ALTAIR

ONLY FORWARD

Altair Accelerator Software 2024.1.0

Installation Guide

Contents

Altair Accelerator Software Installation

The Altair Accelerator software product installation guide describes how to install Altair Monitor, Altair Accelerator, Altair Allocator, Altair FlowTracer, and Altair Accelerator Plus.

This chapter covers the following:

All these products share the same installation files and the same general install instructions. Product specific variations are noted in the install instructions as they are needed.

> 📝 **Note:** If installing Altair Accelerator products in a Windows environment, do not use PowerShell as it is not supported.

Installation Scenarios

There are four types of installation scenarios you might perform.

1. Installing an Altair Accelerator product for the first time.

2. Updating a release of an Altair Accelerator product to add support for a new platform (machine type).

3. Updating a release of an Altair Accelerator product to install a patch.

4. Upgrading to a new release of an Altair Accelerator product.

This guide will assist you through the installation process for each of these scenarios.

General Installation Steps

For UNIX class machines, the installation process follows a straightforward series of steps:

1. Download the tar files to a temporary staging area.

2. Expand the tar files in the temporary staging area.

3. Run a script from the temporary staging area to put the files into the production install area.

4. License the production programs.

5. Run programs from the installed production area in order to validate that the product installed correctly.

The installation process is different for Windows machines. See Windows Installation Overview for details.

ALTAIR

Download the Files

You can download the necessary installation files from the Altair Marketplace. You must have an Altair One account to access the files.

1.   From the Altair One Marketplace, use the upper left search tool to find the product to download.

2.   Click **Download** to download the product.
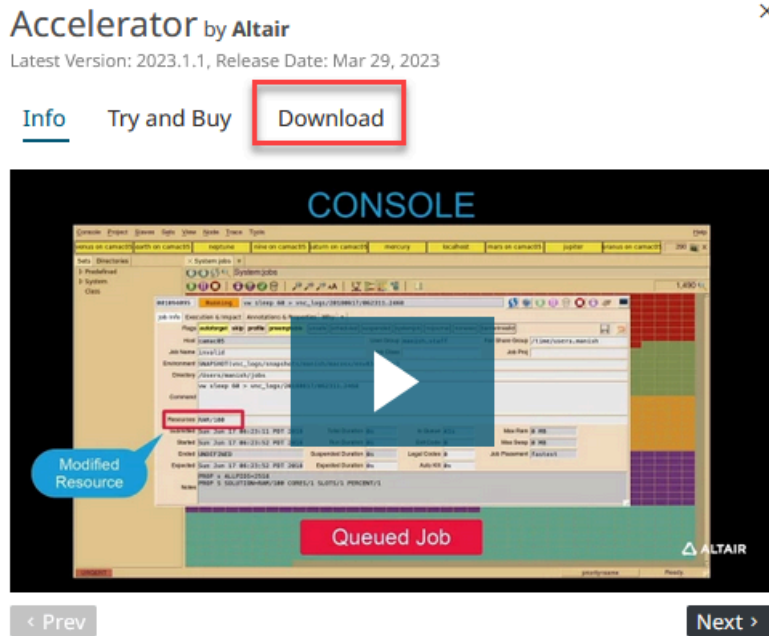
Figure 1:

3.   In the panel that appears on the right side of the screen, select the **Suite Version**, **Language** and **Operating System** to use. A sub folder of updates and patches opens. You only need to download and install the files for your desired release and scenario.

> 📝   **Note:** You may need to click **Show more...** to see the entire list of packages.

Types of Downloads Available

Target Platforms

To perform an installation, download the required installation files from the Altair web site. One installation file is a tar file called `Common.tar`. You need to install this in every case.

Additional tar files are named by platform (machine type). You will need to download the platform-specific installation tar file for each platform you will use. These are your target platform files.

The first target platform you will consider is the one that is your main server machine, where the installed product will run. Download the tar file for this platform.

If you are using the product across a network, you will also be thinking about the platforms for each remote systems in your network. The remote systems will access their executables from the main server system, using the Network File System (NFS) and auto-mounting to access them with a local path. The installation process on the server needs to install executables for the other platforms so that the networked remote systems will have access to the executables that they will run.

