­­

Whitepaper

Application Release Automation (ARA)

HCM 2018.11

For Linux operating system

Continuous Integration, Deployment and Testing by ARA using Jenkins plugin

Document release date: December 2018

Software release date: December 2018



Contents

[What is Application Release Automation (ARA)? 3](#_Toc534394708)

[Why is ARA required? 3](#_Toc534394709)

[Continuous integration and deployment using ARA and Jenkins (build tool) 3](#_Toc534394710)

[Software required for the integration 3](#_Toc534394711)

[Configuring Jenkins 4](#_Toc534394712)

[Uploading the Micro Focus ARA plugin on the Jenkins server 4](#_Toc534394713)

[Installing the Micro Focus ARA plugin 4](#_Toc534394714)

[Before enabling the ARA plugin 4](#_Toc534394715)

[Enabling the ARA plugin 4](#_Toc534394716)

[Configuring the Pet Clinic sample application project 5](#_Toc534394717)

[Configuring the Micro Focus ARA plugin for the Pet Clinic sample application 7](#_Toc534394718)

[Troubleshooting ARA Integration with Jenkins 10](#_Toc534394719)

[Send documentation feedback 11](#_Toc534394720)

[Legal notices 11](#_Toc534394721)

# What is Application Release Automation (ARA)?

ARA is a continuous delivery solution that provides deployment and release management of complex multi-tier applications across the application lifecycle. It automates the deployment of applications by embracing existing content from Chef, SA, and so on and representing this content as components. These components can be used in a graphical designer to create an application model.

One of the important features of ARA is that the model is used to trigger deployments automatically with Jenkins.

This document provides information about integrating ARA with Jenkins.

# Why is ARA required?

Software engineering builds are subject to continuous deployment and testing on the principles of frequent code commits, build automation, faster and frequent builds, automated application deployment, and test automation. On top of continuous integration, software development teams also continuously deliver qualified software applications to their test and production teams. One of the challenges that most software development teams face in the process of continuous integration and continuous delivery is the ability to automate the deployment of applications in a simple and consistent manner and run tests on the deployed instance. ARA is built to solve this problem.

Using ARA, users can deploy the application and run tests automatically by integrating with ALM. This white paper describes how ARA can be integrated with ALM. ARA is integrated with ALM through Jenkins. Jenkins acts as orchestrator between ARA and ALM.

# Continuous integration and deployment using ARA and Jenkins (build tool)

**Import HCM certificate**

1. Identify the nginx container ID using the following command: docker ps | grep -i nginx | grep -i hcm | grep -i /bin/sh
2. Use Docker CP to copy the public cert from nginx container to localhost using the following command: docker cp <nginx container ID>:/ingress-controller/ssl/hcm2-hcm-default-secret.pem /tmp/

## Software required for the integration

Jenkins must be integrated with ARA for the ALM integration to work. To integrate Jenkins with ARA, you must install the following software:

1. Install Jenkins from jenkins-ci.org/
2. Install the JDK version 1.7x on the Jenkins server.
3. Install Collabnet Subversion Edge from collab.net/support/documentation
4. Install TortoiseSVN from tortoisesvn.net. Install the latest version and use the default settings. After the installation, you can see new options when you right-click a file or folder in Windows Explorer.
5. Download and install Maven from maven.apache.org

## Configuring Jenkins

After installing the software listed at Software required for the integration, configure Jenkins as follows (the following steps are for Jenkins version 2.46.3):

1. Ensure that JDK and Maven are installed.
2. Open Jenkins and click the **Manage Jenkins** option in the Jenkins dashboard.
3. Click **Configure System**.
4. In the **JDK** section, click **JDK installations** and then **Add JDK**.
5. Enter the name and path of the JAVA\_HOME environment variable.
6. Deselect the **Install automatically** check box.
7. In the Maven section, click **Maven installations** and then **Add Maven**.
8. Enter the name and path of the MAVEN\_HOME environment variable.
9. Deselect the **Install automatically** check box.
10. Enter the value of the MAVEN\_OPTS environment variable.
11. Click **Save**.

**Note**: The plugin is not supported on Jenkins version above 2.46.3.

## Uploading the Micro Focus ARA plugin on the Jenkins server

You must now upload and enable the Micro Focus ARA plugin on the server in which Jenkins is installed.

