リビジョンとタイムスタンプが更新されます。

前提条件

vRealize Automation Designer コンソールを起動します。

手順

1 [ロード] をクリックします。

2 戻したいワークフローのリビジョンを選択します。

VMware によって提供される、元のワークフローはリビジョン 0（ゼロ）です。

3 [OK] をクリックします。

4 [送信] をクリックし、Model Manager でワークフローを更新します。

結果

以前のリビジョンが、Model Manager で最新のリビジョンになります。たとえば、ワークフローのリビジョン 1 および 2 を作成し、その後でリビジョン 0 をロードして保存すると、リビジョン 0 と 3 は同一になり、そのワークフローを VMware によって提供されるバージョンに戻します。

ワークフローと分散管理

スキルを使用して、ワークフローの実行を特定の Distributed Execution Manager に制限できます。

スキルはワークフローと DEM ワーカー インスタンスの両方に適用可能なタグに似ています。どのスキルにも関連付けられていないワークフローは、すべての DEM ワーカーが実行できます。1 つまたは複数のスキルに関連付けられているワークフローは、そのすべての同じスキルに関連付けられている DEM ワーカーのみが実行できます。

スキルは、特定のワークフローが、特定の前提条件があるホストにインストールされている DEM を必要とする場合に役に立ちます。たとえば、クラウド プロビジョニング ワークフローを、Amazon の URL へのネットワーク アクセスが可能なホストで実行されている特定の DEM に制限できます。

またスキルは、ワークフローを特定のデータセンターの場所に関連付ける場合にも使用できます。たとえば、1 つの DEM をボストンのデータセンターにインストールし、もう 1 つをロンドンのデータセンターにインストールすると、スキルを使用して特定の操作を一方のデータセンターからもう一方のデータセンターに指示することができます。

スキルを使用したワークフローと DEM ワーカーの関連付け

Model Manager にスキルを追加し、そのスキルを 1 つ以上のワークフローおよび DEM ワーカーに関連付けることによって、特定の DEM ワーカーまたは一連のワーカー インスタンスをワークフローに関連付けます。

前提条件

vRealize Automation Designer コンソールを起動します。

手順

1 リボン上で [スキルの管理] をクリックします。

2 [スキルの管理] ダイアログの左上にあるテキスト フィールドに新しいスキルの名前を入力し、[追加] ボタンをクリックします。

スキル名は一意である必要があります。新しいスキルの名前が既存のスキル名と重複している場合、[追加] ボタンは使用可能になりません。

3 左側のリストで、スキルの名前を選択します。

4 1 つ以上の DEM ワーカーにスキルを関連付けます。

a Distributed Execution Manager の横にある [追加] アイコン（  ）をクリックします。

b [DEM の選択] ダイアログで、1 つ以上の DEM ワーカー インスタンスを選択します。

c [OK] をクリックします。

5 1 つまたは複数のワークフローにスキルを関連付けます。

a ワークフローの横にある [追加] アイコン（  ）をクリックします。

b [ワークフローの選択] ダイアログで、1 つ以上のワークフローを選択します。

c [OK] をクリックします。

このスキルに関連付けられたワークフローは、このスキルに関連付けられた DEM ワーカーのみが実行できます。

6 スキルを追加し、DEM ワーカーとワークフローへの関連付けが終了したら、[OK] をクリックして [スキルの管理] ダイアログを閉じ、Model Manager に変更内容を保存します。

スキルと DEM ワーカー間の関連付けの削除

スキルと DEM ワーカー間の関連付けを削除すると、そのワーカー インスタンスはそのスキルに関連付けられたワークフローを実行できなくなります。

前提条件

vRealize Automation Designer コンソールを起動します。

手順

1. リボン上で [スキルの管理] をクリックします。
2. [スキルの管理] ダイアログ ボックスで、左のリストからスキルの名前を選択します。
3. Distributed Execution Manager のリストから 1 つ以上の DEM ワーカー インスタンスの名前を選択し、[削除] アイコンをクリックします（  ）。
4. [OK] をクリックして [スキルの管理] ダイアログを閉じ、Model Manager に対する変更を保存します。

スキルとワークフローの関連付けを削除する

スキルとワークフローの関連付けを削除すると、そのワークフローは同じスキルに関連付けられている DEM ワーカーに制限されなくなります。

前提条件

vRealize Automation Designer コンソールを起動します。

手順

1. リボン上で [スキルの管理] をクリックします。
2. [スキルの管理] ダイアログ ボックスで、左のリストからスキルの名前を選択します。
3. ワークフローのリストから 1 つまたは複数のワークフローの名前を選択し、[削除] アイコン（  ）をクリックします。
4. [OK] をクリックして [スキルの管理] ダイアログを閉じ、Model Manager に対する変更を保存します。