Download the tar file for each platform used by your networked system.

Single File Distributables

The Single File Distributables directory contains files used for special situations. These files are used for irregular occurences; for example, when you are using the product to dispatch programs on a remote machine that does not provide an SSH or RSH interface to use by the central dispatcher to start programs. In this situation, there is a requirement to manually start a helper program on the remote tasker hosts. You get the helper programs in the Single File Distributable section.

These files are not needed for any of the basic installation scenarios. They are only needed for specific cases of expanding a network of tasker hosts where an added tasker host does not provide SSH or RSH access. When this occurs, you will find information in the product Administrator's manual about needing to download specific files to get around the lack of SSH and RSH access, and how to install and configure the downloaded files. This website section of Single File Distributables are the files that are used in those instances.

There are three types of Single File Distributables:

- Monitor Full
- Monitor Agent
- Tasker Service Daemon

The Monitor full file is the version of the Monitor that is described in *Server Operations* in the Altair Monitor Administrator Manual.

The Monitor Agent file is the version of the Monitor that is described in *Network Monitoring* in the Altair Monitor Administrator Manual.

The Tasker Service Daemon file is the download file that includes the `vovtsd` program used to run programs on remote machines that do not allow SSH or RSH access to start programs. The `vovtsd` program is run on the remote tasker host and interacts with the central dispatcher to field requests to run programs on the machine it is on. This is described in *Manage Remote Taskers without SSH/RSH Capabilities* in the Altair Accelerator Administrators Manual.

Documentation Files

You can access the Altair Accelerator documentation through several routes.

- After selecting a product in the Marketplace, the right hand panel lists the documents available for the selected release.
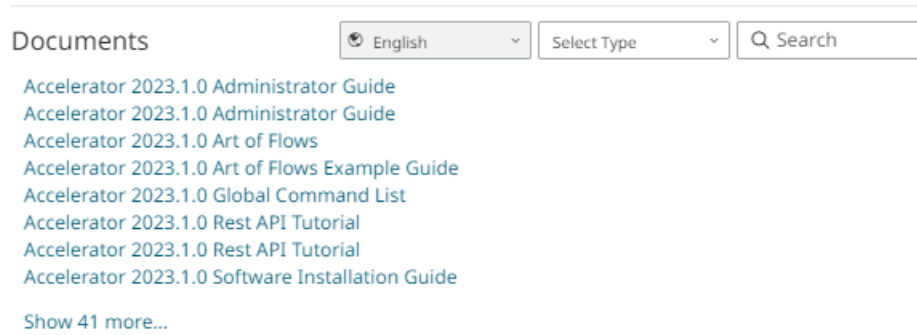
Figure 2:

- From Altair Community, you can view, filter and sort through the most recent Accelerator product guides.

- Access the help from within the product installation (`/doc` directory).

The Documentation directory contains the PDF product help files for all the products. They are available to download.

The *The Art of Flows* by Andrea Casotto and Timothy Barnes is available as a free ebook download here. This book is the definitive explanation about the dynamics of managing a complex job mix such as ones used in the chip design process. Files, programs, and their dependencies are the fundamental issues for scheduling and deploying jobs in any design-oriented activity. This volume explains the importance of the problem, and how best to solve it.

This book is highly recommended as a practical guide to the value of using Altair Accelerator products with its clear introduction to dependency graphs and the importance of run time discovery of input and output files.

Advanced Commands

Advanced commands are available for more detailed operations. For information about the Altair Accelerator family of commands, refer to the Global Command List.

First Time Installation

This first time installation scenario installs a new set of files to implement the product features.

The same installation process is used for each of the products.

- Altair Accelerator
- Altair Monitor
- Altair FlowTracer
- Altair Allocator
- Altair Accelerator Plus

Each product is controlled by either an Altair License Manager license (floating or node-locked) or a legacy node-locked license. Obtaining and installing a license are steps in the installation explained here.