## Installing the Micro Focus ARA plugin

1. Log on to the Jenkins dashboard using the http://<host>:<port>/ URL. Use the host and port information appropriate for your Jenkins environment.
2. Click **Manage Jenkins** on the Jenkins dashboard.
3. Click **Manage Plugins**.
4. Select the **Advanced** tab.
5. In the **Upload Plugin** section, browse to the path of the ARA plugin file at <HCM\_PERSISTENT\_VOLUME>shared/content-tools/codar/CodarIntegrationPoints/Jenkins/microfocus-codar-plugin-01.83.000.hpi
6. Click **Upload**.  
     
   Upon successful upload, the system returns a ‘Success’ message.

## Before enabling the ARA plugin

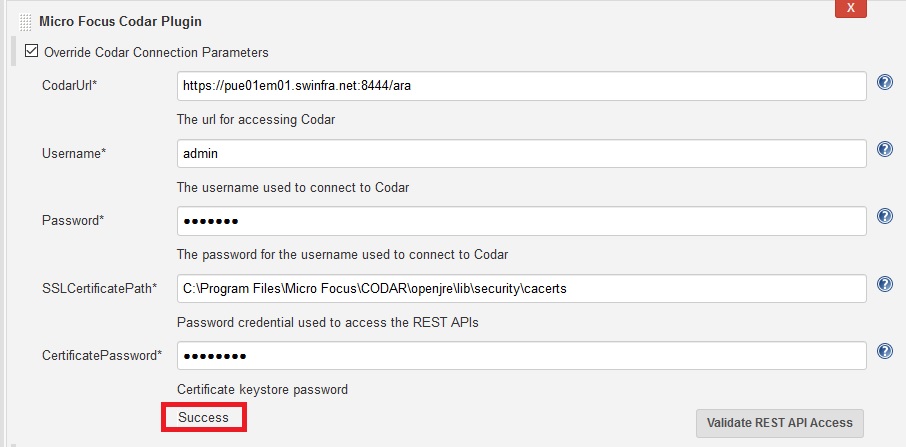
Please manually download the supporting Content pack [ARA OO Content] from the below marketplace URL and manually upload it in the HCM OO. Plugin need this content pack for its functioning.

URL: <https://marketplace.microfocus.com/itom/content/ara-oo-content>

## Enabling the ARA plugin

1. Click **Manage Jenkins** on the Jenkins dashboard.
2. Click **Configure System**.

Scroll down to the **Micro Focus ARA Plugin** section and select the **Enable** check box, if not selected by default.  
  
Enabling the ARA plugin



1. Provide the following details:
   * 1. ARAUrl – URL used to log on to ARA.
     2. Username – Name of the user that has administrative privileges in ARA.  
         Do not use the default administrator user because it may cause a security issue. .
     3. Password – Password of the ARA user.
     4. SSLCertificatePath – Enter SSL certificate path for ARA and pick up the certificate from the ARA setup. If open JRE is used during the ARA installation, then it is on the computer on which ARA is installed (Example: Windows: In the C:\ProgramFiles\MicroFocus\ARA\openjre\lib\security\cacerts path. Linux: In the /root/temp/cacerts path. Ensure that ‘cacerts path’ has the read/write permission on the server on which ARA is installed). Details about the JRE used during installation is located in the **csa.properties** file.
     5. CertificatePassword – Enter the SSL certificate keystore password for ARA. The default password is ‘changeit’.
2. Test the link by clicking ‘Validate REST API Access’.
3. Ensure that the validation returns ‘Success’ as shown in the screenshot below, by resolving connectivity issues, if any.
4. Click **Save**.

