



Red Hat Process Automation Manager 7.8

Operator を使用した Red Hat OpenShift
Container Platform への Red Hat Process
Automation Manager 環境のデプロイメント

ガイド

Red Hat Process Automation Manager 7.8 Operator を使用した Red Hat OpenShift Container Platform への Red Hat Process Automation Manager 環境のデプロイメント

ガイド

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Deploying_a_Red_Hat_Process_Automation_Manager_environment_on_Red_Hat_OpenShift_Co file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本書では、Red Hat OpenShift Container Platform に Operator を使用して Red Hat Process Automation Manager 7.8 環境をデプロイする方法を説明します。

目次

前書き	3
第1章 RED HAT OPENSIFT CONTAINER PLATFORM における RED HAT PROCESS AUTOMATION MANAGER の概要	4
第2章 オーサリング環境のアーキテクチャー	5
単一のオーサリング環境	5
KIE Server のクラスタリングと複数の KIE Server の使用	6
Smart Router	6
高可用性オーサリング環境	6
第3章 OPENSIFT 環境に RED HAT PROCESS AUTOMATION MANAGER をデプロイする準備	9
3.1. RED HAT レジストリーに対してお使いの環境が認証されていることを確認する方法	9
3.2. KIE SERVER のシークレットの作成	9
3.3. BUSINESS CENTRAL へのシークレットの作成	10
3.4. AMQ ブローカー接続のシークレットの作成	11
3.5. SMART ROUTER のシークレットの作成	11
3.6. 外部データベースのカスタム KIE SERVER 拡張イメージのビルド	12
3.7. GIT フックの準備	14
3.8. NFS を使用した READWRITEMANY アクセスモードの永続ボリュームのプロビジョニング	15
3.9. S2I ビルドに使用する BUSINESS CENTRAL からのソースコードの展開	15
3.10. ネットワークが制限された環境でのデプロイメントの準備	16
3.11. オフラインで使用する MAVEN ミラーリポジトリの用意	17
第4章 OPENSIFT OPERATOR を使用した RED HAT PROCESS AUTOMATION MANAGER 環境のデプロイと管理	19
4.1. BUSINESS AUTOMATION OPERATOR のサブスクリプション	19
4.2. OPERATOR を使用した RED HAT PROCESS AUTOMATION MANAGER 環境のデプロイ	19
4.2.1. Business Automation Operator の使用による Red Hat Process Automation Manager 環境のデプロイメントの開始	20
4.2.2. 環境の基本設定の構成	20
4.2.3. 環境のセキュリティ設定の構成	22
4.2.4. 環境の Business Central 設定の構成	24
4.2.5. 環境のカスタム KIE Server 設定の構成	27
4.2.6. 環境の Smart Router 構成の設定	33
4.2.7. 環境のプロセスインスタンス移行 (PIM) 設定	34
4.3. OPERATOR を使用してデプロイした環境の変更	34
4.4. JVM 設定パラメーター	36
4.5. KIE SERVER のカスタムイメージの作成	38
4.5.1. 追加の RPM パッケージを含めたカスタムの KIE Server イメージの作成	38
4.5.2. 追加の JAR ファイルを使用したカスタム KIE Server イメージの作成	40
第5章 RED HAT OPENSIFT CONTAINER PLATFORM バージョン 3 のデプロイメントからの情報の移行 ...	42
5.1. BUSINESS CENTRAL での情報の移行	42
5.2. KIE SERVER の MYSQL データベースの移行	43
5.3. KIE SERVER の POSTGRES SQL データベースの移行	46
付録A バージョン情報	49

前書き

システムエンジニアは、Red Hat OpenShift Container Platform に Red Hat Process Automation Manager 環境をデプロイしてプロセスや他のビジネスアセットを開発または実行するインフラストラクチャを提供できます。OpenShift Operator を使用して、構造化された YAML ファイルに定義された環境をデプロイして、必要に応じてこの環境を維持して変更できます。

前提条件

- Red Hat OpenShift Container Platform バージョン 4 の環境を利用できる。現在のリリースがサポートする OpenShift Container Platform の正確なバージョンについては、「[Red Hat Process Automation Manager 7 でサポートされる構成](#)」を参照してください。
- OpenShift 環境で 4 ギガバイト以上のメモリーが利用できる。
- デプロイメントする OpenShift プロジェクトが作成されている。
- OpenShift Web コンソールを使用してプロジェクトにログインしている。
- 動的永続ボリューム (PV) のプロビジョニングが有効になっている。または、動的 PV プロビジョニングが有効でない場合は、十分な永続ボリュームが利用できる状態でなければなりません。デフォルトでは、デプロイされるコンポーネントには以下の PV サイズが必要です。
 - それぞれの KIE Server デプロイメントで、このデータベースに 1 つの 1Gi PV が必要になります。データベース PV のサイズは変更できます。複数の KIE Server をデプロイでき、それぞれに異なるデータベース PV が必要になります。この要件は、外部データベースサーバーを使用する場合には適用されません。
 - デフォルトでは、Business Central は 1 Gi 分の PV が必要です。Business Central 永続ストレージの PV サイズを変更できます。
 - Business Central Monitoring には、1 つの 64Mi PV が必要です。
 - Smart Router には、1 つの 64Mi PV が必要です。
- Business Central または Business Central Monitoring Pod のいずれかをスケーリングする予定がある場合は、OpenShift 環境では、**ReadWriteMany** モードで永続ボリュームがサポートされます。ご使用の環境がこのモードに対応していない場合は、NFS を使用してボリュームをプロビジョニングできます。OpenShift のパブリックおよび専用クラウドでのアクセスモードのサポートに関する情報は、「[アクセスモード](#)」を参照してください。

第1章 RED HAT OPENSIFT CONTAINER PLATFORM における RED HAT PROCESS AUTOMATION MANAGER の概要

Red Hat Process Automation Manager は、Red Hat OpenShift Container Platform 環境にデプロイすることができます。

この場合に、Red Hat Process Automation Manager のコンポーネントは、別の OpenShift Pod としてデプロイされます。各 Pod のスケールアップおよびスケールダウンを個別に行い、特定のコンポーネントに必要な数だけコンテナを提供できます。標準の OpenShift の手法を使用して Pod を管理し、負荷を分散できます。

以下の Red Hat Process Automation Manager の主要コンポーネントが OpenShift で利用できます。

- KIE Server (**実行サーバー (Execution Server)**とも呼ばれる) は、デシジョンサービス、プロセスアプリケーション、およびその他のデプロイ可能なアセット (**サービス** と総称される) を実行するインフラストラクチャー要素です。サービスのすべてのロジックは実行サーバーで実行されます。

通常、KIE Server にはデータベースサーバーが必要です。別の OpenShift Pod にデータベースサーバーを提供したり、別のデータベースサーバーを使用するように OpenShift で実行サーバーを設定したりできます。また、KIE Server では H2 データベースを使用できますが、使用する場合は、Pod をスケーリングできません。

一部のテンプレートでは、KIE Server Pod をスケールアップして、同一または異なるホストで実行するコピーに必要な数だけ提供できます。Pod をスケールアップまたはスケールダウンすると、そのコピーはすべて同じデータベースサーバーを使用し、同じサービスを実行します。OpenShift は負荷分散を提供しているため、要求はどの Pod でも処理できます。

KIE Server Pod を個別にデプロイし、サービスの異なるグループを実行することができます。この Pod もスケールアップやスケールダウンが可能です。複製された個別の KIE Server Pod を必要な数だけ設定することができます。

- Business Central は、オーサリングサービスに対する Web ベースのインタラクティブ環境です。また、管理および監視コンソールも提供します。Business Central を使用してサービスを開発し、それらを KIE Server にデプロイできます。また、Business Central を使用してプロセスの実行を監視することもできます。

Business Central は一元化アプリケーションです。複数の Pod を実行し、同じデータを共有する高可用性用に設定できます。

Business Central には開発するサービスのソースを保管する Git リポジトリが含まれます。また、ビルトインの Maven リポジトリも含まれます。設定に応じて、Business Central はコンパイルしたサービス (KJAR ファイル) をビルドイン Maven リポジトリに配置できます (設定した場合は外部 Maven リポジトリにも可能)。

- Business Central Monitoring は Web ベースの管理および監視コンソールです。KIE Server へのサービスのデプロイメントを管理し、監視情報を提供しますが、オーサリング機能は含まれません。このコンポーネントを使用して、ステージング環境および実稼働環境を管理できます。
- Smart Router は、KIE Server と、KIE Server と対話するその他のコンポーネントとの間の任意のレイヤーです。環境に、複数の KIE Server で実行するサービスが多数含まれる場合、Smart Router はすべてのクライアントアプリケーションに対応するエンドポイントを1つ提供します。クライアントアプリケーションは、サービスを要求する REST API 呼び出しを実行できます。Smart Router は、特定の要求を処理できる KIE Server を自動的に呼び出します。

OpenShift 内でさまざまな環境設定にこのコンポーネントおよびその他のコンポーネントを配置できます。

第2章 オーサリング環境のアーキテクチャー

Red Hat Process Automation Manager では、Business Central のコンポーネントに、オーサリングサービス用の Web ベースの対話型ユーザーインターフェースが含まれています。KIE Server のコンポーネントでこれらのサービスを実行します。

KIE Server は、データベースサーバーを使用して、プロセスサービスの状態を保存します。

Business Central を使用して、KIE Server 上でサービスをデプロイすることもできます。複数の KIE Server を使用して異なるサービスを実行して同じ Business Central から複数のサーバーを制御できます。

単一のオーサリング環境

単一のオーサリング環境では、Business Central のインスタンスが1つだけ実行されます。複数のユーザーが同時に Web インターフェースにアクセスできますが、パフォーマンスが制限される可能性があります。フェイルオーバー機能はありません。

Business Central には、開発したサービスの各種ビルドバージョン (KJARファイル/アーティファクト) を格納する、ビルトイン Maven リポジトリが含まれています。継続的インテグレーション/継続的デプロイメント (CI/CD) ツールを使用して、リポジトリからこのようなアーティファクトを取得し、必要に応じて移動できます。

Business Central は、ビルトインの Git リポジトリにソースコードを保存します (.niogit ディレクトリに保存)。組み込まれたインデックスメカニズムを使用して、サービス内でアセットをインデックス化します。

Business Central では、Maven リポジトリと Git リポジトリに永続ストレージを使用します。

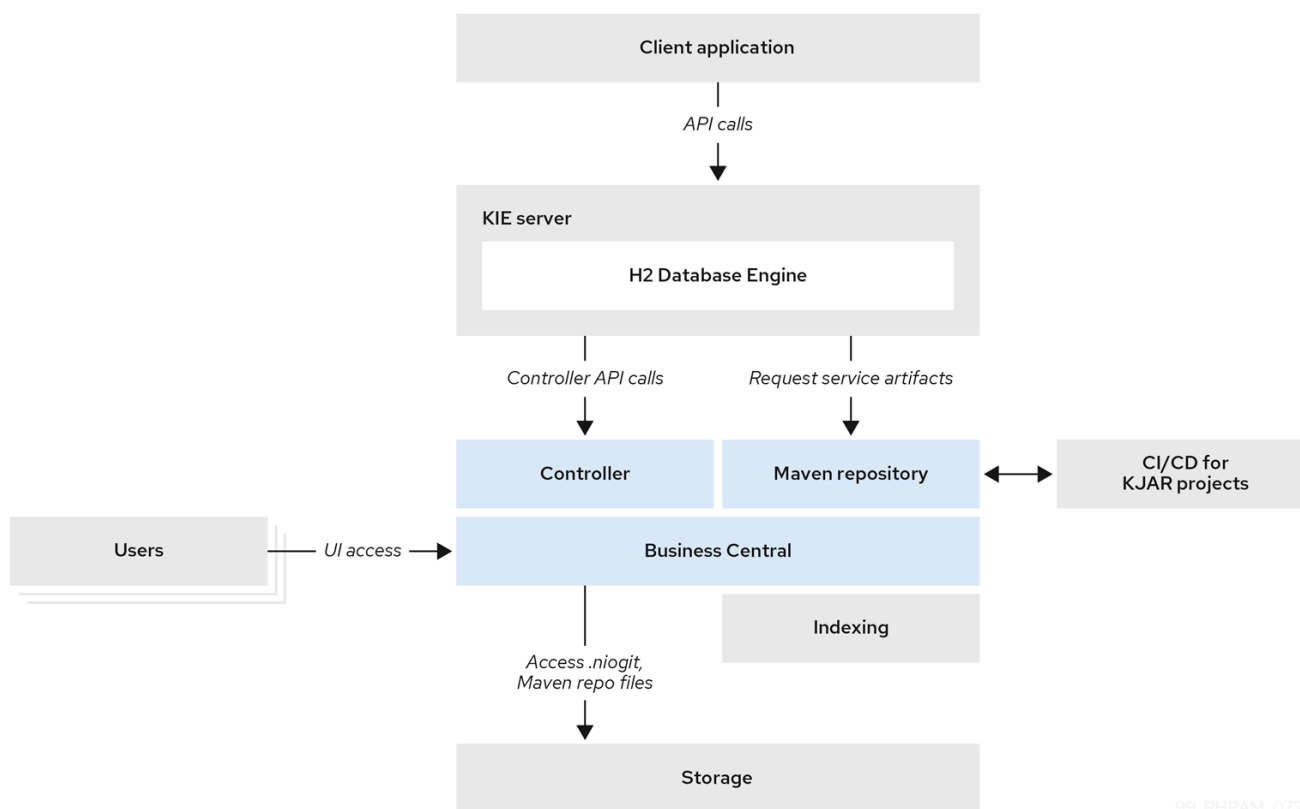
単一のオーサリング環境には、デフォルトで KIE Server が1台含まれています。この KIE Server は、ビルトインの H2 データベースエンジンを使用して、プロセスサービスの状態を保存します。

単一のオーサリング環境では、**コントローラストラテジー** を使用できます。Business Central には、KIE Server を管理できるコンポーネントである **コントローラー** が含まれています。Business Central に接続するように KIE Server を設定した場合、KIE Server は REST API を使用してコントローラーに接続します。この接続を使用すると、WebSocket が永続的に解放されます。コントローラストラテジーを使用する OpenShift デプロイメントでは、KIE Server はそれぞれ、Business Central コントローラーに接続するように初期設定されます。

Business Central ユーザーインターフェースを使用して KIE Server でサービスをデプロイしたり管理したりする場合、KIE Server はコントローラー接続の WebSocket を使用して要求を受け取ります。サービスをデプロイする場合は、KIE Server が Business Central の一部である Maven リポジトリから必要なアーティファクトを要求します。

