Aras DevOps

Aras 3D Visualization Configuration Guide

Document #: D-009026

Last Modified: *2/5/2024*

Copyright Information

Aras Corporation

100 Brickstone Square

Suite 100

Andover, MA 01810

Phone: 978-691-8900

E-mail: support@aras.com

Website: https://www.aras.com

Table of Contents

Send Us Your Comments

Aras Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for future revisions.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where and what level of detail?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, indicate the document title, and the chapter, section, and page number (if available).

You can send comments to us in the following ways:

Email:
TechDocs@aras.com
Subject: Aras Product Documentation

Or,

Postal service:
Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Attention: Aras Technical Documentation

If you would like a reply, provide your name, email address, address, and telephone number.

If you have usage issues with the software, visit https://www.aras.com/support/

1 Introduction

1.1 Purpose

The Aras DevOps – Aras 3D Visualization Configuration Guide provides information for Aras DevOps subscribers to add the Aras 3D Visualization platform component to their Aras Innovator instance.

1.2 Scope

This document provides instructions for installing Aras 3D Visualization which includes Aras CAD Converter, Aras 3D Viewers, and Aras Dynamic Visualization.

1.3 Target Audience

This document is intended for Aras DevOps administrators responsible for installing and configuring the applications and platform components for the Aras Innovator instance utilizing SaaS.

1.4 Prerequisites

Aras DevOps administrators or contributors must have an installed Aras Innovator instance prior to following the steps outlined in this document. It is recommended that all administrators read the Aras DevOps – User Guide, as it explains how to effectively utilize the Aras DevOps service included as part of the Aras Enterprise subscription to manage Aras Innovator customizations throughout the entire lifecycle of the Aras Innovator implementation.

It is also recommended that all administrators or contributors refer to the latest Aras 3D Visualization Installation Guide to install Aras 3D Visualization to the repository.

2 Definitions

This section defines the terms used in this document.

Term	Definition
Code tree	The set of files that make up the application files for Aras Innovator.
Code tree patch	The set of files in an application installation package that should be applied to the code tree to install the code tree component of an application.
Converter binaries	The set of binaries included in a converter utilized for processing a conversion task by the conversion server (HOOPS, Prince, etc.)
Converters	A toolset included with some applications for executing asynchronous tasks on the conversion server (e.g. CAD to PDF, PDF Watermarking, etc.)
Import package	The set of XML files and the corresponding import.mf file that represent a package or set of packages that need to be installed to apply the database changes of an application.
Installation package	The set of files utilized to install an application that may include a code tree patch, an import package, and/or converters.
Platform Component	Aras Platform Components are not release specific and can be installed without a direct dependency of the installed Aras Innovator release.

3 Aras 3D Visualization Installation

Aras 3D Visualization is a platform component which is installed after Aras Innovator. As a platform component, it can be installed without a direct dependency on the Aras Innovator release. Before installing Aras 3D Visualization, it is important to notify users that Aras Innovator will experience downtime. Additionally, it is important to create backups for the Aras Innovator code tree, Conversion Server, and database. Store these backup files securely, as these files will be necessary for restoring the system to its pre-installation state if installation fails.

Note: This is an example and may not reflect the latest changes to Aras 3D Visualization. To successfully install Aras 3D Visualization into the repository, it is recommended to read and follow the Aras 3D Visualization Installation guide along with this guide, as it provides up-to-date instructions.

3.1 Aras CAD Converter Installation

The following steps outline the process of installing Aras CAD Converter:

1. Download the **Aras 3D Visualization CD Image** from Aras FTP site.
2. Unzip the Aras 3D Visualization CD Image on the local machine.
3. Create a `\CodeTree\ConversionServer\customExtensions\Hoops\` folder in the Work.git repository.
4. In the `\CodeTree\ConversionServer\customExtensions\Hoops\` folder, create a file named `onbuild.sh` with the following content:

```
# Install third-party prerequisites - Hoops
if [ -f /app/customExtensions/Hoops/installHoops.sh ]
then
chmod +x /app/customExtensions/Hoops/installHoops.sh
/app/customExtensions/Hoops/installHoops.sh
fi
```