スキルを削除する

スキルを削除すると、DEM ワーカーおよびワークフローとのすべての関連付けも削除されます。

前提条件

vRealize Automation Designer コンソールを起動します。

手順

1. リボン上で [スキルの管理] をクリックします。
2. [スキルの管理] ダイアログ ボックスで、左のリストからスキルの名前を選択します。

3 スキルのリストの最上部にある [削除] アイコン（  ）をクリックします。

スキルの削除を確認すると、その名前の表示が薄くなり、削除対象としてマークされたことを示します。

4 [OK] をクリックして [スキルの管理] ダイアログを閉じ、変更を Model Manager に保存します。スキル、および DEM とワークフローとの関連付けを削除しない場合は、[キャンセル] をクリックします。

CloudUtil コマンド リファレンス

このセクションは、CloudUtil コマンド ライン インターフェイスのコマンド リファレンスです。

CloudUtil は、vRealize Automation Designer 用のコマンド ライン インターフェイスです。これらのコマンドは、Designer を実行中の Windows マシンで実行します。Windows マシンのデフォルトのインストール場所は、`C:\Program Files (x86)\VMware\vCAC\Design Center` です。

注： CloudUtil コマンドでは、Model Manager は `repository` として、Distributed Execution Manager (DEM) は `agent` として参照されます。

DEM コマンド

DEM コマンドを使用すると、Model Manager に登録された Distributed Execution Manager のリストを表示し、スキルと DEM の間の関連付けを追加または削除できるようになります。

DEM-Add-Skills

登録済みの Distributed Execution Manager にスキルを関連付けます。

概要

```
CloudUtil.exe DEM-Add-Skills -n|--name <Name> -s|--skills <Skills> [--repository <Model Manager Root URI>] [-v|--verbose]
```

DEM-Add-Skills の引数

引数	説明
-n | - -name	登録済みの Distributed Execution Manager の名前。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-s | - -skills	この Distributed Execution Manager に関連付けるスキルのカンマ区切りリスト。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

注： スキルが Model Manager に存在している必要があります。「Skill-Install」を参照してください。

DEM-List

Model Manager に登録されているすべての Distributed Execution Manager とそれらに関連付けられているスキルを一覧表示します。

概要

```
CloudUtil.exe DEM-List [--repository <Model Manager Root URI>] [-v|--verbose]
```

DEM-List の引数

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

DEM-Remove-Skills

スキルと Distributed Execution Manager との間の関連付けを削除します。

概要

```
CloudUtil.exe DEM-Remove-Skills -n|--name <Name> -s|--skills <Skills> [--repository <Model Manager
Root URI>] [-v|--verbose]
```

DEM-Remove-Skills の引数

引数	説明
-n | - -name	登録済みの Distributed Execution Manager の名前。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-s | - -skills	この Distributed Execution Manager から削除するスキルのカンマ区切りリスト。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

ファイル コマンド

ファイル コマンドを使用することで、Model Manager 内のファイル（通常はスクリプト）の保存と管理ができます。

File-Export

Model Manager からファイルをエクスポートします。

概要

```
CloudUtil.exe File-Export -n|--name <Name> -o|--output <Output File> [-i|--iteration <Iteration>] [--
repository <Model Manager Root URI>] [-v|--verbose]
```

File-Export の引数

引数	説明
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager のファイルのわかりやすい名前です。

引数	説明
-o | - -output	ファイル出力用のパスです。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-Import

Model Manager にファイルをインポートします。

概要

```
CloudUtil.exe File-Import -n|--name <Name> -f|--filename <File Name> [-d|--description <Description>]
[-i|--iteration <Iteration>] [--repository <Model Manager Root URI>] [-v|--verbose]
```

File-Import の引数

引数	説明
-d | - -description	（オプション）ファイルの説明です。
-f | - -filename	Model Manager にインポートするファイルのパスです。
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager でファイルに割り当てるわかりやすい名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-List

Model Manager にインポートされたすべてのファイルを一覧表示します。

概要

```
CloudUtil.exe File-List [--repository <Model Manager Root URI>] [-v|--verbose]
```

File-List の引数

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-Remove-All

Model Manager から指定されたファイルのバージョンにおけるすべてのリビジョンを削除します。

概要

```
CloudUtil.exe File-Remove-All -n|--name <Name> [-i|--iteration <Iteration>]
[--repository <Model Manager Root URI>] [-v|--verbose]
```

File-Remove-All の引数

表 1-20.