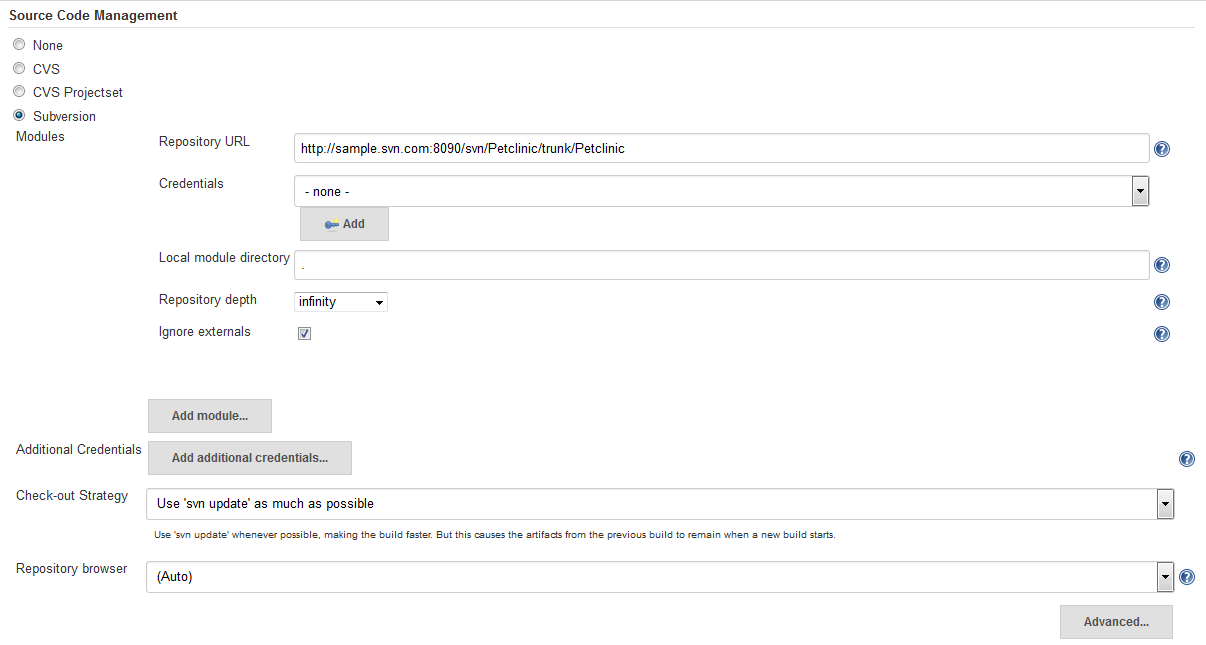
**Note:** For security reasons, you must configure the ARA plugin with SSL and HTTPS enabled and with TLS Protocol Version 1.2 only.

## Configuring the Pet Clinic sample application project

This section describes how to configure the ARA-ALM integration by means of a sample application called Pet Clinic.

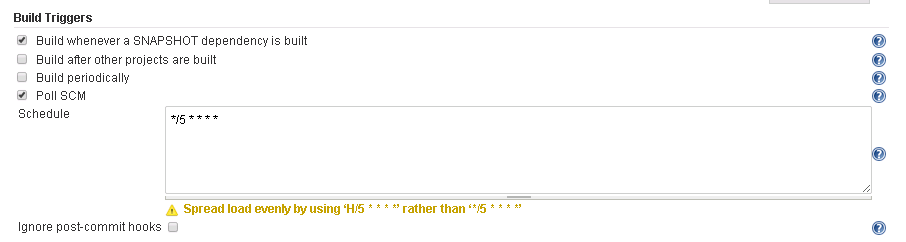
1. Download the Pet Clinic source code from [GitHub](https://github.com/spring-projects/spring-petclinic/archive/master.zip).
2. Check in the source code for the Pet Clinic project into the SVN server.
3. Create a new Pet Clinic project in the Jenkins dashboard:
   * 1. Click **New Item** -> **Maven project**.
     2. Enter Pet Clinic in the **Item name** text box.
     3. Click **OK**.  
        The Pet Clinic link is displayed in the Jenkins dashboard.
4. Click the Pet Clinic link on the Jenkins dashboard, and then click the **Configure** link in the page that opens.
5. Configure SVN for the Pet Clinic project by choosing the **Subversion Modules** option in the **Source Code Management** section area and adding the SVN Pet Clinic source code URL in the **Repository URL** field.
6. After saving, update the SVN credentials as shown below.

Source code management

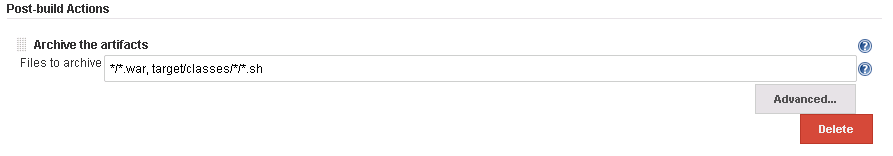


1. Configure Jenkins to automatically trigger a build if some code is checked in by selecting the **Poll SCM** check box and adding \*/5 \* \* \* \* in the **Schedule** field. This indicates the poll every five minutes if a code commit happens.