To proceed with the installation, you need to:

1. Follow the installation planning directions contained in the next section.
2. Download the software to install from the Altair website, with guidance from the download information contained in a later section.
3. Perform the initial software installation, which should require between 15 and 30 minutes depending on disk performance.
4. Obtain license keys for running the software.
5. Validate the installation by setting your command line environment and running the validation steps.

The following sections will lead you through these steps.

Plan the Installation

This section contains the system requirements and a set of preparation checklists that should be followed before installing the software. The checklists include a common section, followed by product specific sections.

Expected System Environment

The Altair Accelerator products are built to run on a well-conditioned network. This outline gives an overview of what is expected to be in place.

- All participating hosts are networked.
- The host clocks are well-synchronized (e.g. via NTP).
- Naming services are functional and fast
    - # User/Group account names and UIDs are uniform on all machines
    - # Resolution of *hostname* works (DNS or NIS).
    - # Reverse DNS lookup works (get host name from IP address).
- There is a remote-shell capability for at least the product administrator's account to all participating machines. If not available, use the example `.bat` files as a guide to create a script, and place it in the appropriate directory. Example startup files are provided in `$VOVDIR/etc/boot`. Choose the one that best fits your scenario.
- There are networked file systems on which to place shared files, so that they can be accessed from each machine, using a local path enabled by the NFS. The shared files include:

    \#   Installed Altair Accelerator software, including at least one branch that must be writable (the *local* directory).

    \#   The Server Working Directory (.swd)

    \#   The users' home directories.

    \#   Work area for each project.

- Standard directories, binaries and libraries are available.

    \#   `/bin/sh`, or `/bin/bash` on Linux

    \#   `/bin/csh`, or `/bin/tcsh`

    \#   `/tmp` exists

    \#   `/usr/tmp` exists (not on Ubuntu by default)

    \#   gcc libs are installed (libstdc++, libfcc_s)

System Configuration Hardware Guidelines

Use the following tables to provision your machines to support your projected usage.

Accelerator and Allocator Hardware Guidelines

Resource	Jobs		
	100,000	**1,000,000**	**>1,000,000**
CPU (x86, e.g. Intel Xeon)	Dual Core Server Class	Quad Core Server Class	Quad Core Server Class
Memory	2 GB	16 GB	32 GB
Disk	5 GB	50 GB	200 GB
Server Working Directory	NFS or Local	NFS or Local	Local

Monitor Hardware Guidelines

Resource	Checkouts		
	100,000	**1,000,000**	**>1,000,000**
CPU (x86, e.g. Intel Xeon)	Dual Core Server Class	Quad Core Server Class	Quad Core Server Class
Memory	1 GB	8 GB	16 GB
Disk	5 GB	50 GB	200 GB
Server Working Directory	NFS or Local	NFS or Local	NFS or Local
Database	Local Disk	Local Disk	Local Disk

FlowTracer Hardware Guidelines

Resource	Seat Licenses			
	1	10	100	>500
CPU (x86, e.g. Intel Xeon)	Single/Dual Core Server Class	Dual Core Server Class	Dual Core Server Class	Quad CoreServer Class
Memory	1 GB	1 GB	8 GB	64 GB
Disk	5 GB	50 GB	100 GB	200 GB

Accelerator Plus Hardware Guidelines

Resource	Jobs		
	100,000	1,000,000	>1,000,000
CPU (x86, e.g. Intel Xeon)	Dual Core Server Class	Quad Core Server Class	Quad Core Server Class
Memory	2 GB	16 GB	32 GB
Disk	5 GB	50 GB	200 GB
Server Working Directory	NFS or Local	NFS or Local	Local

Installation Checklists

The checklists will help you prepare for installing and configuring the software.

General Installation Checklist

If you are performing an evaluation, complete the items in the **EVAL** column. If you are installing a production system, complete the items in the **PROD** column.

EVAL	PROD	Description
☐	☐	Obtain a user account Altair Marketplace for downloading the software.
☐	☐	Calculate the amount of free disk space needed for the installation and operation of the system. Use the tables in Plan the Installation showing disk space requirements.
		The disk storage for Accelerator and FlowTracer must be shared and accessible by all machines via NFS and/or Windows Networking, as applicable. It must be possible to set permissions on files and folders and to create files which have the correct user ID.