引数	説明
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager のファイルのわかりやすい名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-Remove-Rev

Model Manager から特定のファイルのリビジョンを削除します。

概要

```
CloudUtil.exe File-Remove-Rev -n|--name <Name> -r|--revision <Revision> [-i|--iteration <Iteration>]
[--repository <Model Manager Root URI>] [-v|--verbose]
```

File-Export の引数

引数	説明
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager のファイルのわかりやすい名前です。
-r | - -revision	削除するファイルのリビジョンです。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-Rollback

Model Manager のファイルを指定されたリビジョンに戻します。

概要

```
CloudUtil.exe File-Rollback -n|--name <Name> -r|--revision <Revision> [-i|--iteration <Iteration>] [--
repository <Model Manager Root URI>] [-v|--verbose]
```

File-Rollback の引数

表 1-21.

引数	説明
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager のファイルのわかりやすい名前です。
-r | - -revision	戻すファイルのリビジョンです。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

File-Update

Model Manager の以前にインポートされたファイルを新しいリビジョンで更新します。

概要

```
CloudUtil.exe File-Update -n|--name <Name> -f|--filename <File Name> [-i|--iteration <Iteration>] [--
repository <Model Manager Root URI>] [-v|--verbose]
```

File-Update の引数

引数	説明
-f | - -filename	更新されたファイルのパスです。
-i | - -iteration	（オプション） Model Manager のファイルのバージョン文字列です。デフォルトは **0.0** です。
-n | - -name	Model Manager のファイルのわかりやすい名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

操作コマンド

操作コマンドを使用することで、Model Manager のカスタム操作を管理できます。

Operation-Create

操作定義ファイルに基づいてマシン上で実行できるカスタム操作または操作のセットを作成します。

概要

```
CloudUtil.exe Operation-Create -c|--operationConfig <Operation Definition File> [--repository <Model
Manager Root URI>] [-v|--verbose]
```

Operation-Create の引数

引数	説明
-c | - -operationConfig	操作定義ファイル (XML) のパスです。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Operation-Delete

Model Manager からカスタム操作を削除します。

概要

```
CloudUtil.exe Operation-Delete -n|--name <Name> [--force] [--repository <Model Manager
Root URI>] [-v|--verbose]
```

Operation-Delete の引数

引数	説明
- -force	（オプション）操作を強制的に削除します。
-n | - -name	Model Manager のカスタム操作の名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Operation-List

Model Manager のすべてのカスタム操作を一覧表示します。

概要

```
CloudUtil.exe Operation-List [--repository <Model Manager Root URI>] [-v|--verbose]
```

Operation-List の引数

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

スキル コマンド

スキル コマンドを使用することで、Distributed Execution Manager とワークフローに関連付けられたスキルを管理できます。

Skill-Install

Model Manager にスキルをインストールします。

概要

```
CloudUtil.exe Skill-Install -n|--name <Name> [--repository <Model Manager Root URI>] [-v|--verbose]
```

Skill-Install の引数

引数	説明
-n | - -name	Model Manager のスキルの名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Skill-List

Model Manager にインストールされているすべてのスキルを一覧表示します。

概要

```
CloudUtil.exe Skill-List [--repository <Model Manager Root URI>] [-v|--verbose]
```

Skill-List の引数

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Skill-Uninstall

Model Manager からスキルをアンインストールします。

概要

```
CloudUtil.exe Skill-Uninstall -n|--name <Name> [--repository <Model Manager Root URI>]
[-v|--verbose]
```

Skill-Uninstall の引数

引数	説明
-n | - -name	Model Manager からアンインストールするスキルの名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

注： スキルが Distributed Execution Manager またはワークフローと関連付けられている場合は、スキルをアンインストールできません。DEM-Remove-Skills または Workflow-Remove-Skills を参照してください。

ワークフロー コマンド

ワークフロー コマンドを使用することで、Model Manager のカスタマイズ可能な IaaS ワークフローと、ワークフローに関連付けられたスキルを管理できます。

Workflow-Add-Skills

スキルを Model Manager のワークフローに関連付けます。

```
CloudUtil.exe Workflow-Add-Skills -n|--name <Name> -s|--skills <Skills> [--repository <Model Manager Root URI>] [-v|--verbose]
```

表 1-22. Workflow-Add-Skills の引数

引数	説明
Name	Model Manager のワークフローの名前です。
Skills	このワークフローと関連付けるスキルのコンマ区切りのリストです。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

注： スキルが Model Manager に存在している必要があります。「Skill-Install」を参照してください。

Workflow-List

Model Manager にインストールされているすべてのワークフローと関連付けられたスキルを一覧表示します。

```
CloudUtil.exe Workflow-List [--repository <Model Manager Root URI>] [-v|--verbose]
```

表 1-23. Workflow-List の引数

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Workflow-Remove-Skills

Model Manager のスキルとワークフローの関連付けを削除します。

概要

```
CloudUtil.exe Workflow-Remove-Skills -n|--name <Name> -s|--skills <Skills> [--repository
<Model Manager Root URI>] [-v|--verbose]
```

Workflow-Remove-Skills の引数

引数	説明
-n | - -name	Model Manager のワークフローの名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-s | - -skills	このワークフローから削除するスキルのコンマ区切りのリストです。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Workflow-Rollback