クライアントアプリケーションは、REST API 経由で、KIE Server で実行されるサービスを使用します。

図2.1 単一のオーサリング環境のアーキテクチャ図



99_RHPAM_0720

KIE Server のクラスタリングと複数の KIE Server の使用

KIE Server Pod をスケールリングして、KIE Server のクラスター環境を実行できます。KIE Server をスケールリングするには、ビルトインの H2 データベースエンジンではなく、別の Pod でデータベースサーバーを使用するか、外部のデータベースサーバーを使用する必要があります。

クラスターデプロイメントでは、複数の KIE Server インスタンスが同じサービスを実行します。このようなサーバーは、Business Central コントローラーから同じ要求を受信できるように、同じサーバー ID を使用して Business Central コントローラーに接続します。Red Hat OpenShift Container Platform ではサーバー間の負荷分散が可能です。同じクライアントからの要求が別のインスタンスで処理される可能性があるため、クラスター化された KIE Server で実行するデシジョンサービスや Business Optimizer のサービスは、ステートレスでなければなりません。

独立した KIE Server を複数デプロイして、異なるサービスを実行することも可能です。このような場合、サーバーは異なるサーバー ID 値を指定して Business Central コントローラーに接続します。各サーバーにサービスをデプロイする場合は、Business Central UI を使用できます。

Smart Router

任意の Smart Router コンポーネントは、クライアントアプリケーションと KIE Server の間にレイヤーを提供します。独立した KIE Server を複数使用する場合に役立ちます。

クライアントアプリケーションは、異なる KIE Server で実行されるサービスを使用できますが、常に Smart Router に接続されます。Smart Router は自動的に、必要なサービスを実行する KIE Server に要求を渡します。また、Smart Router では、サービスのバージョン管理も可能で、追加の負荷分散レイヤーも提供されます。

高可用性オーサリング環境

高可用性 (HA) のオーサリング環境では Business Central Pod がスケールリングされるため、複数の Business Central インスタンスが実行されます。Red Hat OpenShift Container Platform は、ユーザー要求の負荷分散を提供します。この環境は、複数のユーザーに最適なパフォーマンスを提供し、フェイ

ルオーバーをサポートします。

Business Central の各インスタンスには、構築されたアーティファクト用の Maven リポジトリが含まれており、ソースコードには **.niogit** の Git リポジトリを使用します。このインスタンスは、リポジトリ用に共有の永続ストレージを使用します。このストレージには、**ReadWriteMany** アクセス権のある永続ボリュームが必要です。

Red Hat DataGrid のインスタンスは、Business Central で開発されたすべてのプロジェクトとアセットをインデックス化します。

Red Hat AMQ インスタンスは、Business Central のすべてのインスタンス間に、Java CDI メッセージを伝播します。たとえば、新規プロジェクトが作成された場合、アセットがインスタンスの1つでロックまたは変更された場合に、その情報が即座に他の全インスタンスで反映されます。

コントローラストラテジーは、クラスターデプロイメントには適していません。OpenShift デプロイメントの場合は、高可用性の Business Central は **OpenShift スタートアップストラテジー** を使用して KIE Server を管理する必要があります。

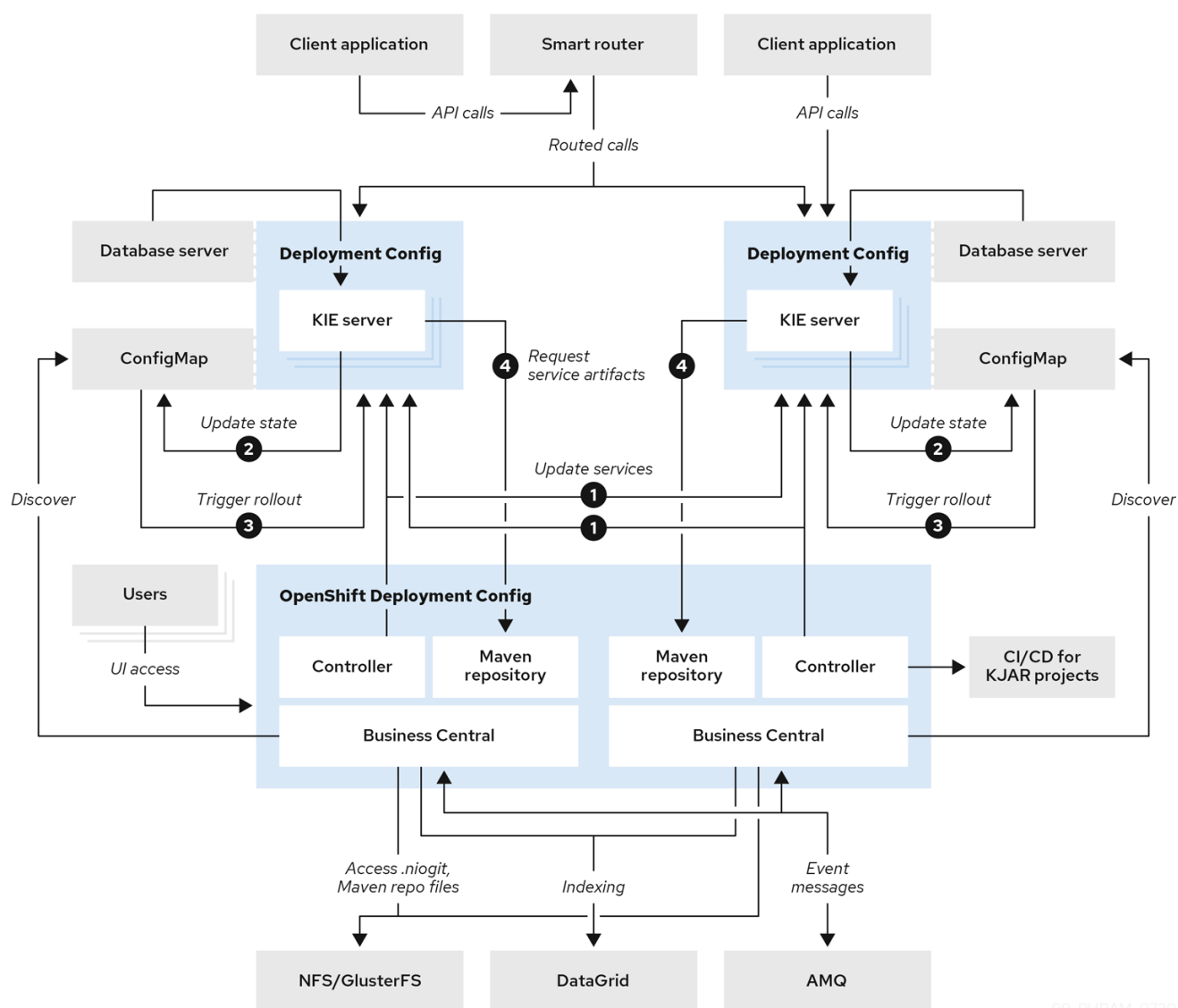
KIE Server デプロイメント (スケーリング可能) ごとに、現在の状態を反映する ConfigMap を作成します。Business Central は、ConfigMap を読み込むことで全 KIE Server を検出します。

ユーザーが KIE Server 設定 (例: サービスのデプロイまたはアンデプロイ) で変更を要求した場合に、Business Central は KIE Server への接続を開始し、REST API 要求を送信します。KIE Server は、全インスタンスが再デプロイされ、新規設定が反映されるように、ConfigMap を変更して新しい設定の状態を反映し、独自の再デプロイをトリガーします。

OpenShift 環境で、独立した KIE Server を複数デプロイできます。KIE Server にはそれぞれ、必要な設定が指定された個別の ConfigMap が設定されます。KIE Server は個別にスケーリングできます。

OpenShift デプロイメントに、Smart Router を追加できます。

図2.2 高可用性オーサリング環境のアーキテクチャ図



99_RHPAM_0720

第3章 OPENSIFT 環境に RED HAT PROCESS AUTOMATION MANAGER をデプロイする準備

OpenShift 環境に Red Hat Process Automation Manager をデプロイする前に、タスクをいくつか完了する必要があります。追加イメージ (たとえば、プロセスの新しいバージョン、または別のプロセス) をデプロイする場合は、このタスクを繰り返す必要はありません。

3.1. RED HAT レジストリーに対してお使いの環境が認証されていることを確認する方法

Red Hat OpenShift Container Platform の Red Hat Process Automation Manager コンポーネントをデプロイするには、OpenShift が Red Hat レジストリーから適切なイメージをダウンロードできることを確認します。

OpenShift は、お使いのサービスアカウントのユーザー名とパスワードを使用して Red Hat レジストリーへの認証が行われるように設定する必要があります。この設定は namespace ごとに固有であり、Operator が機能している場合は、**openshift** namespace に対する設定がすでに完了しています。

ただし、Red Hat Process Automation Manager のイメージストリームが **openshift** 名前空間にない場合や、Red Hat Process Automation Manager を新規バージョンに自動更新するように設定されている場合、Operator はこのプロジェクトの名前空間にイメージをダウンロードする必要があります。対象の名前空間の認証設定を完了する必要があります。

手順

1. **oc** コマンドで OpenShift にログインして、プロジェクトがアクティブであることを確認します。
2. [「Registry Service Accounts for Shared Environments」](#) で説明されている手順を実行します。Red Hat カスタマーポータルにログインして、このドキュメントにアクセスし、レジストリーサービスアカウントを作成する手順を実行します。
3. **OpenShift Secret** タブを選択し、**Download secret** のリンクをクリックして、YAML シークレットファイルをダウンロードします。
4. ダウンロードしたファイルを確認して、**name:** エントリーに記載の名前をメモします。
5. 以下のコマンドを実行します。

```
oc create -f <file_name>.yaml
oc secrets link default <secret_name> --for=pull
oc secrets link builder <secret_name> --for=pull
```

<file_name> はダウンロードしたファイルに、**<secret_name>** はファイルの **name:** のエントリーに記載されている名前に置き換えてください。

3.2. KIE SERVER のシークレットの作成

OpenShift は **シークレット** と呼ばれるオブジェクトを使用してパスワードやキーストアなどの機密情報を保持します。OpenShift のシークレットに関する詳細は、Red Hat OpenShift Container Platform ドキュメントの「[シークレット](#)」の章を参照してください。

KIE Server では HTTPS でアクセスできるように SSL 証明書を使用します。Business Central に SSL 証明書を作成し、OpenShift 環境にシークレットとして提供します。ただし、実稼働環境では、KIE

Server の SSL 証明書を作成し、これをシークレットとして OpenShift 環境に提供する必要があります。

手順

1. KIE Server の SSL 暗号化向けの秘密鍵と公開鍵で SSL キーストアを生成します。自己署名または購入した SSL 証明書でキーストアを作成する方法は、「[SSL 暗号化キーおよび証明書](#)」を参照してください。



注記

実稼働環境で、想定されている KIE Server の URL と一致する、有効な署名済み証明書を生成します。

2. キーストアを **keystore.jks** ファイルに保存します。
3. 証明書の名前をメモします。Red Hat Process Automation Manager 設定におけるこのデフォルト名は **jboss** です。
4. キーストアファイルのパスワードをメモします。Red Hat Process Automation Manager 設定におけるこのデフォルトの値は **mykeystorepass** です。
5. **oc** コマンドを使用して、新しいキーストアファイルからシークレット **kieserver-app-secret** を生成します。

```
$ oc create secret generic kieserver-app-secret --from-file=keystore.jks
```

3.3. BUSINESS CENTRAL へのシークレットの作成

HTTPS アクセスを提供するために、Business Central では SSL 証明書を使用します。Business Central に SSL 証明書を作成し、OpenShift 環境にシークレットとして提供します。ただし、実稼働環境では、Business Central の SSL 証明書を作成し、これをシークレットとして OpenShift 環境に提供する必要があります。

Business Central と KIE Server に同じ証明書およびキーストアを使用しないでください。

手順

1. Business Central の SSL 暗号化の秘密鍵および公開鍵を使用して、SSL キーストアを生成します。自己署名または購入した SSL 証明書でキーストアを作成する方法は、「[SSL 暗号化キーおよび証明書](#)」を参照してください。



注記

実稼働環境で、Business Central の予想される URL と一致する有効な署名済み証明書を生成します。

2. キーストアを **keystore.jks** ファイルに保存します。
3. 証明書の名前をメモします。Red Hat Process Automation Manager 設定におけるこのデフォルト名は **jboss** です。

4. キーストアファイルのパスワードをメモします。Red Hat Process Automation Manager 設定におけるこのデフォルトの値は **mykeystorepass** です。
5. **oc** コマンドを使用して、新しいキーストアファイルからシークレット **businesscentral-app-secret** を生成します。

```
$ oc create secret generic businesscentral-app-secret --from-file=keystore.jks
```

3.4. AMQ ブローカー接続のシークレットの作成

KIE Server を AMQ ブローカーに接続し、AMQ ブローカー接続に SSL を使用する場合は、接続の SSL 証明書を作成し、これを OpenShift 環境にシークレットとして指定する必要があります。

手順

1. AMQ ブローカー接続の SSL 暗号化の秘密鍵および公開鍵を使用して SSL キーストアを生成します。自己署名または購入した SSL 証明書でキーストアを作成する方法は、「[SSL 暗号化キーおよび証明書](#)」を参照してください。



注記

実稼働環境で、AMQ ブローカー接続の予想される URL に一致する有効な署名済みの証明書を作成します。

2. キーストアを **keystore.jks** ファイルに保存します。
3. 証明書の名前をメモします。Red Hat Process Automation Manager 設定におけるこのデフォルト名は **jboss** です。
4. キーストアファイルのパスワードをメモします。Red Hat Process Automation Manager 設定におけるこのデフォルトの値は **mykeystorepass** です。
5. **oc** コマンドを使用して、新しいキーストアファイルから **broker-app-secret** という名前のシークレットを生成します。

```
$ oc create secret generic broker-app-secret --from-file=keystore.jks
```

3.5. SMART ROUTER のシークレットの作成

HTTPS アクセスを提供するために、Smart Router では SSL 証明書を使用します。このデプロイメントでは、サンプルシークレットを自動的に作成できます。ただし、実稼働環境では、Smart Router の SSL 証明書を作成し、これをシークレットとして OpenShift 環境に提供する必要があります。

Smart Router の証明書およびキーストアに、KIE Server または Business Central で使用されているものと同じものを指定しないでください。

手順

1. Smart Router の SSL 暗号化の秘密鍵および公開鍵を使用して SSL キーストアを生成します。自己署名または購入した SSL 証明書でキーストアを作成する方法は、「[SSL 暗号化キーおよび証明書](#)」を参照してください。