5. In the `\CodeTree\ConversionServer\customExtensions\Hoops\` folder, create the converter installation script `installHoops.sh` with the following content:

```
# Initial parameters
INSTALLATION_PACKAGE_NAME="HOOPS_Communicator_2023_Linux.tar.gz"
INSTALLATION_PACKAGE_HOOPS_VERSION="HOOPS_Communicator_2023"
INSTALLATION_PACKAGE_FOLDER="/app/customExtensions/Hoops"
SOURCE_HOOPS_TEMPLATES_PATH="$INSTALLATION_PACKAGE_FOLDER/HoopsConvert
erTemplates"

# extract files from archive
HOOPS_ARCHIVE_PATH="$INSTALLATION_PACKAGE_FOLDER/$INSTALLATION_PACKAGE
_NAME"
tar -xf $HOOPS_ARCHIVE_PATH -C /tmp

# remove folder if already exists
TARGET_HOOPS_FOLDER_NAME="/opt/ts3d"
if [ -d $TARGET_HOOPS_FOLDER_NAME ]
then
```

```
rm -rf /opt/ts3d
fi

# create folder for hoops
mkdir $TARGET_HOOPS_FOLDER_NAME

# copy hoops files to
TARGET_HOOPS_BIN_FOLDER_NAME="$TARGET_HOOPS_FOLDER_NAME/bin"
SOURCE_HOOPS_FOLDER_NAME="/tmp/$INSTALLATION_PACKAGE_HOOPS_VERSION/authoring/converter/bin"
cp -ar $SOURCE_HOOPS_FOLDER_NAME $TARGET_HOOPS_BIN_FOLDER_NAME

# add required permissions for binaries
HOOPS_BINARIES_PATH="$TARGET_HOOPS_BIN_FOLDER_NAME/linux64/libboost_*"
chmod +rx $HOOPS_BINARIES_PATH

# copy templates, if were added
if [ -d $SOURCE_HOOPS_TEMPLATES_PATH ]
then
TARGET_HOOPS_TEMPLATES_PATH="/app/HOOPSConverter/Templates"
mkdir -p $TARGET_HOOPS_TEMPLATES_PATH
cp -ar $SOURCE_HOOPS_TEMPLATES_PATH $TARGET_HOOPS_TEMPLATES_PATH
fi

# add required permissions for converter files
if [ -d "/app/HOOPSConverter" ]
then
chmod -R ugo-w,o-rx "/app/HOOPSConverter"
fi

# cleanup obsoleted installation files
rm -rf $INSTALLATION_PACKAGE_FOLDER
rm -rf "/tmp/$INSTALLATION_PACKAGE_HOOPS_VERSION"
```

6. Update the `# Initial parameters` section of the `installHoops.sh` file with the appropriate values:

```
# Initial parameters
INSTALLATION_PACKAGE_NAME="HOOPS_Communicator_2023_Linux.tar.gz"
INSTALLATION_PACKAGE_HOOPS_VERSION="HOOPS_Communicator_2023"
```

Note: Depending on the specific release of Aras 3D Visualization, the necessary Linux binaries may or may not be included in the installation package. If the binaries cannot be located, contact Aras Support (support@aras.com) for assistance.

7. If HOOPS templates are required, create a `\CodeTree\ConversionServer\customExtensions\Hoops\HoopsConverterTemplates\` folder in the Work.git repository.
8. Add the template files.

9. Navigate to the `\Packages\CADConverter\ConversionServer\` folder of the Aras 3D Visualization CD Image.
10. Copy the “bin” folder into the `\CodeTree\ConversionServer\` folder of the Work.git repository.
11. Add a configuration transformation file for the ConversionServerConfig.xml in the `\TransformationsOfConfigFiles\` folder of the Work.git repository with the following content:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration xmlns:xdt="http://schemas.microsoft.com/XML-Document-Transform">
 <configSections xdt:Transform="Replace" xdt:Locator="XPath(/configuration/configSections)">
    <section name="oauth" type="Aras.OAuth.Configuration.OAuthSection, Aras.OAuth.Configuration" />
    <!-- Common converter service configuration -->
    <section name="ConversionServer" type="Aras.ConversionFramework.ConversionServer.Configuration.ConversionServerConfigurationSection, Conversion.Base" />
    <sectionGroup name="ConverterSettings">
       <section name="ArasCadConverter" type="Aras.ConversionFramework.Converter.Hoops.Configuration.HoopsConverterConfiguration, ArasCadConverter" />
       <section name="ArasCadConverterPrc" type="Aras.ConversionFramework.Converter.Hoops.Configuration.HoopsConverterConfiguration, ArasCadConverter" />
       <section name="HtmlPublishingConfiguration" type="Aras.MPP.Publishing.HtmlPublishingConfiguration, Aras.MPP.Publishing" />
    </sectionGroup>
 </configSections>