ワークフローを指定されたリビジョンに戻します。

概要

```
CloudUtil.exe Workflow-Rollback -n|--name <Name> -r|--revision <Revision> [--repository <Model
Manager Root URI>] [-v|--verbose]
```

Workflow-Rollback の引数

引数	説明
-n | - -name	Model Manager のワークフローの名前です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-r | - -revision	戻すワークフローのリビジョンです。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Workflow-Update

カスタマイズ可能なワークフローを新しいリビジョンで更新します。

```
CloudUtil.exe Workflow-Update -f|--filename <File Name> -n|--name <Name> [-d|--description
<Description>] [--repository <Model Manager Root URI>] [-v|--verbose]
```

表 1-24. Workflow-Update の引数

引数	説明
File Name	更新されたワークフローを含むファイル (XAML) のパスです。
Name	更新するワークフローの名前です。

表 1-24. Workflow-Update の引数 （続き）

引数	説明
Description	（オプション）ワークフローの説明です。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

インポート コマンド

インポート コマンドを使用することで、1 つ以上の仮想マシンを vRealize Automation 環境にインポートできます。

Machine-BulkRegisterExport

仮想マシンを vRealize Automation 環境にインポートするために使用する CSV データ ファイルを作成します。

概要

```
CloudUtil.exe Machine-BulkRegisterExport [-b|--blueprint] [-m|--managed] [-e|--exportNames] [-p|--properties] -f|--filename <Value> [-g|--group <Value>] [-i|--ignore] [-o|--owner <Value>] [-t|--machinetype <Value>] [-n|--resourceName <Value>] [-r|--resourceType <Value>] [--repository <Value>] [-sn|--sourcename <Value>] [-st|--sourcetype <Value>] -u|--user <value> [-v|--verbose]
```

Machine-BulkRegisterExport 引数

表 1-25.

引数	説明
-b | - -blueprint	（オプション）ブループリント名を含みます。
-e | - -exportNames	（オプション）GUID ではなく名前をエクスポートします。
-f | - -filename	マシン名のリストを含む CSV データ ファイルの名前を指定します（たとえば `filename.csv`）。ファイルは、デフォルトでは現在のパスに保存されます。保存したいディレクトリへの完全パスを指定することもできます。
-g | - -group	（オプション）ビジネス グループ名を指定します（たとえば、エンジニアリング）。
-i | - -ignore	（オプション）無効な引数を無視します。
-m | - -managed	（オプション）管理対象の仮想マシンをエクスポートします。デフォルトでは管理対象外の仮想マシンをエクスポートします。
-n | - -resourceName	（オプション）リソース名別にフィルタするには、コンピュート リソースまたはエンドポイントの名前を指定します。
-o | - -owner	（オプション）インポートされた仮想マシンの所有者を指定します（たとえば jsmith）。
-p | - -properties	（オプション）管理対象の仮想マシンのプロパティをエクスポートします。
-r | - -resourceType	（オプション）リソース タイプ別にフィルタするには、コンピュート リソースの場合は 1、エンドポイントの場合は 2 を指定します。

表 1-25. （続き）

引数	説明
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-sn | - -sourcename	（オプション）クラスタまたはエンドポイントの名前を指定します。
-st | - -sourcetype	（オプション）ソース タイプにクラスタまたはエンドポイントを指定します。
-t | - -machinetype	（オプション）エクスポートするマシン タイプを指定します（たとえば、仮想、物理、クラウド、AppService、vApp）。
-u | - -user	バルク登録を実行する、ファブリック管理者を指定します。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。

Machine-BulkRegisterImport

ターゲットの vRealize Automation 環境に、1 台または複数の仮想マシンをインポートします。

概要

```
CloudUtil.exe Machine-BulkRegisterImport [-b|--batch][-d|--delay <value>] -f|--filename <value> [-i|--ignore] [-h|--humanreadable] -n|--name <value> [--repository <value>] [-s|--skipUser] -t|--time <value> -u|--user <value> [-v|--verbose] [-w|--whatIf]
```

Machine-BulkRegisterImport 引数

表 1-26.