注記

実稼働環境で、Smart Router の予想される URL と一致する有効な署名済み証明書を作成します。

2. キーストアを **keystore.jks** ファイルに保存します。
3. 証明書の名前をメモします。Red Hat Process Automation Manager 設定におけるこのデフォルト名は **jboss** です。
4. キーストアファイルのパスワードをメモします。Red Hat Process Automation Manager 設定におけるこのデフォルトの値は **mykeystorepass** です。
5. **oc** コマンドを使用して、新しいキーストアファイルからシークレット **smartrouter-app-secret** を生成します。

```
$ oc create secret generic smartrouter-app-secret --from-file=keystore.jks
```

3.6. 外部データベースのカスタム KIE SERVER 拡張イメージのビルド

KIE Server に外部データベースサーバーを使用し、そのデータベースサーバーが MySQL または PostgreSQL 以外の場合は、環境をデプロイする前にこのサーバー用のドライバーを使用するカスタムの KIE Server 拡張イメージをビルドする必要があります。

このビルド手順の手順を完了して、次のデータベースサーバーのいずれかにドライバーを提供します。

- Microsoft SQL Server
- IBM DB2
- Oracle データベース
- Sybase

必要に応じて、この手順を使用して、以下のデータベースサーバーのいずれかに対応する新しいバージョンのドライバーをビルドできます。

- MySQL
- MariaDB
- PostgreSQL

データベースサーバーのサポートされるバージョンについては、「[Red Hat Process Automation Manager 7 でサポートされる構成](#)」を参照してください。

ビルド手順では、既存の KIE Server イメージを拡張するカスタム拡張イメージを作成します。このカスタム拡張イメージは OpenShift 環境にインポートしてから、**EXTENSIONS_IMAGE** パラメーターで参照する必要があります。

前提条件

- **oc** コマンドを使用して OpenShift 環境にログインしている。OpenShift ユーザーには **registry-editor** ロールが必要です。

- Oracle Database、IBM DB2、または Sybase の場合は、データベースサーバーベンダーから JDBC ドライバーをダウンロードしている。
- 以下の必要なソフトウェアをインストールしている。
 - Docker: インストール手順は、「[Get Docker](#)」を参照してください。
 - Cekit バージョン 3.2: インストール手順は、「[Installation](#)」を参照してください。
 - Cekit の以下のライブラリーおよび拡張機能。詳細は、「[依存関係](#)」を参照してください。
 - **docker: python3-docker** パッケージまたは同様のパッケージで提供される。
 - **docker-squash: python3-docker-squash** または同様のパッケージで提供される。
 - **behave: python3-behave** パッケージまたは同様のパッケージで提供される。

手順

1. IBM DB2、Oracle Database、または Sybase の場合、JDBC ドライバー JAR ファイルをローカルディレクトリーに指定します。
2. Red Hat カスタマーポータルの [Software Downloads](#) ページから利用可能な **rhpmam-7.8.0-openshift-templates.zip** の製品配信可能ファイルをダウンロードします。
3. ファイルを展開し、コマンドラインを使用して解凍したファイルの **templates/contrib/jdbc/cekit** ディレクトリーに移動します。このディレクトリーには、カスタムビルドのソースコードが含まれます。
4. データベースサーバーのタイプに応じて、以下のコマンドのいずれかを入力します。

- Microsoft SQL Server の場合:

```
make mssql
```

- MySQL の場合:

```
make mysql
```

- PostgreSQL の場合:

```
make postgresql
```

- MariaDB の場合:

```
make mariadb
```

- IBM DB2 の場合:

```
make db2 artifact=/tmp/db2jcc4.jar version=10.2
```

このコマンドで、**/tmp/db2jcc4.jar** を IBM DB2 ドライバーのパス名に置き換え、**10.2** をドライバのバージョンに置き換えます。

- Oracle Database の場合:

```
make oracle artifact=/tmp/ojdbc7.jar version=7.0
```

このコマンドで、**/tmp/ojdbc7.jar** を Oracle Database ドライバーのパス名に置き換え、**7.0** をドライバーのバージョンに置き換えます。

- Sybase の場合:

```
make build sybase artifact=/tmp/jconn4-16.0_PL05.jar version=16.0_PL05
```

このコマンドで、**/tmp/jconn4-16.0_PL05.jar** をダウンロードされた Sybase ドライバーのパス名に置き換え、**16.0_PL05** をドライバーのバージョンに置き換えます。

または、Sybase ドライバーのドライバークラスまたはドライバーの XA クラスを更新する必要がある場合は、以下のコマンドに **DRIVER_CLASS** 変数または **DRIVER_XA_CLASS** 変数を設定してください。

```
export DRIVER_CLASS=another.class.Sybase && make sybase artifact=/tmp/jconn4-16.0_PL05.jar version=16.0_PL05
```

5. 以下のコマンドを入力して、ローカルで利用可能な Docker イメージを一覧表示します。

```
docker images
```

ビルドされたイメージの名前 (例: **jboss-kie-db2-extension-openshift-image**) およびイメージのバージョンタグ (**11.1.4.4** など (**latest** タグではない)) をメモします。

6. OpenShift 環境のレジストリーに直接アクセスし、イメージをレジストリーにプッシュします。ユーザーパーミッションに応じて、イメージを **openshift** 名前空間またはプロジェクト名前空間にプッシュできます。レジストリーへのアクセスおよびイメージのプッシュの手順については、Red Hat OpenShift Container Platform 製品ドキュメントの「[クラスターからレジストリーへの直接アクセス](#)」を参照してください。

3.7. GIT フックの準備

オーサリング環境では、Business Central のプロジェクトのソースコードが変更された場合に Git フックを使用してカスタムの操作を実行できます。Git フックは一般的に、アップストリームのリポジトリを操作する時に使用します。

Git フックが SSH 認証を使用してアップストリームのリポジトリを操作できるようにするには、リポジトリに、認証用の秘密鍵と既知のホストファイルも指定する必要があります。

Git フックを設定しない場合は、この手順を飛ばして次に進んでください。

手順

1. Git フックファイルを作成します。方法は、「[Git hooks reference documentation](#)」を参照してください。



注記

Business Central では **pre-commit** スクリプトはサポートされません。 **post-commit** スクリプトを使用してください。

2. 設定マップ (ConfigMap)、またはこれらのファイルを含む永続ボリュームを作成します。

- Git フックが1つまたは複数の固定スクリプトファイルで構成される場合は、**oc** コマンドを使用して設定アップを作成します。以下に例を示します。

```
oc create configmap git-hooks --from-file=post-commit=post-commit
```

- Git フックはロングファイルで構成されるか、実行可能ファイルや JAR ファイルなどのバイナリーに依存する場合は、永続ボリュームを使用します。永続ボリュームと永続ボリューム要求を作成し、ボリュームと要求を関連付けて、このファイルをボリュームに転送する必要があります。
永続ボリュームおよび永続ボリューム要求の説明は、[Red Hat OpenShift Container Platform ドキュメントの「ストレージ」](#)を参照してください。永続ボリュームへのファイルのコピー方法は、[「Transferring files in and out of containers」](#)を参照してください。

3. Git フックスクリプトが SSH 認証を使用してアップストリームのリポジトリと対話する必要がある場合は、必要なファイルでシークレットを作成します。

- リポジトリに格納されている公開鍵に一致する秘密鍵を使用して、**id_rsa** ファイルを作成します。
- リポジトリの正しい名前、アドレス、公開鍵で **known_hosts** ファイルを作成します。
- 以下のように **oc** コマンドを使用して、2つのファイルでシークレットを作成します。

```
oc create secret git-hooks-secret --from-file=id_rsa=id_rsa --from-file=known_hosts=known_hosts
```



注記

デプロイメントでこのシークレットを使用する場合は、**id_rsa** と **known_hosts** ファイルを、Business Central の Pod にある **/home/jboss/.ssh** ディレクトリにマウントします。

3.8. NFS を使用した READWRITEMANY アクセスモードの永続ボリュームのプロビジョニング

Business Central Monitoring または高可用性 Business Central をデプロイする場合、ご使用の環境は **ReadWriteMany** アクセスモードで永続ボリュームをプロビジョニングする必要があります。

お使いの設定で **ReadWriteMany** アクセスモードの永続ボリュームのプロビジョニングが必要であるものの、環境がそのようなプロビジョニングに対応しない場合は、NFS を使用してボリュームをプロビジョニングします。それ以外の場合、この手順は省略します。

手順

NFS サーバーをデプロイし、NFS を使用して永続ボリュームをプロビジョニングします。NFS を使用して永続ボリュームをプロビジョニングする方法については、『[OpenShift Container Platform 4.3 ストレージ](#)』の「[NFS を使用した永続ストレージ](#)」のセクションを参照してください。

3.9. S2I ビルドに使用する BUSINESS CENTRAL からのソースコードの展開

Source-to-Image (S2I) プロセスを使用してイミュータブル KIE Server を作成する予定がある場合は、

Git リポジトリにサービスのソースコードを提供する必要があります。オーサリングサービスに Business Central を使用する場合は、サービスのソースコードを展開して、S2I ビルドを使用する別の Git リポジトリ (GitHub や GitLab のオンプレミスインストールなど) に配置できます。

S2I プロセスを使用する予定がない場合や、サービスのオーサリングに Business Central を使用していない場合は、この手順を飛ばして次に進んでください。

手順

1. 以下のコマンドを使用してソースコードを展開します。

```
git clone https://<business-central-host>:443/git/<MySpace>/<MyProject>
```

このコマンドでは、以下の変数を置き換えてください。

- **<business-central-host>**: Business Central を実行しているホスト
- **<MySpace>**: プロジェクトが配置された Business Central 領域の名前
- **<MyProject>**: プロジェクトの名前



注記

Business Central でプロジェクトの完全な URL を表示するには、**Menu → Design → <MyProject> → Settings** の順にクリックします。



注記

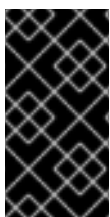
HTTPS 通信に自己署名証明書を使用している場合にこのコマンドを実行すると、エラーメッセージ **SSL certificate problem** が表示され失敗する可能性があります。このような場合は、**GIT_SSL_NO_VERIFY** 環境変数を使用するなど、**git** で SSL 証明書の検証を無効にします。

```
env GIT_SSL_NO_VERIFY=true git clone https://<business-central-host>:443/git/<MySpace>/<MyProject>
```

2. S2I ビルドの別の Git リポジトリ (GitHub または GitLab など) へのソースコードのアップロード

3.10. ネットワークが制限された環境でのデプロイメントの準備

公開インターネットに接続されていないネットワークが制限された環境に Red Hat Process Automation Manager をデプロイできます。ネットワークが制限された環境での Operator のデプロイメント方法は、Red Hat OpenShift Container Platform ドキュメントの「[ネットワークが制限された環境での Operator Lifecycle Manager の使用](#)」を参照してください。



重要

Red Hat Process Automation Manager 7.8 では、Operator インストーラーウィザードはテクノロジープレビュー機能となっています。Red Hat のテクノロジープレビュー機能のサポートの詳細は、「[テクノロジープレビュー機能のサポート範囲](#)」を参照してください。

公開インターネットへの送信アクセスが設定されていないデプロイメントを使用するには、必要なすべてのアーティファクトのミラーが含まれる Maven リポジトリを用意する必要があります。このリポジトリを作成する方法は、「[オフラインで使用する Maven ミラーリポジトリの用意](#)」を参照してください。

3.11. オフラインで使用する MAVEN ミラーリポジトリの用意

Red Hat OpenShift Container Platform 環境に公開インターネットへの送信アクセスが設定されていない場合には、必要なアーティファクトすべてのミラーが含まれる Maven リポジトリを用意して、このリポジトリを使用できるようにする必要があります。



注記

Red Hat OpenShift Container Platform 環境がインターネットに接続されている場合は、この手順を飛ばして次に進むことができます。

前提条件

- 公開インターネットへの送信アクセスが設定されているコンピューターが利用できる。

手順

- 書き込みアクセス権がある Maven リリースリポジトリを設定します。リポジトリは認証なしで読み取りアクセスを許可する必要があり、OpenShift 環境にはこのリポジトリへのネットワークアクセスが必要です。

OpenShift 環境に、Nexus リポジトリマネージャーをデプロイできます。OpenShift への Nexus の設定方法は、Red Hat OpenShift Container Platform 3.11 ドキュメントの「[Nexus の設定](#)」を参照してください。記載の手順は、OpenShift Container Platform バージョン 4 にも該当します。このリポジトリを別個のミラーリポジトリとして使用します。

または、サービスにカスタムの外部リポジトリ (Nexus など) を使用する場合、同じリポジトリをミラーリポジトリとして使用できます。

- 公開インターネットに送信アクセスができるコンピューターで、以下のアクションを実行します。
 - Red Hat Process Automation Manager 7.8.0 Offliner Content List**をクリックして、Red Hat カスタマーポータルの [Software Downloads](#) ページから **rhpmam-7.8.0-offliner.zip** の製品配信可能ファイルをダウンロードします。
 - rhpmam-7.8.0-offliner.zip** ファイルの内容を任意のディレクトリーに展開します。
 - ディレクトリーに移動し、以下のコマンドを入力します。

```
./offline-repo-builder.sh offliner.txt
```

このコマンドは、**repository** サブディレクトリーを作成し、必要なアーティファクトをこのサブディレクトリーにダウンロードします。

一部のダウンロードが失敗したことを示すメッセージが表示された場合は、同じコマンドを再度実行してください。ダウンロードが再び失敗する場合は、Red Hat サポートに連絡してください。

- repository** サブディレクトリーのすべてのアーティファクトを、作成した Maven ミラーリポジトリにアップロードします。アーティファクトをアップロードするには、Git リポジ

トリー [Maven repository tools](#) から利用できる Maven Repository Provisioner ユーティリティーを使用できます。

3. Business Central 外でサービスを開発し、追加の依存関係がある場合は、ミラーリポジトリにその依存関係を追加します。サービスを Maven プロジェクトとして開発した場合は、以下の手順を使用し、これらの依存関係を自動的に用意します。公開インターネットへに送信接続できるコンピューターで、この手順を実行します。
 - a. ローカルの Maven キャッシュディレクトリー (`~/.m2/repository`) のバックアップを作成して、ディレクトリーを削除します。
 - b. **mvn clean install** コマンドを使用してプロジェクトのソースをビルドします。
 - c. すべてのプロジェクトで以下のコマンドを入力し、Maven を使用してプロジェクトで生成したすべてのアーティファクトのランタイムの依存関係をすべてダウンロードするようにします。

```
mvn -e -DskipTests dependency:go-offline -f /path/to/project/pom.xml --batch-mode -Djava.net.preferIPv4Stack=true
```