Figure 4: Triggering a build



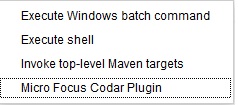
1. Scroll down to the **Post-build Actions** section and click **Add post-build action**, select **Archive the artifacts**, and then enter \*/\*.war, target/classes/\*/\*.sh in the **Files to archive** text box.



## Configuring the Micro Focus ARA plugin for the Pet Clinic sample application

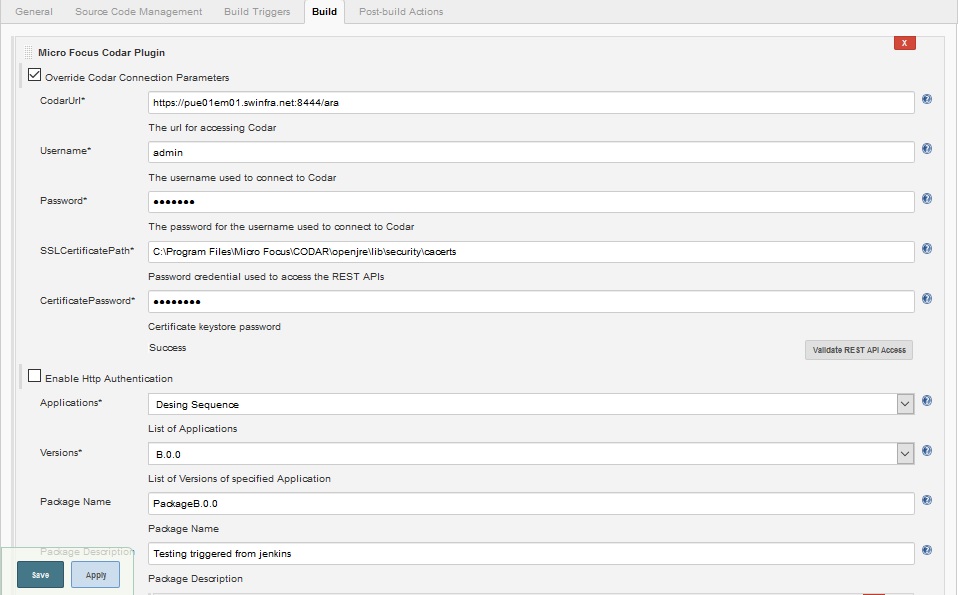
1. Click the Pet Clinic link on the Jenkins dashboard, and then click the **Configure** link on the page that opens.
2. Click **Add build step** and select **Micro Focus ARA Plugin**.   
   **Note**: For builds with pre-build and post-build actions, you must configure the ARA plugin as part of post-build actions. For example, in a Maven project, configure the ARA plugin-in as a post-build action.