引数	説明
-b | - -batch	（オプション）バッチ サイズ。
-d | - -delay	（オプション）処理の遅延時間を hh:mm:ss の形式で指定します（02:20:10 など）。
-f | - -filename	マシン名のリストを含む CSV データ ファイルの名前を指定します。たとえば、`filename.csv` のようになります。
-h | --humanreadable	（オプション）入力ファイルには GUID ではなく仮想マシン名が含まれます。
-i | - -ignore	（オプション）登録済みまたは管理対象の仮想マシンを無視します。
-n | - -name	ターゲット vRealize Automation へのインポートを実行する作業キューの名前を指定します。
- -repository	（オプション）Model Manager のルート URI（http://*hostname*/repository など）。デフォルトは、CloudUtil 構成ファイル内の「appSettings」セクションの repositoryAddress キーで指定されます。
-s | - -skipUser	（オプション）ユーザーの有無を検証せずに、マシンの所有者を CSV データ ファイルの [所有者] 列に記載された値に設定します。このオプションを選択すると、インポートに必要な時間を減らすことができます。

表 1-26. （続き）

引数	説明
-t | - -time	ワークフローの開始日時を MM/DD/YYYY hh:mm GMT の形式で指定します（04/18/2014 10:01 GMT など）。指定される開始日時はサーバのローカル時間を想定したもので、ユーザーのワークステーションのローカル時間ではありません。
-u | - -user	バルク登録を実行する、ファブリック管理者を指定します。
-v | - -verbose	（オプション）エラーが発生する場合、例外メッセージだけではなくスタック トレースも出力します。
whatif	（オプション）CSV ファイルを検証するために設定されますが、仮想マシンはインポートされません。

vRealize Automation ワークフロー アクティビティ リファレンス

VMware は、ワークフローのカスタマイズで使用するためのワークフロー アクティビティのライブラリを vRealize Automation Designer で提供しています。

注： CDK は vRealize Automation 7.0 から廃止される予定です。vRealize Orchestrator ワークフローを使用すると、以前に CDK によって対処したユースケースに対処できます。

制御フロー、フローチャート、プリミティブ、コレクション、エラー処理を含む、Windows Workflow Foundation アクティビティの 5 つのカテゴリも vRealize Automation Designer に含まれています。

このセクションでは、vRealize Automation Designer で `DynamicOps.Repository.Activities` および `DynamicOps.Cdk.Activities` の名前空間に含まれている IaaS ワークフロー アクティビティのリファレンスを提供します。vRealize Orchestrator ワークフローの呼び出しに関連するアクティビティについては、vRealize Orchestrator ワークフロー アクティビティの使用で説明されています。

注： IaaS アクティビティ ライブラリでは、Model Manager は `repository` と呼ばれています。

DynamicOps.Repository.Activities

`DynamicOps.Repository.Activities` 名前空間には、IaaS ワークフローの基本的なワークフロー アクティビティが含まれています。

注： CDK は vRealize Automation 7.0 から廃止される予定です。vRealize Orchestrator ワークフローを使用すると、以前に CDK によって対処したユースケースに対処できます。

AddLink

`DataServiceContext` が追跡している一連のオブジェクトに、指定されたリンクを追加します。

表 1-27. AddLink アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	リンクの追加先となる `DataServiceContext`。
Source	Object	新しいリンクのソース オブジェクト。

表 1-27. AddLink アクティビティの入力パラメータ （続き）

引数	タイプ	説明
SourceProperty	文字列	関連オブジェクトを返すソース オブジェクト上のナビゲーション プロパティの名前。
Target	Object	新しいリンクによってソース オブジェクトと関連付けられるオブジェクト。

AddObject

DataServiceContext が追跡している一連のオブジェクトに、指定されたオブジェクトを追加します。

表 1-28. AddObject アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	オブジェクトの追加先となる DataServiceContext。
Instance	Object	DataServiceContext が追跡するオブジェクト。

AttachTo

指定したリソースの追跡を開始することを DataServiceContext に通知します。

表 1-29. AttachTo アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	リソースを追跡する DataServiceContext。
Instance	Object	DataServiceContext が追跡するリソース。リソースは Unchanged の状態で接続されます。

CreateRepositoryServiceContext<T>

Model Manager にロードされているモデルのコンテキストを作成します。

vRealize Automation Designer でワークフローにこのアクティビティを追加する際に、RepositoryServiceContext クラスから継承するクラスを選択する必要があります。

表 1-30. CreateRepositoryServiceContext<T> アクティビティの入力パラメータ

引数	タイプ	説明
Uri	URI	（オプション）モデルへの接続に使用するルート URI。
Username	文字列	（オプション）コンテキストへの接続に使用するユーザー名。

表 1-31. CreateRepositoryServiceContext<T> アクティビティの出力パラメータ

引数	タイプ	説明
Result	RepositoryServiceContext	ワークフローにアクティビティが追加されたときに選択されたクラスのインスタンスが、特定のタイプとして返されます。

DeleteLink

DataServiceContext で追跡するリンクのリスト内で、リンクの状態を削除済みに変更します。

表 1-32. DeleteLink アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	リンクの削除元となる DataServiceContext。
Source	Object	削除としてマークされるリンクのソース オブジェクト。
SourceProperty	文字列	ターゲット オブジェクトへのアクセスに使用するソース オブジェクト上のナビゲーション プロパティの名前。
Target	Object	ソース オブジェクトにバインドされているリンクに関連するターゲット オブジェクト。ターゲット オブジェクトのタイプは、ソース プロパティまたはサブタイプによって識別されるタイプである必要があります。