`/path/to/project/pom.xml` は、プロジェクトの **pom.xml** ファイルへの正しいパスに置き換えます。

- d. ローカルの Maven キャッシュディレクトリー (`~/.m2/repository`) から作成した Maven ミラーリポジトリにすべてのアーティファクトをアップロードします。アーティファクトをアップロードするには、Git リポジトリ [Maven repository tools](#) から利用できる Maven Repository Provisioner ユーティリティーを使用できます。

第4章 OPENSIFT OPERATOR を使用した RED HAT PROCESS AUTOMATION MANAGER 環境のデプロイと管理

Red Hat Process Automation Manager 環境をデプロイするために、OpenShift Operator は環境を記述する YAML ソースを使用します。Red Hat Process Automation Manager は、YAML ソースを作成し、環境をデプロイするために使用できるインストーラーを提供します。

Business Automation Operator で環境をデプロイする場合は、環境の YAML 記述を作成し、環境が常にこの記述と一致していることを確認します。記述を編集して環境を変更することができます。

Red Hat OpenShift Container Platform で Operator アプリケーションを削除することで、環境を削除できます。



注記

高可用性の Business Central で環境を削除すると、Operator は、JBoss Datagrid および JBoss AMQ StatefulSet の生成時に作成された永続ボリューム要求 (PVC) を削除しません。この動作は、Kubernetes 設計の一部で、永続ボリューム要求を削除するとデータが損失される可能性があります。StatefulSet の削除時における永続ボリューム要求の取り扱いは、[Kubernetes documentation](#) を参照してください。

同じ namespace とアプリケーション名を使用する新規環境を構築すると、その環境ではパフォーマンスが向上されるように永続ボリュームを再利用します。

永続ボリューム要求は、手作業で削除できます。

4.1. BUSINESS AUTOMATION OPERATOR のサブスクリプション

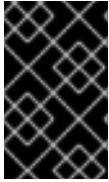
Operator を使用して Red Hat Process Automation Manager をデプロイできるようにするには、OpenShift で Business Automation Operator にサブスクリプションする必要があります。

手順

1. OpenShift Web クラスターコンソールでプロジェクトに移動します。
2. OpenShift Web コンソールのナビゲーションパネルで、**Catalog → OperatorHub** または **Operators → OperatorHub** を選択します。
3. **Business Automation** を検索し、これを選択してから **Install** をクリックします。
4. **Create Operator Subscription** ページで、ターゲットの名前空間および承認ストラテジーを選択します。
必要に応じて、**承認ストラテジー** を **Automatic** に設定して、Operator の自動更新を有効にします。Operator の更新は直ちに製品を更新しませんが、製品を更新する前に必要になります。特定のすべての製品デプロイメントの設定を使用して、自動または手動の製品更新を設定します。
5. **Subscribe** をクリックしてサブスクリプションを作成します。

4.2. OPERATOR を使用した RED HAT PROCESS AUTOMATION MANAGER 環境のデプロイ

Business Automation Operator にサブスクリプションした後に、インストーラーウィザードを使用して Red Hat Process Automation Manager 環境を設定し、デプロイできます。



重要

Red Hat Process Automation Manager 7.8 では、Operator インストーラーウィザードはテクノロジープレビュー機能となっています。Red Hat のテクノロジープレビュー機能の詳細は、「[テクノロジープレビュー機能のサポート範囲](#)」を参照してください。

4.2.1. Business Automation Operator の使用による Red Hat Process Automation Manager 環境のデプロイメントの開始

Business Automation Operator を使用して Red Hat Process Automation Manager 環境のデプロイメントを開始するには、インストーラーウィザードにアクセスします。インストーラーウィザードは Operator にサブスクライブするとデプロイされます。

前提条件

- Business Automation Operator にサブスクライブしている。Operator にサブスクライブする方法は、「[Business Automation Operator のサブスクライブ](#)」を参照してください。

手順

1. Red Hat OpenShift Container Platform Web クラスターコンソールメニューで、**Catalog → Installed operators** または **Operators → Installed operators** を選択します。
2. **businessautomation** が含まれる Operator の名前をクリックします。この Operator の情報が表示されます。
3. ウィンドウの右側にある **Installer** リンクをクリックします。
4. プロンプトが出されたら、OpenShift 認証情報でログインします。

結果

ウィザードの **Installation** タブが表示されます。

4.2.2. 環境の基本設定の構成

Business Automation Operator を使用して Red Hat Process Automation Manager 環境のデプロイを開始した後に、環境のタイプを選択し、他の基本的な設定を行う必要があります。

前提条件

- 「[Business Automation Operator の使用による Red Hat Process Automation Manager 環境のデプロイメントの開始](#)」の説明に従って、Business Automation Operator を使用して Red Hat Process Automation Manager 環境のデプロイを開始し、インストーラーウィザードにアクセスしていること。

手順

1. **Application Name** フィールドに、OpenShift アプリケーションの名前を入力します。この名前は、すべてのコンポーネントのデフォルト URL で使用されます。
2. **Environment** 一覧で、環境のタイプを選択します。このタイプは、デフォルトの設定を定めるものです。この設定を必要に応じて変更することができます。以下のタイプは Red Hat Process Automation Manager で利用できます。

- **rhpm-trial:** すばやく設定でき、アセットの開発や実行の評価やデモに使用できる試用版の環境。Business Central と KIE Server 1 台が含まれています。この環境では永続ストレージを使用しないため、この環境で実行した作業内容は保存されません。
- **rhpm-authoring:** Business Central を使用してサービスを作成および修正する環境。これは、オーサリング作業用に Business Central を提供する Pod およびサービスのテスト実行用に KIE Server を提供する Pod で構成されます。
- **rhpm-authoring-ha:** Business Central を使用してサービスを作成し、変更する環境。これは、オーサリング作業用に Business Central を提供する Pod およびサービスのテスト実行用に KIE Server を提供する Pod で構成されます。このバージョンのオーサリング環境は、高可用性が確保されるように Business Central Pod のスケーリングをサポートします。



重要

Red Hat Process Automation Manager 7.8 では、Operator を使用した高可用性 Business Central 機能のデプロイメントはテクノロジープレビューとしてのみご利用いただけます。Red Hat Technology Preview 機能の詳細は、「[テクノロジープレビュー機能のサポート範囲](#)」を参照してください。完全にサポートされた高可用性デプロイメントの場合、Red Hat OpenShift Container Platform バージョン 3.11 で高可用性オーサリングテンプレートを使用します。このテンプレートをデプロイする手順については、『Red Hat [OpenShift Container Platform への Red Hat Process Automation Manager オーサリング環境のデプロイ](#)』を参照してください。

- **rhpm-production:** ステージングおよび実稼働用として既存のサービスを実行するのに使用する環境。この環境には、Business Central Monitoring、Smart Router、および KIE Server Pod の 2 つのグループが含まれます。このようなすべてのグループに対してサービスのデプロイおよびデプロイ解除を実行できます。Business Central Monitoring を使用してサービスをデプロイし、実行し、停止し、またそれらの実行を監視します。
- **rhpm-production-immutable:** ステージングおよび実稼働の目的で既存のサービスを実行するための別の環境。ソースからサービスをビルドしたり、Maven リポジトリからサービスをプルする KIE Server Pod を 1 つ以上設定できます。その後、必要に応じて各 Pod を複製できます。
Pod からサービスを削除したり、新しいサービスを Pod に追加したりすることはできません。サービスの別のバージョンを使用するか、他の方法で設定を変更する場合は、新規のサーバーイメージをデプロイして、以前のイメージを置き換えます。コンテナベースの統合ワークフローを使用して、Pod を管理できます。

この環境を設定する場合は、**KIE Servers** タブで KIE Server をカスタマイズし、**Set immutable server configuration** ボタンをクリックするか、**KIE_SERVER_CONTAINER_DEPLOYMENT** 環境変数を設定します。KIE Server の設定手順は、「[環境のカスタム KIE Server 設定の構成](#)」を参照してください。

必要に応じて、**Console** タブを使用して、この環境に Business Central Monitoring を追加して、プロセスサービスの実行を監視、停止、および再起動することもできます。Business Central Monitoring の設定手順は、「[環境の Business Central 設定の構成](#)」を参照してください。

3. 新しいバージョンへの自動アップグレードを有効にするには、**Enable Upgrades** ボックスを選択します。このボックスを選択すると、Red Hat Process Automation Manager 7.8 の新しいパッチバージョンが利用可能になると、Operator は自動的にこのバージョンにデプロイメントをアップグレードします。サービスはすべて確保され、アップグレードプロセス全体で通常通り利用できます。

同じ自動アップグレードプロセスを、Red Hat Process Automation Manager 7.x の新規マイナーバージョンが利用できる場合にも有効にする場合は、**Include minor version upgrade** のチェックボックスを選択します。



注記

Red Hat Process Automation Manager のコンポーネントにカスタムイメージを使用する場合は、自動更新を無効にします。

4. **Custom registry** のカスタムイメージレジストリーを使用する場合は、**Image registry** フィールドにレジストリーの URL を入力します。このレジストリーに適切に署名され、認識された SSL 証明書がない場合は、**Insecure** ボックスを選択します。
5. **Admin user** の **Username** フィールドおよび **Password** フィールドに、Red Hat Process Automation Manager の管理者ユーザーのユーザー名およびパスワードを入力します。



重要

RH-SSO または LDAP 認証を使用する場合は、同じユーザーを、Red Hat Process Automation Manager の **kie-server,rest-all,admin** ロールで認証システムに設定する必要があります。

6. イメージのカスタムバージョンタグを使用する必要がある場合は、以下の手順を実行します。
 - a. **Next** をクリックして **Security** タブにアクセスします。
 - b. ウィンドウの下部までスクロールします。
 - c. イメージタグを **Image tag** フィールドに入力します。

次のステップ

デフォルト設定で環境をデプロイする必要がある場合は、**Finish** をクリックしてから **Deploy** をクリックして環境をデプロイします。それ以外の場合は、引き続き他の設定パラメーターの設定を行います。

4.2.3. 環境のセキュリティー設定の構成

Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本的な設定を行ったら、必要に応じて環境の認証 (セキュリティー) を設定できます。

前提条件

- 「[環境の基本設定の構成](#)」の説明に従って、インストーラーウィザードで Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本設定を行っている。
- 認証に RH-SSO または LDAP を使用する必要がある場合には、認証システムに適切なロールを持つユーザーを作成していること。**kie-server,rest-all,admin** ロールを持つ少なくとも1人の管理ユーザー (たとえば、**adminUser**) を作成する必要があります。このユーザーには、**Installation** タブで設定したユーザー名とパスワードが必要です。
- RH-SSO 認証を使用する必要がある場合は、環境のすべてのコンポーネントの RH-SSO システムでクライアントを作成しており、正しい URL を指定している。この動作により、最大限の制御が確保されます。他の方法として、デプロイメントでクライアントを作成できます。

手順

1. **Installation** タブが開いている場合は、**Next** をクリックして **Security** タブを表示します。
2. **Authentication mode** 一覧で、以下のモードのいずれかを選択します。
 - **Internal**: 環境のデプロイ時に初期ユーザーを設定します。このユーザーは Business Central を使用して他のユーザーを随時セットアップできます。
 - **RH-SSO**: Red Hat Process Automation Manager は認証に Red Hat Single Sign-On を使用します。
 - **LDAP**: Red Hat Process Automation Manager は認証に LDAP を使用します。
3. 選択した **Authentication mode** に基づいてセキュリティー設定を完了します。
RH-SSO を選択している場合は、RH-SSO 認証を設定します。

- a. **RH-SSO URL** フィールドに、RH-SSO URL を入力します。
- b. **Realm** フィールドに、RH-SSO レalm 名を入力します。
- c. 環境のコンポーネントに RH-SSO クライアントを作成していない場合は、**SSO admin user** フィールドおよび **SSO admin password** フィールドに、RH-SSO システムの管理者ユーザーの認証情報を入力します。
- d. RH-SSO システムに適切な署名済みの SSL 証明書がない場合は、**Disable SSL cert validation** ボックスを選択します。
- e. **Principal attribute** フィールドで、ユーザー名に使用される RH-SSO プリンシパル属性を変更する必要がある場合は、新規属性の名前を入力します。

LDAP を選択した場合は、LDAP 認証を設定します。

- a. **LDAP URL** フィールドに、LDAP URL を入力します。
 - b. Red Hat JBoss EAP の LdapExtended ログインモジュールの設定に対応する LDAP パラメーターを設定します。これらの設定に関する説明は、「[LdapExtended ログインモジュール](#)」を参照してください。
4. **RH-SSO** または **LDAP** を選択した場合、RH-SSO または LDAP システムがデプロイメントに必要なすべてのロールを定義していない場合、認証システムのロールを Red Hat Process Automation Manager のロールにマップできます。
ロールマッピングを有効にするには、プロジェクト namespace の OpenShift 設定マップまたはシークレットオブジェクトにロールマッピング設定ファイルを指定する必要があります。
ファイルには、次の形式のエントリーが含まれている必要があります。

```
ldap_role = product_role1, product_role2...
```

以下に例を示します。

```
admins = kie-server,rest-all,admin
```

このファイルの使用を有効にするには、以下の変更を行います。

- a. **Roles properties file** フィールドの **RoleMapper** の下に、ルールマッピング設定ファイルの完全修飾パス名を入力します (例:
`/opt/eap/standalone/configuration/rolemapping/rolemapping.properties`)。

- b. 認証システムで定義されているロールをマッピングファイルで定義されているロールに置き換える場合は、**Replace roles** ボックスを選択します。それ以外の場合は、RH-SSO または LDAP で定義されたロールと設定ファイルで定義されたロールの両方が利用可能です。
 - c. **RoleMapper Configuration object** の下のフィールドで、ファイルを提供するオブジェクトの **Kind (ConfigMap または Secret)** を選択し、オブジェクトの **Name** を入力します。このオブジェクトは、ロールマッピング設定ファイルに指定したパスで Business Central および KIE Server Pod に自動的にマウントされます。
5. 他のパスワードを設定します (必要な場合)。
- **AMQ password** および **AMQ cluster password** は、JMS API を使用した ActiveMQ との対話に使用するパスワードです。
 - **Keystore password** は、HTTPS 通信のシークレットで使用されるキーストアファイルのパスワードです。「[KIE Server のシークレットの作成](#)」または「[Business Central へのシークレットの作成](#)」の説明にしたがってシークレットを作成した場合は、このパスワードを設定します。
 - **Database password** は、環境の一部であるデータベースサーバー Pod のパスワードです。