Configuring the ARA plugin



1. Enter the ARA plugin properties as follows:

ARA properties

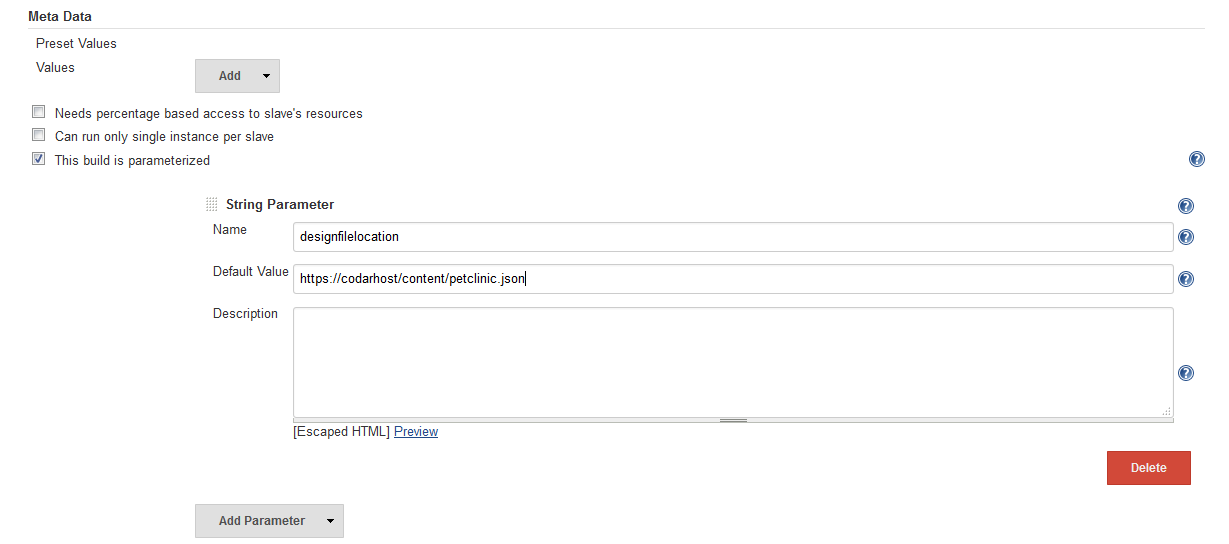


**Note:** If you select the ‘Override ARA Connection Parameters’ option, you must provide the details from steps ‘a’ to ‘e’. Skip to step ‘f’, if you do not select this option.

* + 1. ARAUrl – URL used to log on to ARA.
    2. Username – Name of the user that has administrative privileges in ARA.  
        Do not use the default administrator user because it may cause a security issue. After installing ARA and configuring LDAP, a user is added to the Application Architect role. Use the credentials for that user here.
    3. Password – Password of the ARA user.
    4. SSLCertificatePath – Enter SSL certificate path for ARA and pick up the certificate from the ARA setup. If open JRE is used during the ARA installation, then it is in the computer in which ARA is installed (in the Program Files\Micro Focus\ARA\openjre\lib\cacerts path). Details about the JRE used during installation is located in the **csa.properties** file.
    5. CertificatePassword – Enter the SSL certificate keystore password for ARA. The default password is ‘changeit’.
    6. Enable Http Authentication – Select to enable the Jenkins user name and password. This is required for ARA to pull deployment artifacts from Jenkins.
    7. HttpUsername - User name for accessing artifacts from the HTTP location. For example, if the artifacts are at a location in Jenkins, enter the user name of the Jenkins server.
    8. HttpPassword – Password for accessing artifacts from the HTTP location. For example, if the artifacts are at a location in Jenkins, enter the password of the Jenkins server.
    9. Application Design Location – This is an optional parameter which can be configured in 3 ways:
* Relative path and file name of the application design JSON file from the source repository URL. The relative path must be separated by a slash. For example, designs\PetClinicApp.json.

For example, if the source repository is <https://myrepo.mydomain.com/mypetclinicapp>, the application JSON can exist in a directory named designs and the JSON file can be created with the any name as required. The JSON file is a part of the repository in which the application source code is located.

* URL of design location can be specified. For example, you can specify the location of design file **http://<hostname>/<designfile.json**>
* You can create an environment variable and pass it as a variable.



* + 1. Continuous Promote – If this option is selected, packages are created in the first stage and release gate actions are executed. If all actions are successful, only then is the package promoted to the next stage and so on till it reaches the last stage. In the last stage, the package is executed after all actions are executed. If this option is not selected, then the package is created and deployed. Release gate actions are not executed and not promoted.
* If this option is selected as ‘Yes’, you cannot specify the environment.
* If this option is selected as ‘No’, you must specify the environment.
  + 1. Package Name – Name of the package  
         
       **Note:** Package Name can be a build parameter or Jenkins inbuilt variables like $BUILD\_ID. For example, package name can be derived from the build parameter ’PackageName’ and the value can be ’Package: $BUILD\_ID’. You can set the package name as ‘$PackageName’.
    2. Package Description - Description of the package. The Jenkins URL is appended to the package description.
    3. Application Design Type – From the list, select either Topology Designs or Sequenced Designs. If you select ‘Topology Designs’, then the List of Designs drop-down lists all topology designs for selection. If you select ‘Sequenced Design’, then the List of Designs drop-down lists all sequenced designs.
    4. List of Designs – Select the design from the list. The list shows topology designs or sequenced designs based on the application design type you selected.
    5. Versions – Select the version number of the design. The list shows all versions of the design selected above.
    6. Environment – Environment in which the provider to be used for deployment is contained. Environments are created in the **Resource provider** tile under ARA.