DeleteObject

指定オブジェクトの状態を、DataServiceContext 内で削除済みに変更します。

表 1-33. DeleteObject アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	リソースの削除元となる DataServiceContext。
Instance	Object	削除済みの状態に変更される、追跡対象エンティティ。

InvokeRepositoryWorkflow

Model Manager にインストールされているワークフローを実行します。

表 1-34. InvokeRepositoryWorkflow アクティビティの入力パラメータ

引数	タイプ	説明
WorkflowType	WorkflowDefinition エンティティ	実行するワークフローです。
WorkflowInputs	Dictionary<string, object>	（オプション）ワークフローに対する入力です。
CallingInstance	WorkflowInstance エンティティ	（オプション）実行されたワークフローを呼び出し、実行されたワークフローに返すワークフローです。

LoadProperty

データ サービスから指定されたプロパティの保留中コンテンツをロードします。

表 1-35. LoadProperty アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	プロパティのロード元となる DataServiceContext です。
Instance	Object	ロードするプロパティを含むエンティティです。
InstanceProperty	文字列	指定したエンティティの、ロードするプロパティの名前です。

SaveChanges

DataServiceContext がストレージにトラッキングしている変更を保存します。

表 1-36. SaveChanges アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	保存する変更をトラッキングしている DataServiceContext です。

SetLink

指定されたオブジェクトの間に新しいリンクが存在すること、およびそのリンクが SourceProperty 引数で指定されたプロパティによって表されることを DataServiceContext に通知します。

表 1-37. SetLink アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	リンクを通知する DataServiceContext です。
Source	Object	新しいリンクのソース オブジェクト。
SourceProperty	文字列	新しいリンクのターゲット オブジェクトを識別するソース オブジェクトのプロパティです。
Target	Object	このメソッドを呼び出すことで初期化される新しいリンクに関連する子オブジェクトです。ターゲット オブジェクトは、SourceProperty によって識別されるタイプのサブタイプである必要があります。Target が Null に設定されている場合、呼び出しはリンク削除の操作を表します。

UpdateObject

DataServiceContext の指定されたオブジェクトの状態を、[修正済み] に変更します。

表 1-38. UpdateObject アクティビティの入力パラメータ

引数	タイプ	説明
DataServiceContext	RepositoryServiceContext	更新するエンティティをトラッキングする DataServiceContext です。
Instance	Object	[修正済み] の状態に割り当てる、トラッキングされたエンティティです。

DynamicOps.Cdk.Activities

DynamicOps.Cdk.Activities 名前空間には、IaaS ワークフローの高度なアクティビティが含まれています。

注： CDK は vRealize Automation 7.0 から廃止される予定です。vRealize Orchestrator ワークフローを使用すると、以前に CDK によって対処したユースケースに対処できます。

ExecutePowerShellScript

指定された名前で Model Manager に保存されている PowerShell スクリプトを実行します。

ExecutePowerShellScript アクティビティを使用する前に、CloudUtil File–Import コマンドを使用して Model Manager に実行するスクリプトをロードする必要があります。

表 1-39. ExecutePowerShellScript アクティビティの入力パラメータ

引数	タイプ	説明
ScriptName	文字列	実行するスクリプトの Model Manager での名前。
ScriptVersion	Object	(オプション)実行するスクリプトの Model Manager でのバージョン。デフォルトは 0.0 です。
MachineId	Guid	(オプション) 指定すると、マシンがロードされ、そのすべてのプロパティがスクリプトに渡されます。
Arguments	Dictionary<string,string>	スクリプトに渡す追加の引数。MachineId が指定され、引数と同じ名前のマシン プロパティ (大文字と小文字を区別しない) がある場合、マシン プロパティの値が引数の値をオーバーライドします。
PSModules	IEnumerable<string>	(オプション) コマンド実行中に PowerShell ランタイムにロードされるモジュール。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。

表 1-40. ExecutePowerShellScript アクティビティの出力パラメータ

引数	タイプ	説明
Output	Collection<PSObject>	スクリプトの出力 (該当する場合)。エラー時には例外をスローします。

`ExecutePowerShellScript` の出力の処理中に vRealize Automation Designer コンソールで Type `PSObject is not defined` というエラー メッセージを受信した場合は、次の手順を実行します。

1 [Designer] ペインの左下隅の [インポート] をクリックします。

2 [System.Management.Automation] アセンブリを選択します。

ExecuteSshScript

指定された名前で Model Manager に保存されている SSH スクリプトを実行します。

`ExecuteSshScript` アクティビティを使用する前に、`CloudUtil File-Import` コマンドを使用して Model Manager に実行するスクリプトをロードする必要があります。