次のステップ

すべてのコンポーネントのデフォルト設定で環境をデプロイする必要がある場合は、**Finish** をクリックしてから **Deploy** をクリックして環境をデプロイします。それ以外の場合は、引き続き Business Central、KIE Server、および Smart Router の設定パラメーターを設定します。

4.2.4. 環境の Business Central 設定の構成

Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本的なセキュリティ設定を行った後に、必要に応じて環境の Business Central または Business Central Monitoring コンポーネントを設定できます。

前提条件

- 「[環境の基本設定の構成](#)」の説明に従って、インストーラーウィザードで Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本設定を行っている。
- 認証に RH-SSO または LDAP を使用する必要がある場合は、「[環境のセキュリティ設定の構成](#)」の説明に従ってセキュリティ設定を完了している。

手順

1. **Installation** または **Security** タブが開いている場合は、**Console** タブが表示されるまで **Next** をクリックします。
2. 「[Business Central へのシークレットの作成](#)」の説明に従って Business Central のシークレットを作成している場合、**Secret** フィールドにシークレットの名前を入力します。
3. Git フックを設定します (任意)。
オーサリング環境では、Git フックを使用して、Business Central の内部 Git リポジトリと外部 Git リポジトリ間の操作を容易化できます。Git フックを使用する場合は、プロジェクト namespace の OpenShift 設定マップ、シークレット、または Persistent Volume Claim (PVC:

永続ボリューム要求) オブジェクトに Git フックディレクトリーを準備する必要があります。Git の SSH 認証用の SSH キーと既知のホストファイルでシークレットを作成することもできます。Git フックの作成に関する詳細は、「[Git フックの準備](#)」を参照してください。

Git フックディレクトリーを使用するには、以下の変更を加えます。

- a. **Mount path** フィールドの **GitHooks** の下に、ディレクトリーの完全修飾名を入力します (例: `/opt/kie/data/git/hooks`)。
 - b. **GitHooks Configuration object** の下のフィールドで、ファイルを提供するオブジェクトの **Kind** (**ConfigMap**、**Secret**、または **PersistentVolumeClaim**) を選択し、オブジェクトの **Name** を入力します。このオブジェクトは、Git フックディレクトリーの指定したパスで Business Central Pod に自動的にマウントされます。
 - c. 必要に応じて、**SSH secret** フィールドに、SSH キーと既知のホストファイルを含む、シークレットを入力します。
4. 必要に応じて、Business Central または Business Central monitoring のレプリカ数を **Replicas** フィールドに入力します。この数は **rhcam-authoring** 環境では変更しません。
 5. 必要に応じて、**Resource quotas** 下のフィールドに必要な CPU およびメモリーの上限值を入力します。
 6. Business Central Pod の Java 仮想マシンの設定をカスタマイズする必要がある場合は、**Enable JVM configuration** ボックスを選択してから、**Enable JVM configuration** の下のフィールドに情報を入力します。すべてのフィールドは任意です。設定可能な JVM パラメーターは、「[JVM 設定パラメーター](#)」を参照してください。
 7. RH-SSO 認証を選択している場合は、Business Central の RH-SSO を設定します。
 - a. **Client name** フィールドにクライアント名を入力し、**Client secret** フィールドにクライアントシークレットを入力します。この名前を持つクライアントが存在しない場合は、デプロイメントでこの名前およびシークレットを持つ新規クライアントの作成を試行します。
 - b. デプロイメントで新規クライアントを作成する場合は、KIE Server へのアクセスに使用する HTTP および HTTPS URL を **SSO HTTP URL** フィールドおよび **SSO HTTPS URL** フィールドに入力します。この情報は、クライアントに記録されます。
 8. 必要に応じて、環境変数を随時設定します。環境変数を設定するには、**Add new Environment variable** をクリックしてから、変数の名前および値を **Name** フィールドおよび **Value** フィールドに入力します。
 - **rhcam-production** または **rhcam-production-immutable** 環境で、ファイルシステムを使用しない簡易モードで Business Central Monitoring を実行する場合には、**ORG_APPFORMER_SIMPLIFIED_MONITORING_ENABLED** を **true** に設定します。簡易モードの場合には、Business Central Monitoring では永続ボリューム要求が必要ではありません。永続ストレージへの **ReadWriteMany** アクセスをサポートしない環境で、このモードを使用できます。カスタムダッシュボードの設計には、簡易モードで Business Central Monitoring を使用できません。
 - 外部 Maven リポジトリーを使用する必要がある場合は、以下の変数を設定します。
 - **MAVEN_REPO_URL**: Maven リポジトリーの URL
 - **MAVEN_REPO_ID**: Maven リポジトリーの ID (例: **repo-custom**)
 - **MAVEN_REPO_USERNAME**: Maven リポジトリーのユーザー名

- **MAVEN_REPO_PASSWORD** Maven リポジトリのパスワード



重要

オーサリング環境で、Business Central を使用して外部の Maven リポジトリにプロジェクトをプッシュする場合は、デプロイメント時にこのリポジトリを設定して、全プロジェクトのリポジトリへのエクスポートを設定する必要があります。外部の Maven リポジトリへの Business Central プロジェクトのエクスポートに関する情報は、『[Red Hat Process Automation Manager プロジェクトのパッケージ化およびデプロイ](#)』を参照してください。

- OpenShift 環境が公開インターネットに接続されていない場合は、『[オフラインで使用する Maven ミラーリポジトリの用意](#)』に従って設定した Maven ミラーにアクセスできるように設定します。以下の変数を設定してください。
 - **MAVEN_MIRROR_URL**: 『[オフラインで使用する Maven ミラーリポジトリの用意](#)』でセットアップした Maven ミラーリポジトリの URL。この URL は、OpenShift 環境の Pod からアクセスできるようにする必要があります。
 - **MAVEN_MIRROR_OF**: ミラーから取得されるアーティファクトを定める値。**mirrorOf** 値の設定方法は、Apache Maven ドキュメントの『[Mirror Settings](#)』を参照してください。デフォルト値は **external:*** です。この値の場合、Maven はミラーから必要なアーティファクトをすべて取得し、他のリポジトリにクエリーを送信しません。外部の Maven リポジトリ (**MAVEN_REPO_URL**) を設定する場合は、ミラーからこのリポジトリ内のアーティファクトを除外するように **MAVEN_MIRROR_OF** を変更します (例: **external:*,!repo-custom**)。repo-custom は、**MAVEN_REPO_ID** で設定した ID に置き換えます。

オーサリング環境でビルトイン Business Central Maven リポジトリを使用する場合、ミラーからこのリポジトリのアーティファクトを除外するように **MAVEN_MIRROR_OF** を変更します (例: **external:*,!repo-rhpbamcentr**)。

- 一部のオーサリング環境では、複数のユーザーが同じ KIE Server に同時にサービスをデプロイできることを確認する必要があります。デフォルトでは、ユーザーは、Business Central を使用して KIE Server にサービスをデプロイして数秒待ってから追加サービスをデプロイする必要があります。**OpenShiftStartupStrategy** 設定はデフォルトで有効になり、この制限が発生します。制限を削除するには、**コントローラストラテジー** を使用するように **rhpbam-authoring** 環境を設定します。これについての特定のニーズがない限り、この変更を行わないでください。コントローラストラテジーを有効にする場合は、Business Central と同じ環境内のすべての KIE Server でこの変更を行ってください。



注記

高可用性の Business Central を使用する環境でコントローラストラテジーを有効にしないでください。この環境では、コントローラストラテジーは正しく機能しません。

Business Central でコントローラストラテジーを有効にするには、**KIE_SERVER_CONTROLLER_OPENSIFT_ENABLED** 環境変数を **false** に設定します。

次のステップ

Smart Router やプロセスインスタンス移行 (PIM) を使用せずに、KIE Server のデフォルト設定で環境をデプロイする場合は、**Finish** をクリックしてから **Deploy** をクリックし、この環境をデプロイします。それ以外の場合は、引き続き KIE Server および Smart Router の設定パラメーターを設定します。

4.2.5. 環境のカスタム KIE Server 設定の構成

Business Automation Operator のすべての環境タイプには、デフォルトで1つまたは複数の KIE Server が含まれます。

必要に応じて、KIE Server のカスタム設定を構成できます。この場合、デフォルトの KIE Server は作成されず、設定する KIE Server のみがデプロイされます。

前提条件

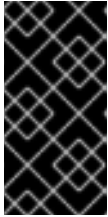
- 「[環境の基本設定の構成](#)」の説明に従って、インストーラーウィザードで Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本設定を行っている。
- 認証に RH-SSO または LDAP を使用する必要がある場合は、「[環境のセキュリティ設定の構成](#)」の説明に従ってセキュリティ設定を完了している。

手順

1. **Installation**、**Security**、または **Console** タブが開いている場合は、**KIE Servers** タブが表示されるまで **Next** をクリックします。
2. **Add new KIE Server** をクリックして、新規の KIE Server 設定を追加します。
3. **Id** フィールドに、KIE Server の ID を入力します。KIE Server が Business Central または Business Central Monitoring インスタンスに接続される場合、この ID はサーバーが加わるサーバーグループを決めるものとなります。
4. **Name** フィールドに、KIE Server の名前を入力します。
5. **Deployments** フィールドに、デプロイする同様の KIE Server の数を入力します。インストーラーは、同じ設定で複数の KIE Server をデプロイできます。KIE Server の ID および名前は自動的に変更され、一意な状態に保たれます。
6. 「[KIE Server のシークレットの作成](#)」の説明に従って KIE Server のシークレットを作成している場合、**Keystore secret** フィールドにシークレットの名前を入力します。
7. 必要に応じて、KIE Server のレプリカ数を **Replicas** フィールドに入力します。
8. KIE Server のカスタムイメージを使用する必要がある場合は、以下の追加の手順を実行します。
 - a. **Set KIE Server image** をクリックします。
 - b. OpenShift イメージストリームタグではなく Docker イメージ名を使用する必要がある場合は、**Kind** の値を **DockerImage** に変更します。
 - c. イメージストリームの名前を **Name** フィールドに入力します。
 - d. イメージストリームが **openshift** 名前空間にない場合は、名前空間を **Namespace** フィールドに入力します。

カスタムイメージの作成手順については、「[KIE Server のカスタムイメージの作成](#)」を参照してください。

9. Source to Image (S2I) ビルドを使用して、イミュータブル KIE Server を設定する必要がある場合は、以下の追加の手順を実行します。



重要

Maven リポジトリからサービスをプルするイミュータブル KIE Server を設定する必要がある場合は、**Set Immutable server configuration** をクリックせず、この手順も実行しないでください。代わりに、**KIE_SERVER_CONTAINER_REPLOYMENT** 環境変数を設定します。

- a. **Set Immutable server configuration** をクリックします。
- b. **KIE Server コンテナデプロイメント** フィールドに、デプロイメントが Source to Image (S2I) ビルドの結果から展開する必要があるサービスの識別情報 (KJAR ファイル) を入力します。形式は `<containerId>=<groupId>:<artifactId>:<version>` になります。また、コンテナのエイリアス名で指定する場合には、形式は `<containerId>(<aliasId>)=<groupId>:<artifactId>:<version>` になります。以下の例に示されるように、区切り文字 | を使用して 2 つ以上の KJAR ファイルを指定できます (例:
`containerId=groupId:artifactId:version|c2(alias2)=g2:a2:v2`)。
- c. OpenShift 環境に公開インターネットへの接続がない場合は、「[オフラインで使用する Maven ミラーリポジトリの用意](#)」に従って、Maven mirror URL フィールドに設定した **Maven ミラーの URL** を入力します。
- d. **Artifact directory** フィールドで、Maven が正常にビルドされた後に、必要なバイナリーファイル (KJAR ファイルおよびその他の必要なファイル) が含まれるプロジェクト内のパスを入力します。通常、このディレクトリーはビルドのターゲットディレクトリーです。ただし、Git リポジトリのこのディレクトリーにビルド済みのバイナリーを提供できます。
- e. S2I ビルドにカスタムベース KIE Server イメージを使用する必要がある場合は、**Set Base build image** をクリックして、**Name** フィールドにイメージストリームの名前を入力します。イメージストリームが **openshift** 名前空間にない場合は、名前空間を **Namespace** フィールドに入力します。OpenShift イメージストリームタグではなく Docker イメージ名を使用する必要がある場合は、**Kind** の値を **DockerImage** に変更します。
- f. **Set Git source** をクリックし、以下のフィールドに情報を入力します。
 - **S2I Git URI**: サービスのソースが含まれる Git リポジトリの URI。
 - **Reference**: Git リポジトリのブランチ。
 - **コンテキストディレクトリー**: (オプション) Git リポジトリからダウンロードされたプロジェクト内のソースへのパス。デフォルトで、ダウンロードされたプロジェクトのルートディレクトリーはソースディレクトリーです。
- g. **Git Webhook** を設定して Git リポジトリの変更が KIE Server の自動リビルドをトリガーするように設定するには、**Add new Webhook** をクリックします。**Type** リストから Webhook のタイプを選択し、**Secret** フィールドに Webhook のシークレット文字列を入力します。

10. 必要に応じて、**Resource quotas** 下のフィールドに必要な CPU およびメモリーの上限值を入力します。複数の KIE Server を設定している場合は、制限値はそれぞれのサーバーに別々に適用されます。
11. RH-SSO 認証を選択している場合は、KIE Server の RH-SSO を設定します。
 - a. **Client name** フィールドにクライアント名を入力し、**Client secret** フィールドにクライアントシークレットを入力します。この名前を持つクライアントが存在しない場合は、デプロイメントでこの名前およびシークレットを持つ新規クライアントの作成を試行します。
 - b. デプロイメントで新規クライアントを作成する場合は、KIE Server へのアクセスに使用する HTTP および HTTPS URL を **SSO HTTP URL** フィールドおよび **SSO HTTPS URL** フィールドに入力します。この情報は、クライアントに記録されます。
12. 外部 AMQ メッセージブローカーを使用して JMS API から KIE Server と対話する場合は、**Enable JMS Integration** 設定を有効にします。JMS 統合を設定するための追加のフィールドが表示され、必要に応じて値を入力する必要があります。
 - **User name、Password:** ブローカーのユーザー認証が環境で必要な場合の、標準ブローカーユーザーのユーザー名およびパスワード。
 - **Executor :** この設定を選択して JMS Executor を無効にします。Executor はデフォルトで有効になります。
 - **Executor transacted:** この設定を選択して、Executor キューで JMS トランザクションを有効にします。
 - **Enable signal:** この設定を選択して JMS 経由でシグナルの設定を有効にします。
 - **Enable audit** この設定を選択して JMS 経由で監査ロギングを有効にします。
 - **Audit transacted:** この設定を選択して、監査キューで JMS トランザクションを有効にします。
 - **Queue executor、Queue request、Queue response、Queue signal、Queue audit** 使用するキューのカスタム JNDI 名。これらの値のいずれかを設定する場合は、**AMQ キューパラメーター**も設定する必要があります。
 - **AMQ Queues:** AMQ キュー名はコマンドで区切られます。これらのキューはブローカーの起動時に自動的に作成され、JBoss EAP サーバーの JNDI リソースとしてアクセスできます。カスタムキュー名を使用する場合は、このフィールドでサーバーが使用するすべてのキューの名前を入力する必要があります。
 - **Enable SSL integration** AMQ ブローカーへの SSL 接続を使用する場合は、この設定を選択します。この場合は、「[AMQ ブローカー接続のシークレットの作成](#)」で作成したシークレットの名前や、シークレットに使用したキーストアおよび信頼ストアの名前およびパスワードも指定する必要があります。
13. KIE Server Pod の Java 仮想マシンの設定をカスタマイズする必要がある場合は、**Enable JVM configuration** ボックスを選択してから、**Enable JVM configuration** の下のフィールドに情報を入力します。すべてのフィールドは任意です。設定可能な JVM パラメーターは、「[JVM 設定パラメーター](#)」を参照してください。
14. **Database type** フィールドで、KIE Server が使用する必要のあるデータベースを選択します。以下の値を使用できます。
 - **mysql:** 個別の Pod に作成される MySQL サーバー。