 <ConversionServer>
    <Converters>
       <Converter name="Aras CAD to PDF Converter" type="Aras.ConversionFramework.Converter.Hoops.HoopsConverter, ArasCadConverter"  xdt:Transform="Insert" />
       <Converter name="Aras PRC to SCS Converter" type="Aras.ConversionFramework.Converter.Hoops.HoopsConverterPrc, ArasCadConverter" xdt:Transform="Insert" />
       <Converter name="mpp_HtmlPublishingConverter" type="Aras.MPP.Publishing.HtmlPublishingConverter, Aras.MPP.Publishing" xdt:Transform="Insert" />
    </Converters>
 </ConversionServer>

 <ConverterSettings xdt:Transform="Replace">
    <ArasCadConverter>
```

```
        <Application converterPath="/usr/bin/xvfb-run"/>
        <Command arguments="'--auto-servernum' '-s' '-screen 0
640x480x24' /opt/ts3d/bin/linux64/converter --
sc_compute_bounding_boxes 'All' --load_all_configurations 'True' --
input_pdf_template_file '/app/HOOPS
Converter/Templates/Blank_Template_L.pdf' --output_pdf
'%filepath%/%filename%.pdf' --output_png '%filepath%/%filename%.png' -
-output_png_resolution '150x150' --output_scs
'%filepath%/%filename%.scs' --output_xml_assemblytree
'%filepath%/%filename%.xml' --output_prc '%filepath%/%filename%.prc' -
-background_color '1.0, 1.0, 1.0' --output_logfile
'%filepath%/%filename%.log'" />
        <AssemblyCommand dynamicEnabled="True" arguments="'--auto-
servernum' '-s' '-screen 0 640x480x24' /opt/ts3d/bin/linux64/converter
--load_all_configurations 'True' --input_pdf_template_file '/app/HOOPS
Converter/Templates/Blank_Template_L.pdf' --output_pdf
'%filepath%/%filename%.pdf' --output_png '%filepath%/%filename%.png' -
-output_png_resolution '150x150' --output_scs
'%filepath%/%filename%.scs' --output_xml_assemblytree
'%filepath%/%filename%.xml' --output_prc '%filepath%/%filename%.prc' -
-background_color '1.0, 1.0, 1.0' --output_logfile
'%filepath%/%filename%.log'" />
        <Output>
            <UploadToVault>
                <File extension="prc" argsMarkers="--output_prc"/>
                <File extension="scs" argsMarkers="--output_scs"/>
                <File extension="pdf" argsMarkers="--output_pdf"/>
                <File extension="png" argsMarkers="--output_png"/>
                <File extension="stl" argsMarkers="--output_stl"/>
                <File extension="xml" argsMarkers="--
output_xml_assemblytree"/>
            </UploadToVault>
        </Output>
    </ArasCadConverter>
    <ArasCadConverterPrc>
        <Application converterPath="/usr/bin/xvfb-run"/>
        <Command arguments="'--auto-servernum' '-s' '-screen 0
640x480x24' /opt/ts3d/bin/linux64/converter --
sc_compute_bounding_boxes 'All' --load_all_configurations 'True' --
output_scs '%filepath%/%filename%.scs' --output_xml_assemblytree
'%filepath%/%filename%.xml' --output_logfile
'%filepath%/%filename%.log'" />
        <Output>
            <UploadToVault>
                <File extension="prc" argsMarkers="--output_prc"/>
                <File extension="scs" argsMarkers="--output_scs"/>
                <File extension="pdf" argsMarkers="--output_pdf"/>
                <File extension="png" argsMarkers="--output_png"/>
                <File extension="stl" argsMarkers="--output_stl"/>
```

```
            <File extension="xml" argsMarkers="--
output_xml_assemblytree"/>
          </UploadToVault>
       </Output>
    </ArasCadConverterPrc>

    <PdfPublishingConverter>
       <ConvertionTool path="/usr/bin/prince"/>
    </PdfPublishingConverter>

    <HtmlPublishingConfiguration>
       <BaseResourceUrls xdt:Transform="Replace">
          <add url="${ADT_INNOVATOR_URL}/Client/images" />
       </BaseResourceUrls>
       <ConvertionTool path="/usr/bin/prince"/>
    </HtmlPublishingConfiguration>

 </ConverterSettings>
</configuration>
```

Note: The **ConversionServerConfig.xml** is available from the Baselines Artifact feed of the SDE for reference when building configuration transformation files.

12. Copy the ArasCadToPdfConverter, com, and PLM folders from the `..\Aras 3D Visualization CD Image\Packages\CADConverter\Imports\` folder into the AML-packages folder of the Work.git repository.

Note: Do not copy the **imports.mf** file, as it will overwritie the existing **imports.mf** file in the Work.git repository.

13. Open the **imports.mf** file.
14. Copy the following `<package>` elements:

```
<imports>
  <package name="hoops_converter" path="ArasCadToPdfConverter">
    <dependson name="com.aras.innovator.conversion" />
  </package>
  <package name="com.aras.innovator.cui_default" path=".\" />
  <package name="com.aras.innovator.solution.PLM" path="PLM" />
</imports>
```

15. Paste the `<package>` elements into the **imports.mf** file of the Work.git repository.
16. Stage and review the changes to ensure that no customizations are overwritten.
17. Commit the changes.
18. Run the "**continuous-integration**" pipeline.
19. Run the "**deploy-innovator**" pipeline to complete the installation.

3.2 Aras 3D Viewers Installation

The following steps outline the process of installing the Aras 3D Viewers:

1. In the unzipped Aras 3D Visualization CD Image on the local machine, go to the `\Packages\3DViewers\` folder.
2. Copy the contents of the "**Innovator**" folder into the `CodeTree\Innovator\` folder of the Work.git repository.
3. Copy the Aras3DViewers and com folders from `the ..\Aras 3D Visualization CD Image\Packages\3DViewers\Imports\` folder into the AML-packages folder of your Work.git repository.

   Note: Do not copy the **imports.mf** file. Doing so will result in overwriting the existing **imports.mf** file of the Work.git repository.
4. Open the **imports.mf** file.
5. Copy the following <package> elements:

```
<imports>
  <package name="aras_3dviewers" path="Aras3DViewers">
    <dependson name="com.aras.innovator.ssvc" />
    <dependson name="hoops_converter" />
  </package>
  <package name="com.aras.innovator.cui_default" path=".\" />
  <package name="com.aras.innovator.ssvc" path=".\" />
  <package name="com.aras.innovator.viewers" path=".\" />
 </imports>
```

6. Paste the `<package>` elements into the **imports.mf** file of the Work.git repository.
7. Stage and review the changes to ensure that no customizations are overwritten.
8. Commit the changes.
9. Run the "**continuous-integration**" pipeline.
10. Run the "**deploy-innovator**" pipeline to complete the installation.

3.3 Aras Dynamic Visualization Installation

The following steps outline the process of installing the Aras Dynamic Visualization:

1. In the unzipped Aras 3D Visualization CD Image on the local machine, go to the `\Packages\DPN\` folder.
2. Copy the contents of the "**Innovator**" folder into the `CodeTree\Innovator\` folder of the Work.git repository.
3. Add a configuration transformation file for the \Innovator\Server\method-config.xml in the `\TransformationsOfConfigFiles\Innovator\Server\` folder of the Work.git repository with the following content:

```
<MethodConfig xmlns:xdt="http://schemas.microsoft.com/XML-Document-Transform">
  <ReferencedAssemblies>
    <name id="Aras.DynamicModelViewer.Core"
xdt:Transform="InsertIfMissing"
xdt:Locator="Match(id)">$(binpath)/Aras.DynamicModelViewer.Core.dll</name>
```

```
    <name id="Aras.DynamicModelViewer.DataModel"
xdt:Transform="InsertIfMissing"
xdt:Locator="Match(id)">$(binpath)/Aras.DynamicModelViewer.DataModel.d
ll</name>
    <name id="Aras.DynamicModelViewer.QueryProcessor"
xdt:Transform="InsertIfMissing"
xdt:Locator="Match(id)">$(binpath)/Aras.DynamicModelViewer.QueryProces
sor.dll</name>
  </ReferencedAssemblies>
</MethodConfig>
```

Note: The \Innovator\Server\method-config.xml is available from the Baselines Artifact feed of the SDE for reference when building configuration transformation files.

4. Copy the **DynamicModelConstructor andDynamicModelConstructor_default** folders from the `..\Aras 3D Visualization CD Image\Packages\DPN\Imports\` folder into the AML-packages folder of the Work.git repository.

Note: Do not copy the **imports.mf** file. Doing so will result in overwriting the existing **imports.mf** file of the Work.git repository.

5. Open the `dynamic_model_constructor_imports.mf` file.
6. Copy the following `<package>` elements:

```
<imports>
  <package name="dynamic_model_constructor"
path="DynamicModelConstructor">
    <dependson name="com.aras.innovator.ssvc" />
    <dependson name="hoops_converter" />
  </package>
  <package name="dynamic_model_constructor_default"
path="DynamicModelConstructor_default">
    <dependson name="dynamic_model_constructor" />
  </package>
 </imports>
```

7. Paste the `<package>` elements into the **imports.mf** file of the Work.git repository.
8. Stage and review the changes to ensure that no customizations are overwritten.
9. Commit the changes.
10. Run the “**continuous-integration**” pipeline.
11. Run the “**deploy-innovator**” pipeline to complete the installation.