表 1-41. ExecuteSshScript アクティビティの入力パラメータ

引数	タイプ	説明
ScriptName	文字列	実行するスクリプトの Model Manager での名前。
Host	文字列	スクリプトを実行するサーバの名前。
Username	文字列	ホストへの接続で使用するユーザー名。
Password	文字列	ホストへの接続で使用するパスワード。
ScriptVersion	Object	(オプション)実行するスクリプトの Model Manager でのバージョン。デフォルトは 0.0 です。
Timeout	TimeSpan	(オプション) スクリプトの実行がタイムアウトするまでの期間。デフォルトは 30 分です。

表 1-42. ExecuteSshScript アクティビティの出力パラメータ

引数	タイプ	説明
EnvironmentVariables	Dictionary<string, string>	スクリプト実行結果（該当する場合）。

GetMachineName

マシンの名前を取得します。

表 1-43. GetMachineName アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	名前を取得するマシンです。

表 1-44. GetMachineName アクティビティの出力パラメータ

引数	タイプ	説明
MachineName	文字列	`MachineId` によって識別されるマシンの名前です。

GetMachineOwner

マシンの所有者のユーザー名を取得します。

表 1-45. GetMachineOwner アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	所有者を取得するマシンです。

表 1-46. GetMachineOwner アクティビティの出力パラメータ

引数	タイプ	説明
Owner	文字列	`MachineId` によって識別されるマシンの所有者で、所有者がいない場合には Null になります。

GetMachineProperties

マシンに関連付けられているカスタム プロパティのリストを取得します。

表 1-47. GetMachineProperties アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	プロパティを取得するマシンです。

表 1-48. GetMachineProperties アクティビティの出力パラメータ

引数	タイプ	説明
Properties	Dictionary<string, string>	マシンのプロパティのリストです。値が暗号化されて格納されている場合には、復号化されて返されます。

GetMachineProperty

マシンの指定されたプロパティの値を取得します。

表 1-49. GetMachineProperty アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	プロパティを取得するマシン。
PropertyName	文字列	値が返されるプロパティの名前。
IsRequired	bool	必要なプロパティが見つからない場合、アクティビティは例外をスローし、それ以外は null を返します。

表 1-50. GetMachineProperty アクティビティの出力パラメータ

引数	タイプ	説明
PropertyValue	文字列	`PropertyName` で指定されたプロパティの値。暗号化された値は、復号化された状態で返されます。

GetScriptFromName

指定した名前で Model Manager に保存されているスクリプトのコンテンツを取得します。

表 1-51. GetScriptFromName アクティビティの入力パラメータ

引数	タイプ	説明
ScriptName	文字列	取得するスクリプトの Model Manager での名前です。
ScriptVersion	Object	（オプション）取得するスクリプトの Model Manager でのバージョンです。デフォルトは 0.0 です。

表 1-52. GetScriptFromName アクティビティの出力パラメータ

引数	タイプ	説明
ScriptContent	文字列	`ScriptName` によって識別されるスクリプトのコンテンツです。

InvokePowerShell

PowerShell コマンドを実行します。

表 1-53. InvokePowerShell アクティビティの入力パラメータ

引数	タイプ	説明
CommandText	文字列	実行するコマンド。
Arguments	IEnumerable<string>	（オプション）コマンドの引数。
Input	IEnumerable	（オプション）入力パイプライン。
IsScript	bool	（オプション）`CommandText` がスクリプトかどうかを示します。デフォルトは False です。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
Parameters	Collection	（オプション）PowerShell スクリプトにパラメータとして渡される名前/値ペアのコレクション。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。

表 1-53. InvokePowerShell アクティビティの入力パラメータ （続き）

引数	タイプ	説明
PowerShellVariables	Collection	（オプション）PowerShell ランタイムにコピーされる変数。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
PSModules	IEnumerable<string>	（オプション）コマンド実行中に PowerShell ランタイムにロードされるモジュール。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
Runspace	Runspace	（オプション）PowerShell 実行空間を作成してこの引数に提供することで、複数の PowerShell 呼び出しで同じ実行空間を再利用でき、パフォーマンスを向上させることができます。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。

表 1-54. InvokePowerShell アクティビティの出力パラメータ

引数	タイプ	説明
Output	Collection<PSObject>	コマンドの出力（該当する場合）。エラー時には例外をスローします。
Errors	Collection<ErrorRecord>	実行によるエラー（該当する場合）。

`ExecutePowerShellScript` の出力の処理中に vRealize Automation Designer コンソールで Type `PSObject is not defined` というエラー メッセージを受信した場合は、次の手順を実行します。