- **postgresql**: 個別の Pod に作成される PostgreSQL サーバー。他の設定を使用する特別な理由のない限り、この設定を使用します。
 - **h2**: 個別の Pod を必要としないビルトインされた **h2** データベースエンジン。この設定を使用する場合は、KIE Server Pod をスケーリングしないでください。
 - **external**: 外部データベースサーバー。
15. **external** 以外のデータベースを選択している場合は、データベースを保存するために Persistent Volume Claim (PVC: 永続ボリューム要求) が作成されます。必要に応じて、永続ボリュームの設定パラメーターを指定します。
- また、**Size** フィールドに、永続ボリュームのサイズを入力します。
 - **StorageClass name** フィールドで、永続ボリュームのストレージクラス名を入力します。
16. **external** データベースを選択した場合は、必要に応じて、KIE Server 拡張イメージを設定します。PostgreSQL、MySQL、MariaDB 以外のデータベースサーバーを使用する場合には、「[外部データベースのカスタム KIE Server 拡張イメージのビルド](#)」の説明に従い、KIE Server 拡張イメージに、データベースサーバーのドライバーを指定する必要があります。KIE Server をこの拡張イメージを使用するように設定するには、以下の変更を加えます。
- a. **Enable extension image stream** ボックスを選択します。
 - b. **Extension image stream tag** フィールドに、作成したイメージの ImageStreamTag 定義を入力します (例: **jboss-kie-db2-extension-openshift-image:11.1.4.4**)。
 - c. 必要に応じて、**Extension image stream namespace** フィールドに、イメージをプッシュした名前空間を入力します。このフィールドに値を入力しない場合、Operator はイメージが **openshift** 名前空間にあると予想します。
 - d. オプションで、**Extension image install directory** フィールドに、各種の拡張機能が置かれている拡張イメージ内のディレクトリーを入力します。「[外部データベースのカスタム KIE Server 拡張イメージのビルド](#)」の手順を使用してイメージをビルドした場合は、このフィールドに値を入力しないでください。
17. 外部データベースサーバーを選択している場合は、追加のフィールドに以下の情報を入力します。
- a. **Driver**: サーバーの種類に応じてデータベースサーバードライバーを入力します。
 - **mysql**
 - **postgresql**
 - **mariadb**
 - **mssql**
 - **db2**
 - **oracle**
 - **sybase**
 - b. **Dialect**: サーバーの種類に応じて、サーバーの Hibernate 方言を入力します。共通の設定は以下のとおりです。

- `org.hibernate.dialect.MySQL5InnoDBDialect`
- `org.hibernate.dialect.MySQL8Dialect`
- `org.hibernate.dialect.MariaDB102Dialect`
- `org.hibernate.dialect.PostgreSQL95Dialect`
- `org.hibernate.dialect.PostgresPlusDialect` (EnterpriseDB Postgres Advanced Server で使用される)
- `org.hibernate.dialect.SQLServer2012Dialect` (MS SQL で使用される)
- `org.hibernate.dialect.DB2Dialect`
- `org.hibernate.dialect.Oracle10gDialect`
- `org.hibernate.dialect.SybaseASE15Dialect`
サポートされる方言の完全リストは、Red Hat JBoss EAP ドキュメントの「[Hibernate properties](#)」の表 A.7 を参照してください。

- c. **Host:** 外部データベースサーバーのホスト名を入力します。
- d. **Port:** 外部データベースサーバーのポート番号を入力します。
- e. **Jdbc URL:** 外部データベースサーバーの JDBC URL を入力します。



注記

EnterpriseDB Postgres データベースサーバーを使用している場合は、**jdbc:postgresql://** で始まる URL を使用し、**jdbc:edb://** は使用しないでください。または、URL を設定せず、代わりにホストとポートのパラメーターを設定します。

- f. **NonXA:** データソースを XA 以外のモードで設定する必要がある場合にこのボックスを選択します。
 - g. **JNDI name:** アプリケーションがデータソースに使用する JNDI 名を入力します。
 - h. **User name** および **Password:** 外部データベースサーバーのユーザー名およびパスワードを入力します。
 - i. **Background validation:** 必要に応じて、このボックスを選択してバックグラウンド SQL 検証を有効にし、バックグラウンド検証の間隔を入力します。
 - j. 必要に応じて、最小および最大の接続プールサイズ、有効な接続チェッカークラス、およびデータベースサーバーの例外ソータークラスを設定します。
18. MySQL バージョン 8 の外部データベースサーバーを使用する場合は、**mysql_native_password** プラグインを有効にして、認証に使用してください。このプラグインに関する詳細は、MySQL 8.0 Reference Manual の [Native Pluggable Authentication](#) を参照してください。
- Red Hat on Red Hat OpenShift Container Platform が提供する MySQL バージョン 8 のイメージを使用してプラグインを有効にするには、**MYSQL_DEFAULT_AUTHENTICATION_PLUGIN** 環境変数を **mysql_native_password** に設定してください。

MySQL バージョン 8 サーバーでユーザーを作成してから **mysql_native_password** プラグインを有効にした場合には、プラグインを有効にしてから **mysql-user** テーブルを更新する必要があります。

19. 必要に応じて、環境変数を随時設定します。環境変数を設定するには、**Add new Environment variable** をクリックしてから、変数の名前および値を **Name** フィールドおよび **Value** フィールドに入力します。

- 設定した Maven リポジトリからサービスをプルするイミュータブル KIE Server を設定する必要がある場合は、以下の設定を入力します。

- i. **KIE_SERVER_CONTAINER_DEPLOYMENT** 環境変数を設定します。変数には、デプロイメントが Maven リポジトリからプルする必要のあるサービス (KJAR ファイル) の ID 情報が含まれている必要があります。形式は **<containerId>=<groupId>:<artifactId>:<version>** になります。また、コンテナのエイリアス名で指定する場合には、形式は **<containerId>(<aliasId>)=<groupId>:<artifactId>:<version>** になります。以下の例に示されるように、区切り文字 **|** を使用して 2 つ以上の KJAR ファイルを指定できます (例: **containerId=groupId:artifactId:version|c2(alias2)=g2:a2:v2**)。

- ii. 外部 Maven リポジトリの設定

- 外部 Maven リポジトリを設定する必要がある場合には、以下の変数を設定します。

- **MAVEN_REPO_URL**: Maven リポジトリの URL
- **MAVEN_REPO_ID**: Maven リポジトリの ID (例: **repo-custom**)
- **MAVEN_REPO_USERNAME**: Maven リポジトリのユーザー名
- **MAVEN_REPO_PASSWORD**: Maven リポジトリのパスワード

- OpenShift 環境が公開インターネットに接続されていない場合は、**「オフラインで使用する Maven ミラーリポジトリの用意」** に従って設定した Maven ミラーにアクセスできるように設定します。以下の変数を設定してください。

- **MAVEN_MIRROR_URL**: **「オフラインで使用する Maven ミラーリポジトリの用意」** でセットアップした Maven ミラーリポジトリの URL。この URL は、OpenShift 環境の Pod からアクセスできるようにする必要があります。この KIE Server を S2I として設定している場合は、この URL をすでに入力されています。
- **MAVEN_MIRROR_OF**: ミラーから取得されるアーティファクトを定める値。この KIE Server を S2I として設定している場合は、この値を設定しません。**mirrorOf** 値の設定方法は、Apache Maven ドキュメントの **「Mirror Settings」** を参照してください。デフォルト値は **external:*** です。この値の場合、Maven はミラーから必要なアーティファクトをすべて取得し、他のリポジトリにクエリーを送信しません。外部の Maven リポジトリ (**MAVEN_REPO_URL**) を設定する場合は、ミラーからこのリポジトリ内のアーティファクトを除外するように **MAVEN_MIRROR_OF** を変更します (例: **external:*,!repo-custom**)。repo-custom は、**MAVEN_REPO_ID** で設定した ID に置き換えます。

オーサリング環境でビルトイン Business Central Maven リポジトリを使用する場合は、ミラーからこのリポジトリのアーティファクトを除外するように

MAVEN_MIRROR_OF を変更します (例: **external:*,!repo-rhpamcentr**)。

- Prometheus を使用してメトリクスを収集し、保存するように KIE Server デプロイメントを設定する必要がある場合は、**PROMETHEUS_SERVER_EXT_DISABLED** 環境変数を **false** に設定します。Prometheus メトリクス収集の設定方法は、**「KIE Server の管理および監視」** を参照してください。

- {RH-SSO} 認証を使用し、{RH-SSO} を使用したアプリケーションの対話で CORS のサポートが必要な場合は、SSO_ENABLE_CORS 変数を **true** に設定します。
- 一部のオーサリング環境では、複数のユーザーが同じ KIE Server に同時にサービスをデプロイできることを確認する必要があります。デフォルトでは、ユーザーは、Business Central を使用して KIE Server にサービスをデプロイして数秒待ってから追加サービスをデプロイする必要があります。**OpenShiftStartupStrategy** 設定はデフォルトで有効になり、この制限が発生します。制限を削除するには、**コントローラストラテジー** を使用するように **rhpm-authoring** 環境を設定します。これについての特定のニーズがない限り、この変更を行わないでください。コントローラストラテジーを有効にする場合は、Business Central と同じ環境内のすべての KIE Server でこの変更を行ってください。



注記

高可用性の Business Central を使用する環境でコントローラストラテジーを有効にしないでください。この環境では、コントローラストラテジーは正しく機能しません。

KIE Server でコントローラストラテジーを有効にするには、**KIE_SERVER_STARTUP_STRATEGY** 環境変数を **ControllerBasedStartupStrategy** に設定し、**KIE_SERVER_CONTROLLER_OPENSIFT_ENABLED** 環境変数を **false** に設定します。

次のステップ

追加の KIE Server を設定するには、**Add new KIE Server** を再びクリックし、新規サーバー設定の手順を繰り返します。

Smart Router およびプロセスインスタンス移行 (PIM) を使用せずに環境をデプロイする場合は、**Finish** をクリックしてから、**Deploy** をクリックして環境をデプロイします。それ以外の場合は、引き続き Smart Router の設定パラメーターの設定を行います。

4.2.6. 環境の Smart Router 構成の設定

デフォルトでは、デプロイされた環境には Smart Router が含まれていません。Smart Router は環境に追加できます。Smart Router の構成オプションを設定することもできます。

前提条件

- 「[環境の基本設定の構成](#)」の説明に従って、インストーラーウィザードで Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本設定を行っている。

手順

1. **Installation** タブ、**Security** タブ、**Console** タブ、または **KIE Servers** タブが開いている場合は、**Smart Router** タブが表示されるまで **Next** をクリックします。
2. **Set Smart Router** をクリックして Smart Router を環境に追加し、Smart Router を設定します。
3. 「[Smart Router のシークレットの作成](#)」の説明に従って Smart Router のシークレットを作成している場合、**Secret** フィールドにシークレットの名前を入力します。
4. 必要に応じて、Smart Router のレプリカ数を **Replicas** フィールドに入力します。

5. 必要に応じて、**Resource quotas** 下のフィールドに必要な CPU およびメモリーの上限值を入力します。

次のステップ

プロセスインスタンス移行 (PIM) サービスをデプロイする場合は、引き続きサービスをデプロイします。それ以外の場合は、**Finish** をクリックしてから、**Deploy** をクリックして環境をデプロイします。

4.2.7. 環境のプロセスインスタンス移行 (PIM) 設定

Operator を使用してプロセスインスタンス移行 (PIM) サービスをデプロイできます。プロセスインスタンスの移行サービスを使用して、2 つの異なるプロセス定義間の移行を定義できます。これは移行プランと呼ばれます。特定の KIE Server で実行中のプロセスインスタンスに対して、この移行プランを適用できます。

PIM サービスでは、サービスの操作にデータベースサーバーを使用します。

前提条件

- 「[環境の基本設定の構成](#)」の説明に従って、インストーラーウィザードで Business Automation Operator を使用して Red Hat Process Automation Manager 環境の基本設定を行っている。

手順

1. **Installation** タブ、**Security** タブ、**Console** タブ、**KIE Servers** タブ、または **Smart Router** タブが開いている場合は、**Process Instance Migration** タブが表示されるまで **Next** をクリックします。
2. **Set Process Instance Migration** をクリックして、PIM を環境に追加して PIM を設定します。
3. **Database type** フィールドで、PIM サービスが使用する必要のあるデータベースを選択します。以下の値を使用できます。
 - **mysql**: 個別の Pod に作成される MySQL サーバー。
 - **postgresql**: 個別の Pod に作成される PostgreSQL サーバー。他の設定を使用する特別な理由のない限り、この設定を使用します。
 - **h2**: 個別の Pod を必要としないビルトインされた **h2** データベースエンジン。
4. オプションで、データベースの永続ボリュームの設定パラメーターを指定します。
 - また、**Size** フィールドに、永続ボリュームのサイズを入力します。
 - **StorageClass name** フィールドで、永続ボリュームのストレージクラス名を入力します。

次のステップ

Finish をクリックしてから **Deploy** をクリックし、環境をデプロイします。

PIM サービスの使用に関する説明は、『[Business Central でのビジネスプロセスの管理とモニタリング](#)』の「[プロセスインスタンスの移行](#)」を参照してください。