**Note:** A resource environment is required for the ARA plug-in to work.

* + 1. Package Properties – The following Jenkins environment variables are supported:
    - ${build\_id}
    - ${build\_number}
    - ${build\_url}
    - ${job\_name}

You can specify the value of the property in the following ways:

* Artifact name like petclinic.war   
    
  **Note:** An artifact is a group of files with a .war .jar or .zip. extension. You can directly specify the Jenkins artifact name as ‘petclinic.war’ for Jenkins. The ‘petclinic.war’ will then automatically be resolved to the http URL of the Jenkins artifact, and substituted in the package property value. For artefacts other than Jenkins, you need to specify the complete http URL.
* Property value – The property values allows user to specify parameterized values which allows users to dynamically reference the current build artifacts built by the Jenkins Job.
  + 1. Extended Properties File –Enter the name of the properties file. This is an optional field. Specify this file only if you want to specify a different CI process than what is provided by default. The properties file can specify a different OO flow containing the necessary CI logic. You can also specify a different flow D by creating a properties file with the UUID as the key and UUID of the OO flow as the value. For example, uuid=asdaasdasdsdasdad99f. You can also specify the required properties to this flow as key value pairs in the properties file.

# Troubleshooting ARA Integration with Jenkins

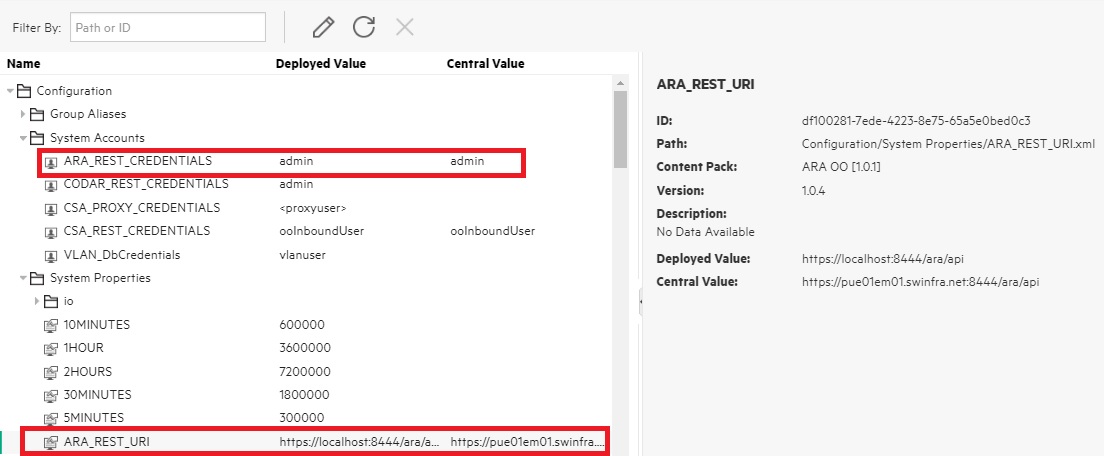
This section contains some of the issues that you may encounter when integrating ARA with Jenkins, and workarounds to troubleshoot these issues.

#### Plug-in execution begins but fails

Problem: ARA plug-in execution begins but fails with some error

Solution/Workaround: In the Operations Orchestration console>>Content Management>>Configuration Items>>Systems account>>System properties, overwrite the following properties with the appropriate credentials:

* ARA\_REST\_CREDENTIALS
* ARA\_REST\_URI



#### Trouble integrating ARA with Jenkins

Problem: Exception/error while integrating ARA with Jenkins.

Solution/Workaround: Check the following logs for detailed information:

* Jenkins-related issues: jenkins.err.log (This file exists in the Jenkins installation directory).
* ARA-related-issues: csa.log (This file exists in the following path: <CODAR\_HOME>\jboss-as\standalone\log)
* OO-related issues: Operations Orchestrations console.

Send documentation feedback

If you have comments about this document, you can [contact the documentation team](mailto:docs.feedback@microfocus.com) by email.

If an email client is configured on this system, click the link above and an email window opens. Add subject and your feedback to the email, and click send. If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to [docs.feedback@microfocus.com](mailto:docs.feedback@microfocus.com).

We appreciate your feedback!

# Legal notices

© Copyright 2008 - 2018 Micro Focus or one of its affiliates.

The only warranties for products and services of Micro Focus and its affiliates and licensors (“Micro Focus”) are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.  
  
Contains Confidential Information. Except as specifically indicated otherwise, a valid license is required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.