1 [Designer] ペインの左下隅の [インポート] をクリックします。

2 [System.Management.Automation] アセンブリを選択します。

InvokeSshCommand

SSH コマンドを実行します。

表 1-55. InvokeSshCommand アクティビティの入力パラメータ

引数	タイプ	説明
CommandText	文字列	実行するコマンド。
Host	文字列	コマンドの実行対象のサーバ名です。
Username	文字列	ホストへの接続で使用するユーザー名。
Password	文字列	ホストへの接続で使用するパスワード。
Timeout	TimeSpan	（オプション）コマンドの実行がタイムアウトになるまでの時間です。デフォルトは 30 分です。

表 1-56. InvokeSshCommand アクティビティの出力パラメータ

引数	タイプ	説明
EnvironmentVariables	Dictionary<string, string>	コマンドの出力（該当する場合）。エラー時には例外をスローします。

LogMachineEvent

マシン所有者が表示可能なユーザー ログにマシン イベントを記録します。

表 1-57. LogMachineEvent アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	ログに記録するイベントを生成するマシンです。
Message	文字列	ユーザー ログに書き込むメッセージです。
Type	文字列	ドロップダウン リストからメッセージ タイプを選択します（情報、警告、エラー）。

LogMessage

Distributed Execution Manager ログに記録します。

表 1-58. LogMessage アクティビティの入力パラメータ

引数	タイプ	説明
Message	文字列	DEM ログに書き込むメッセージです。
MessageCategory	文字列	ドロップダウン メニューからカテゴリを選択するか（[デバッグ]、[エラー]、[情報]、[トレース]）、カスタム カテゴリを入力します。
MessageSeverity	文字列	ドロップダウン メニューから重要度を選択します。`System.Diagnostics.TraceEventType` で提供される重要度のリストにバインドされます。

RunProcess

このアクティビティを実行する DEM と同じマシン上でプロセスを実行します。

注： vRealize Automation は、`RunProcess` アクティビティによって開始された処理からユーザーに対してユーザー インターフェイスを表示することはできません。そのため、これらの処理は、非インタラクティブ型である必要があります。DEM マシン上で実体なしのプロセスが残らないようにするには、プロセスを自己終了型にする必要もあります。

表 1-59. RunProcess アクティビティの入力パラメータ

引数	タイプ	説明
Command	文字列	DEM マシン上で実行する実行可能ファイルへのパス。
WorkingDirectory	文字列	（オプション）プロセスを実行するワーキング ディレクトリ。
Arguments	文字列	（オプション）コマンドに渡すコマンドライン引数のリスト。
WaitForExit	bool	（オプション）True の場合、ワークフローはプロセスが完了するのを待ってからワークフローを続行します。デフォルトは False です。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。

SendEmail

指定されたアドレスの設定に E メールを送信します。

表 1-60. SendEmail アクティビティの入力パラメータ

引数	タイプ	説明
To	IEnumerable<string>	E メールの宛先アドレスのリスト。
From	文字列	E メールの「送信元」フィールドに入力するアドレス。
Subject	文字列	E メールの件名の行。
Body	文字列	E メールの本文テキスト。
Host	文字列	発信 SMTP サーバのホスト名または IP アドレス。
Port	Integer	ホストで指定されたサーバの SMTP ポート。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
CC	IEnumerable<string>	(オプション) E メールでコピーするアドレスまたはアドレス一覧。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
Bcc	IEnumerable<string>	(オプション) E メールでブラインド コピーするアドレスまたはアドレス一覧。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
EnableSsl	bool	(オプション) SSL を使用するかどうかを示します。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
UserName	文字列	`Host` で指定された SMTP サーバで認証するユーザー名。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。
Password	文字列	`UserName` で指定されたユーザーのパスワード。 このオプションは、[プロパティ] ペインでのみ利用可能で、[Designer] ペインでは利用できません。

SetMachineProperty

マシン上でカスタム プロパティを作成または更新します。

表 1-61. SetMachineProperty アクティビティの入力パラメータ

引数	タイプ	説明
MachineId	Guid	カスタム プロパティを作成または更新するマシン。
PropertyName	文字列	作成または更新するプロパティの名前。
PropertyValue	文字列	プロパティを作成または更新する値。
IsEncrypted	bool	(オプション) プロパティの値を暗号化するかどうかを示します。
IsHidden	bool	(オプション) プロパティが非表示のプロパティであるかどうかを示します。
IsRuntime	bool	(オプション) 申請元ユーザーが申請時にプロパティ値を提供するかどうかを示します (vRealize Automation コンソールで [プロンプト表示] とマークされる場合と同じ)。

SetWorkflowResult

外部ワークフローの状態を [完了] または [失敗] に設定し、この状態が `ExternalWF.xml` の設定に受け取られます。

表 1-62. SetWorkflowResult アクティビティの入力パラメータ

引数	タイプ	説明
WorkflowId	Guid	状態を設定するワークフローです。
Next State	WorkflowState	ドロップダウン メニューから [完了] または [失敗] を選択します。