4.3. OPERATOR を使用してデプロイした環境の変更

環境を Operator を使用してデプロイした場合は、通常の OpenShift の手法を使用して環境を変更することはできません。たとえば、デプロイメント設定またはサービスを削除しても、これは同じパラメーターで自動的に再作成されます。

環境を変更するには、環境の YAML の記述を変更する必要があります。パスワードなどの一般的な設定を変更し、KIE Server を追加してスケーリングできます。

手順

1. OpenShift Web クラスターコンソールでプロジェクトに移動します。
2. OpenShift Web コンソールナビゲーションパネルで **Catalog → Installed operators** または **Operators → Installed operators** を選択します。
3. 表で **Business Automation** Operator 行を見つけ、その行で **KieApp** をクリックします。この Operator を使用してデプロイした環境の情報が表示されます。
4. デプロイした環境の名前をクリックします。
5. **YAML** タブを選択します。
YAML ソースが表示されます。YAML ソースの **spec:** でコンテンツを編集して、環境の設定を変更できます。
6. Red Hat Process Automation Manager のデプロイバージョンを変更する場合は、**spec:** に以下の行を追加します。

```
version: 7.8.0
```

7.8.0 は、必要な別のバージョンに置き換えることができます。カスタムイメージを使用する場合など、自動更新が無効になっている場合は、この設定を使用して Red Hat Process Automation Manager を新規バージョンにアップグレードしてください。

7. パスワードなどの共通の設定を変更するには、**commonConfig:** の値を編集します。
8. 新しい KIE Server を追加する場合は、以下の例に示されているように、**servers:** のブロックの最後にそれらの記述を追加します。

- 名前が **server-a** と **server-a-2** のサーバー 2 台を追加するには、以下の行を追加します。

```
- deployments: 2
  name: server-a
```

- S2I プロセスのソースからビルドされるサービスを含む、イミュータブルな KIE Server を追加するには、以下の行を追加します。

```
- build:
  kieServerContainerDeployment: <deployment>
  gitSource:
    uri: <url>
    reference: <branch>
    contextDir: <directory>
```

以下の値を置き換えます。

- **<deployment>**: ソースからビルドしたデシジョンサービス (KJAR ファイル) の識別情報。形式は **<containerId>=<groupId>:<artifactId>:<version>** になります。区切り記

号 | を使用して 2 つ以上の KJAR ファイルを指定できます (例:

containerId=groupId:artifactId:version|c2=g2:a2:v2)。Maven ビルドプロセスは、Git リポジトリのソースからこのようなファイルをすべて生成する必要があります。

- **<url>**: デジジョンサービスのソースを含む Git リポジトリの URL。
 - **<branch>**: Git リポジトリのブランチ。
 - **<directory>**: Git リポジトリからダウンロードしたプロジェクトのソースへのパス。
9. KIE Server をスケーリングする場合は、**servers:** のブロックに含まれるサーバーの記述を検索して、その記述の下に **replicas:** 設定を追加します。たとえば、**replicas: 3** はサーバーを Pod 3 つにスケーリングします。
 10. 他に変更を加える場合は、利用可能な設定の CRD ソースを確認します。CRD ソースを表示するには、管理ユーザーで **oc** コマンドを使用して Red Hat OpenShift Container Platform 環境にログインし、以下のコマンドを入力します。

```
oc get crd kieapps.app.kiegroup.org -o yaml
```

11. **Save** をクリックしてから **has been updated** ポップアップメッセージを待機します。
12. **Reload** をクリックして、環境の新しい YAML の記述を表示します。

4.4. JVM 設定パラメーター

Operator を使用して Red Hat Process Automation Manager をデプロイする場合は、必要に応じて Business Central および KIE Server の多数の JVM 設定パラメーターを設定できます。これらのパラメーターは、対応するコンテナの環境変数を設定します。

以下の表では、Operator を使用して Red Hat Process Automation Manager をデプロイする際に設定できるすべての JVM 設定パラメーターの一覧を表示しています。

デフォルト設定は、ほとんどのユースケースに最適です。必要な場合にのみ変更を行ってください。

表4.1 JVM 設定パラメーター

設定フィールド	環境変数	説明	例
Java Opts の追加	JAVA_OPTS_APPEND	JAVA_OPTS で生成されたオプションに追加されるユーザー指定の Java オプション。	- Dsome.property=foo
Java 最大メモリ比	JAVA_MAX_MEM_RATIO	Java 仮想マシンに使用できるコンテナメモリーの最大パーセンテージ。残りのメモリーはオペレーティングシステムに使用されます。デフォルト値は 50 であり、50% の制限があります。 -Xmx JVM オプションを設定します。 0 の値を入力した場合、 -Xmx オプションは設定されません。	40

設定フィールド	環境変数	説明	例
Java 初期メモリー比	JAVA_INITIAL_MEM_RATIO	Java 仮想マシンに最初に使用されるコンテナメモリーの割合。デフォルト値は 25 であるため、この値が Java Max Initial Memory 値を超えない場合は、Pod メモリーの 25% が最初に JVM に割り当てられます。 -Xms JVM オプションを設定します。 0 の値を入力すると、 -Xms オプションは設定されません。	25
Java 最大初期メモリー	JAVA_MAX_INITIAL_MEMORY	Java 仮想マシンで最初に使用できるメモリーの最大量 (メガバイト単位)。 Java initial memory ratio パラメーターで設定されるように初期の割り当てメモリーがこの値よりも大きい場合、この値で設定されたメモリー量は -Xms JVM オプションを使用して割り当てられます。デフォルト値は 4096 です。	4096
Java 診断	JAVA_DIAGNOSTICS	この設定を有効にすると、追加の JVM 診断情報の標準出力への出力が有効になります。デフォルトでは無効にされています。	true
Java デバッグ	JAVA_DEBUG	この設定を有効にして、リモートデバッグをオンに切り替えます。デフォルトでは無効にされています。JVM オプション -agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=\${debug_port} を追加します。ここで、 \${debug_port} はデフォルトで 5005 に設定されます。	true
Java デバッグポート	JAVA_DEBUG_PORT	リモートデバッグに使用されるポート。デフォルト値は 5005 です。	8787
GC の最小ヒープ解放比率	GC_MIN_HEAP_FREE_RATIO	拡張を回避するためのガベージコレクション (GC) 後のヒープ解放の最小パーセンテージ。JVM オプション -XX:MinHeapFreeRatio を設定します。	20
GC の最大ヒープ解放比率	GC_MAX_HEAP_FREE_RATIO	縮小を回避するための GC 後のヒープ解放の最大パーセンテージ。JVM オプション -XX:MaxHeapFreeRatio を設定します。	40

設定フィールド	環境変数	説明	例
GC 時間比率	GC_TIME_RATIO	ガベージコレクションに費やした時間に対する、ガベージコレクション外で費やした時間 (アプリケーションの実行に費やした時間など) の比率を指定します。JVM オプション -XX:GCTimeRatio を設定します。	4
GC 適応サイズポリシーの重み	GC_ADAPTIVE_SIZE_POLICY_WEIGHT	以前の GC 時間に対する現在の GC 時間の重み付け。JVM オプション -XX:AdaptiveSizePolicyWeight を設定します。	90
GC の最大メタスペースサイズ	GC_MAX_METASPACE_SIZE	メタスペースの最大サイズ。JVM オプション -XX:MaxMetaspaceSize を設定します。	100

4.5. KIE SERVER のカスタムイメージの作成

カスタムイメージを作成して、KIE Server のデプロイメントにファイルを追加できます。次に、イメージを独自のコンテナレジストリーにプッシュする必要があります。Red Hat Process Automation Manager をデプロイする場合は、カスタムイメージを使用するように Operator を設定できます。

カスタムイメージを使用する場合は、自動のバージョンアップグレードを無効にする必要があります。新規バージョンをインストールする場合は、以前と同じ名前と、新規バージョンタグを指定してイメージをビルドし、レジストリーにそのイメージをプッシュします。その後にバージョンを変更すると、Operator が自動的に新規イメージをプルします。Operator での製品バージョンの変更に関する説明は、[「Operator を使用してデプロイした環境の変更」](#) を参照してください。

特に、次のタイプのカスタムイメージを作成できます。

- 追加の RPM パッケージを含めた KIE Server のカスタムイメージ
- 追加の JAR ライブラリーを含めた KIE Server のカスタムイメージ

4.5.1. 追加の RPM パッケージを含めたカスタムの KIE Server イメージの作成

追加の RPM パッケージのインストール先のカスタム KIE Server イメージを作成できます。このイメージをカスタムレジストリーにプッシュして、KIE Server のデプロイに使用できます。

Red Hat Enterprise Linux 8 リポジトリから任意のパッケージをインストールできます。以下の例では、**ps** ユーティリティーが含まれる **procps-ng** パッケージをインストールしていますが、変更して他のパッケージをインストールすることができます。

手順

1. **podman login** コマンドを使用して **registry.redhat.io** レジストリーの認証を行います。レジストリーの認証に関する詳細は、[「Red Hat コンテナレジストリーの認証」](#) を参照してください。

- サポートされている KIE Server のベースイメージをダウンロードするには、次のコマンドを入力します。

```
podman pull registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
```

- ベースイメージをもとにカスタムイメージを定義する **Dockerfile** を作成します。このファイルで、現在のユーザーを **root** に変更して、**yum** コマンドで RPM パッケージをインストールしてから **USER 185** (Red Hat JBoss EAP ユーザー) に戻します。以下の例では、**Dockerfile** ファイルの内容を示します。

```
FROM registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
USER root
RUN yum -y install procps-ng
USER 185
```

必要に応じて RPM ファイルの名前を置き換えます。**yum** コマンドは自動的に Red Hat Enterprise Linux 8 リポジトリからの全依存関係を自動的にインストールします。複数の RMP ファイルをインストールする必要がある場合があります。今回は、**RUN** コマンドを複数回使用します。

- Dockerfile** を使用してカスタムイメージをビルドします。レジストリー名など、イメージの完全修飾名を指定します。ベースイメージと同じバージョンタグを使用する必要があります。イメージをビルドするには、以下のコマンドを入力します。

```
podman build . --tag registry_address/image_name:7.8.0
```

以下に例を示します。

```
podman build . --tag registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
```

- ビルドが完了したら、イメージを実行してログインし、カスタマイズが成功したことを確認します。ターミナルで以下のコマンドを入力します。

```
podman run -it --rm registry_address/image_name:7.8.0 /bin/bash
```

以下に例を示します。

```
podman run -it --rm registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0 /bin/bash
```

イメージのシェルプロンプトで、コマンドを入力して RPM がインストールされていることをテストし、**exit** と入力します。たとえば、**procps-ng** の場合は **ps** コマンドを実行します。

```
[jboss@c2fab36b778e ~]$ ps
PID TTY      TIME CMD
  1 pts/0    00:00:00 bash
 13 pts/0    00:00:00 ps
[jboss@c2fab36b778e ~]$ exit
```

- カスタムイメージをレジストリーにプッシュするには、次のコマンドを入力します。

```
podman push registry_address/image_name:7.8.0
docker://registry_address/image_name:7.8.0
```

以下に例を示します。

```
podman push registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
docker://registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
```

次のステップ

KIE Server をデプロイする場合は、イメージ名と namespace を設定してレジストリーにカスタムイメージを指定します。**Set KIE Server image** をクリックして、**Kind** の値を **DockerImage** に変更してから、バージョンタグがないレジストリー名など、イメージ名を指定します。以下に例を示します。

```
registry.example.com/custom/rhpam-kieserver-rhel8
```

Operator を使用した KIE Server のデプロイに関する詳細は、[「環境のカスタム KIE Server 設定の構成」](#) を参照してください。

4.5.2. 追加の JAR ファイルを使用したカスタム KIE Server イメージの作成

追加の JAR ファイル (単数、複数問わず) のインストール先のカスタムの KIE Server イメージを作成してサーバーの機能を拡張できます。このイメージをカスタムレジストリーにプッシュして、KIE Server のデプロイに使用できます。

手順

1. KIE Server で動作するカスタムライブラリーを開発します。以下のドキュメントと例を使用して、ライブラリーを開発できます。
 - [『KIE Server の管理とモニタリング』の「KIE Server 機能および拡張」](#)
 - [「Domain-specific Prometheus metrics with Red Hat Process Automation Manager and Decision Manager」](#)
 - [Extend KIE Server with additional transport](#)
2. JAR ファイルが **target** ディレクトリーに配置されるように Maven を使用してライブラリーをビルドします。この例では、**custom-kieserver-ext-1.0.0.Final.jar** のファイル名を使用します。
3. **podman login** コマンドを使用して **registry.redhat.io** レジストリーの認証を行います。レジストリーの認証に関する詳細は、[「Red Hat コンテナレジストリーの認証」](#) を参照してください。
4. サポートされている KIE Server のベースイメージをダウンロードするには、次のコマンドを入力します。

```
podman pull registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
```

5. ベースイメージをもとにカスタムイメージを定義する **Dockerfile** を作成します。このファイルは JAR ファイル (単数、複数を問わず) を **/opt/eap/standalone/deployments/ROOT.war/WEB-INF/lib/** ディレクトリーにコピーする必要があります。以下の例では、**Dockerfile** ファイルの内容を示します。

```
FROM registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
COPY target/custom-kieserver-ext-1.0.0.Final.jar
/opt/eap/standalone/deployments/ROOT.war/WEB-INF/lib/
```

6. **Dockerfile** を使用してカスタムイメージをビルドします。レジストリー名など、イメージの完全修飾名を指定します。ベースイメージと同じバージョンタグを使用する必要があります。イメージをビルドするには、以下のコマンドを入力します。

```
podman build . --tag registry_address/image_name:7.8.0
```

以下に例を示します。

```
podman build . --tag registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
```

7. カスタムイメージをレジストリーにプッシュするには、次のコマンドを入力します。

```
podman push registry_address/image_name:7.8.0  
docker://registry_address/image_name:7.8.0
```

以下に例を示します。

```
podman push registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0  
docker://registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
```

次のステップ

KIE Server をデプロイする場合は、イメージ名と namespace を設定してレジストリーにカスタムイメージを指定します。**Set KIE Server image** をクリックして、**Kind** の値を **DockerImage** に変更してから、バージョンタグがないレジストリー名など、イメージ名を指定します。以下に例を示します。

```
registry.example.com/custom/rhpam-kieserver-rhel8
```

Operator を使用した KIE Server のデプロイに関する詳細は、[「環境のカスタム KIE Server 設定の構成」](#) を参照してください。

第5章 RED HAT OPENSIFT CONTAINER PLATFORM バージョン 3 のデプロイメントからの情報の移行

以前に Red Hat OpenShift Container Platform バージョン 3 で Red Hat Process Automation Manager デプロイメントを使用していた場合は、バージョン 3 のデプロイメントから Red Hat OpenShift Container Platform バージョン 4 の新しいデプロイメントに情報を移行できます。

情報を移行する前に、Operator を使用して、新しい Red Hat Process Automation Manager インフラストラクチャーを Red Hat OpenShift Container Platform バージョン 4 にデプロイする必要があります。以前のインフラストラクチャーのデプロイメントに存在する要素を、新しいデプロイメントにも追加します。以下に例を示します。

- 既存のオーサリングデプロイメントの場合は、Business Central と最低でも KIE Server 1 台を含めて新しいオーサリングインフラストラクチャーを作成します。
- 既存のイミュータブル KIE Server の場合は、同じアーティファクトで新しいイミュータブル KIE Server をデプロイします。
- MySQL または PostgreSQL データベース Pod を使用する既存の KIE Server の場合は、データベース Pod のタイプが同じ KIE Server を新たにデプロイします。
- 外部データベースサーバーを使用する既存の KIE Server の場合には、同じ認証情報、同じ外部データベースサーバーを使用する KIE Server を新たにデプロイします。このサーバーは、同じデータベースに接続するため、プロセスコンテンツの状態を読み込むことができます。



注記

KIE Server が H2 ビルトインのデータベースを使用する場合、プロセスコンテキストの状態の移行はサポートされません。

Smart Router では、移行は必要ありません。Smart Router の新規デプロイメントは、自動的に新しい KIE Server 上のサービスと連携します。

5.1. BUSINESS CENTRAL での情報の移行

Red Hat OpenShift Container Platform バージョン 3 に、既存のオーサリング環境がある場合は、この環境の Business Central から **.niogit** リポジトリと Maven リポジトリを Red Hat OpenShift Container Platform バージョン 4 の新規デプロイメントにある Business Central にコピーします。このアクションで、新しいデプロイメントにすべて同じプロジェクトとアーティファクトが作成されます。

前提条件

- Red Hat OpenShift Container Platform バージョン 3 および Red Hat OpenShift Container Platform バージョン 4 のインフラストラクチャーの両方に、ネットワークでアクセスできるマシンが必要です。
- 対象のマシンに Red Hat OpenShift Container Platform バージョン 4 からの **oc** コマンドラインクライアントをインストールしておく必要があります。コマンドラインクライアントのインストール方法は、Red Hat OpenShift Container Platform ドキュメントの「[CLI ツール](#)」を参照してください。

手順

1. Business Central や KIE Server など、以前のデプロイメントや新しいデプロイメントの要素に接続されている Web クライアントやクライアントアプリケーションがないことを確認します。
2. 空の一時ディレクトリを作成して、そのディレクトリに移動します。
3. **oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 3 インフラストラクチャーにログインし、以前のデプロイメントが含まれるプロジェクトに切り替えます。
4. 以前のデプロイメントにある Pod 名を表示するには、以下のコマンドを実行します。

```
oc get pods
```

Business Central の Pod を検索します。この Pod の名前には **rhpmcentr** が含まれます。高可用性のデプロイメントでは、Business Central Pod はどれでも使用できます。

5. 以下の例のように、**oc** コマンドを使用して、**.niogit** リポジトリと Maven リポジトリを Pod からローカルマシンにコピーします。

```
oc cp myapp-rhpmcentr-5-689mw:/opt/kie/data/.niogit .niogit
oc cp myapp-rhpmcentr-5-689mw:/opt/kie/data/maven-repository maven-repository
```

6. **oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 4 インフラストラクチャーにログインし、新しいデプロイメントが含まれるプロジェクトに切り替えます。
7. 新しいデプロイメントにある Pod 名を表示するには、以下のコマンドを実行します。

```
oc get pods
```

Business Central の Pod を検索します。この Pod の名前には **rhpmcentr** が含まれます。高可用性のデプロイメントでは、Business Central Pod はどれでも使用できます。

8. 以下の例のように、**oc** コマンドを使用して、**.niogit** リポジトリと Maven リポジトリをローカルマシンから Pod にコピーします。

```
oc cp .niogit myappnew-rhpmcentr-abd24:/opt/kie/data/.niogit
oc cp maven-repository myappnew-rhpmcentr-abd24:/opt/kie/data/maven-repository
```

5.2. KIE SERVER の MYSQL データベースの移行

Red Hat OpenShift Container Platform バージョン 3 の環境に、MySQL データベース Pod を使用する KIE Server が含まれる場合は、MySQL データベースコンテンツを、以前のデプロイメントから新しいデプロイメントにコピーします。このアクションにより、既存のプロセスの状態が新しいデプロイメントにコピーされます。

前提条件

- Red Hat OpenShift Container Platform バージョン 3 および Red Hat OpenShift Container Platform バージョン 4 のインフラストラクチャーの両方に、ネットワークでアクセスできるマシンが必要です。
- 対象のマシンに Red Hat OpenShift Container Platform バージョン 4 からの **oc** コマンドラインクライアントをインストールしておく必要があります。コマンドラインクライアントのインストール方法は、Red Hat OpenShift Container Platform ドキュメントの「[CLI ツール](#)」を参照してください。

- MySQL バージョン 8 以降または MariaDB バージョン 10 以降で提供されるクライアントアプリケーション **mysql** および **mysqldump** がインストールされている。

手順

- Business Central や KIE Server など、以前のデプロイメントや新しいデプロイメントの要素に接続されている Web クライアントやクライアントアプリケーションがないことを確認します。
- 空の一時ディレクトリを作成して、そのディレクトリに移動します。
- oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 3 インフラストラクチャーにログインし、以前のデプロイメントが含まれるプロジェクトに切り替えます。
- 以前のデプロイメントにあるデプロイメント設定名を表示するには、以下のコマンドを実行します。

```
oc get dc
```

KIE Server に対応する **mysql** デプロイメント設定を検索します。

- 以下のようにデプロイメント設定の YAML を表示します。

```
oc edit dc/myapp-mysql
```

以下のように、このファイルでデータベースサーバーのユーザー名 (通常 **rhpm**) およびパスワードを検索します。

```
- name: MYSQL_USER
  value: rhpm
- name: MYSQL_PASSWORD
  value: NDajIV7!
```

ユーザー名とパスワードを記録します。ファイルに変更を加えないでください。



注記

次のコマンドを使用して、ユーザー名とパスワードを取得することもできます。

```
oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_USER")]}'.value

oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_PASSWORD")]}'.value
```

- 以前のデプロイメントにあるサービス名を表示するには、以下のコマンドを実行します。

```
oc get svc
```

KIE Server に対応する **mysql** サービスを検索します。

- 以下の例のように、別のターミナルウィンドウで、サービスの名前およびポート番号を使用してローカルホストから **mysql** サービスへのポート転送を開始します。


```
oc port-forward service/myapp-mysql 3306:3306
```

8. 以下の例のように、記録したユーザー名を使用して、フルデータベースダンプを作成します。

```
mysqldump --all-databases -u rhpam -p -h 127.0.0.1 > mysqldump.sql
```

プロンプトが表示されたら、記録したパスワードを入力します。ダンプの作成にはかなり時間がかかる場合があります。

9. kbd:[Ctrl+C] のキーボードの組み合わせを使用して、別のウィンドウのポート転送を停止します。
10. **oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 4 インフラストラクチャーにログインし、新しいデプロイメントが含まれるプロジェクトに切り替えます。
11. 新しいデプロイメントにあるデプロイメント設定名を表示するには、以下のコマンドを実行します。

```
oc get dc
```

KIE Server に対応する **mysql** デプロイメント設定を検索します。

12. 以下のようにデプロイメント設定の YAML を表示します。

```
oc edit dc/myappnew-mysql
```

このファイルでデータベースサーバーのユーザー名 (通常 **rhpam**) およびパスワードを検索します。ユーザー名とパスワードを記録します。ファイルに変更を加えないでください。



注記

次のコマンドを使用して、ユーザー名とパスワードを取得することもできます。

```
oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_USER")]}'.value
```

```
oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_PASSWORD")]}'.value
```

13. 新しいデプロイメントにあるサービス名を表示するには、次のコマンドを実行します。

```
oc get svc
```

KIE Server に対応する **mysql** サービスを検索します。

14. 以下の例のように、別のターミナルウィンドウで、サービスの名前およびポート番号を使用してローカルホストから **mysql** サービスへのポート転送を開始します。

```
oc port-forward service/myappnew-mysql 3306:3306
```

15. 以下のように、記録したユーザー名を使用してデータベースのダンプを復元します。

```
mysql -u rhpam -p -h 127.0.0.1 < mysqldump.sql
```

-

プロンプトが表示されたら、記録したパスワードを入力します。復元にはかなり時間がかかる場合があります。

16. `kbd:[Ctrl+C]` のキーボードの組み合わせを使用して、別のウィンドウのポート転送を停止します。

5.3. KIE SERVER のPOSTGRESQL データベースの移行

Red Hat OpenShift Container Platform バージョン 3 の環境に、PostgreSQL データベース Pod を使用する KIE Server が含まれる場合は、PostgreSQL データベースコンテンツを、以前のデプロイメントから新しいデプロイメントにコピーします。このアクションにより、既存のプロセスの状態が新しいデプロイメントにコピーされます。

前提条件

- Red Hat OpenShift Container Platform バージョン 3 および Red Hat OpenShift Container Platform バージョン 4 のインフラストラクチャーの両方に、ネットワークでアクセスできるマシンが必要です。
- 対象のマシンに Red Hat OpenShift Container Platform バージョン 4 からの **oc** コマンドラインクライアントをインストールしておく必要があります。コマンドラインクライアントのインストール方法は、Red Hat OpenShift Container Platform ドキュメントの「[CLI ツール](#)」を参照してください。
- PostgreSQL バージョン 10 以降で提供される **psql** および **pg_dump** クライアントアプリケーションがインストールされている。

手順

1. Business Central や KIE Server など、以前のデプロイメントや新しいデプロイメントの要素に接続されている Web クライアントやクライアントアプリケーションがないことを確認します。
2. 空の一時ディレクトリーを作成して、そのディレクトリーに移動します。
3. **oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 3 インフラストラクチャーにログインし、以前のデプロイメントが含まれるプロジェクトに切り替えます。
4. 以前のデプロイメントにあるデプロイメント設定名を表示するには、以下のコマンドを実行します。

```
oc get dc
```

KIE Server に対応する **postgresql** デプロイメント設定を検索します。

5. 以下のようにデプロイメント設定の YAML を表示します。

```
oc edit dc/myapp-postgresql
```

以下のように、このファイルで、データベースサーバーのユーザー名 (通常 **rhpmam**)、パスワード、およびデータベース名 (通常 **rhpmam7**) を検索します。

```
- name: POSTGRESQL_USER
  value: rhpmam
- name: POSTGRESQL_PASSWORD
```

```
value: NDaJIV7!
- name: POSTGRESQL_DATABASE
value: rhpam7
```

ユーザー名、パスワード、およびデータベース名を記録します。ファイルに変更を加えないでください。



注記

次のコマンドを使用して、ユーザー名とパスワード、データベース名を取得することもできます。

```
oc get dc/myapp-postgresql -
  ojsonpath='{.spec.template.spec.containers[0].env[?
  (@.name=="POSTGRESQL_USER")]}'.value

oc get dc/myapp-postgresql -
  ojsonpath='{.spec.template.spec.containers[0].env[?
  (@.name=="POSTGRESQL_PASSWORD")]}'.value

oc get dc/myapp-postgresql -
  ojsonpath='{.spec.template.spec.containers[0].env[?
  (@.name=="POSTGRESQL_DATABASE")]}'.value
```

+

6. 以前のデプロイメントにあるサービス名を表示するには、以下のコマンドを実行します。

```
oc get svc
```

KIE Server に対応する **postgresql** サービスを検索します。

7. 以下の例のように、別のターミナルウィンドウで、サービスの名前およびポート番号を使用してローカルホストから **postgresql** サービスへのポート転送を開始します。

```
oc port-forward service/myapp-postgresql 5432:5432
```

8. 以下のように、記録したユーザー名とデータベース名を使用して、データベースのダンプを作成します。

```
pg_dump rhpam7 -h 127.0.0.1 -U rhpam -W > pgdump.sql
```

プロンプトが表示されたら、記録したパスワードを入力します。ダンプの作成にはかなり時間がかかる場合があります。

9. kbd:[Ctrl+C] のキーボードの組み合わせを使用して、別のウィンドウのポート転送を停止します。
10. **oc** コマンドを使用して、Red Hat OpenShift Container Platform バージョン 4 インフラストラクチャーにログインし、新しいデプロイメントが含まれるプロジェクトに切り替えます。
11. 新しいデプロイメントにあるデプロイメント設定名を表示するには、以下のコマンドを実行します。

```
oc get dc
```

KIE Server に対応する **postgresql** デプロイメント設定を検索します。

12. 以下のようにデプロイメント設定の YAML を表示します。

```
oc edit dc/myappnew-postgresql
```

このファイルで、データベースサーバーのユーザー名 (通常 **rhpm**)、パスワード、およびデータベース名 (通常 **rhpm7**) を検索します。ユーザー名、パスワード、およびデータベース名を記録します。ファイルに変更を加えないでください。



注記

次のコマンドを使用して、ユーザー名とパスワード、データベース名を取得することもできます。

```
oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_USER")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_PASSWORD")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_DATABASE")]}'.value
```

13. 新しいデプロイメントにあるサービス名を表示するには、次のコマンドを実行します。

```
oc get svc
```

KIE Server に対応する **postgresql** サービスを検索します。

14. 以下の例のように、別のターミナルウィンドウで、サービスの名前およびポート番号を使用してローカルホストから **postgresql** サービスへのポート転送を開始します。

```
oc port-forward service/myappnew-postgresql 5432:5432
```

15. 以下のように、記録したユーザー名、データベース名を使用してデータベースのダンプを復元します。

```
psql -h 127.0.0.1 -d rhpm7 -U rhpm -W < pgdump.sql
```

プロンプトが表示されたら、記録したパスワードを入力します。復元にはかなり時間がかかる場合があります。

表示されたデータベースエラーメッセージを確認します。すでに存在するオブジェクトに関するメッセージは正常です。

16. kbd:[Ctrl+C] のキーボードの組み合わせを使用して、別のウィンドウのポート転送を停止します。

付録A バージョン情報

本書の最終更新日：2021年6月25日（金）