△ **ALTAIR**

EVAL	PROD	Description
		Accelerator requires the ability to execute a `setuid` application from shared storage.
☐	☐	Select a product administrator user who will own the installed files and the running processes. This product administrator user account can be the same as one used for managing other tools. ⚠️ **Important:** This product administrator user account cannot be the UNIX root user. It should be a normal user without special privileges or capabilities.
☐	☐	Select the networked file server where the software will be installed. Use the information as a guideline for selecting the machine with sufficient disk space.
☐	☐	Select the server machine where the software will run. Use the System Configuration Hardware Guidelines for selecting the server machine with sufficient capacity.
	☐	To activate the email notifications feature, you will need to configure an SMTP outbound email server after doing the install. Gather information about your SMTP outbound email server to prepare for this configuration step. You can read about the Notification feature and configuring SMTP in Connect to an SMTP Server.
	☐	To activate the LDAP server feature for authentication of users, and for defining special groups of users, you will need to configure access to the LDAP server after doing the install. Gather information about your LDAP server to prepare for this configuration step. You can read about the feature of using an LDAP server, and Configuring LDAP in Connect to an LDAP Server.
☐	☐	Ensure that the system clocks (including the license server if applicable) are correct on all machines and synchronized with a time server.
☐	☐	Designate a server machine to use for the Reprise license server (required for FlowTracer, optional for Monitor and Accelerator). The Reprise license server can be the same machine as the Monitor server.

☐	☐	Submit a support request at Altair Community for a license file. Provide the name of your company, host name and host ID for the license server or for the server which will run the software.
		
		On Linux and Windows, the host ID is the Ethernet MAC address of the primary network interface card. Use the following commands to get the host ID:
		
		<pre>% /sbin/ifconfig -a # Linux - use
		 eth0
		c:\> ipconfig /all # Windows</pre>

△ALTAIR

EVAL	PROD	Description
		📝 **Note:** Do not use the Linux `hostid` command, which returns an incorrect result.
	☐	Increase the number of open file descriptors on the server you are planning to use. You can read about calculating the number of needed file descriptors in the Client Limitation and Tuning chapter.
☐	☐	Follow the Product Specific checklist. If you are installing Accelerator to run applications for which you have a limited number of licenses, you must follow the Monitor checklist as well.

Product Specific Checklists

Monitor Installation Checklist

EVAL	PROD	Description
☐	☐	Gather a list of license servers (port@host) to be monitored.
	☐	If interested in monitoring denial information, read *Debug Log Monitoring* in the *Altair Monitor Administration Guide* to determine an appropriate strategy for your site. Gather the list of debug logs to be monitored.
☐	☐	Be sure all FlexNet Publisher license files specify a port on the SERVER line.

Accelerator (and Allocator) Installation Checklist

EVAL	PROD	Description
☐	☐	Determine the number of Accelerator instances you will have running.
☐	☐	Identify the number of farm machines (tasker hosts) you will have.
☐	☐	Identify the number of processors per machine.
☐	☐	Determine if you use licensed tools in jobs that Accelerator will manage. This is typically a yes, which leads to choosing how to manage the licenses - either the built-in Monitor-basic or ControlCenter.
☐	☐	Identify the license server machine. If using a keyfile, it will be the same machine as the one running Accelerator.

△ ALTAIR

EVAL	PROD	Description
☐	☐	Identify which desired features will be used and note the auxiliary daemons that are needed to provide each feature: • Enable optimized resource management. (`vovresourced` daemon) • Enable email notification about events. (`vovnotifyd` daemon) • Enable higher priority jobs to run first. (`vovpreemptd` daemon) • Enable reconciliation of mismatched resource requests. (`vovreconciled` daemon)

Accelerator Plus Installation Checklist

EVAL	PROD	Description
☐	☐	Determine the number of Accelerator Plus instances you will have running.
☐	☐	Identify the number of farm machines (tasker hosts) you will have.
☐	☐	Identify the number of processors per machine.
☐	☐	Determine if you use licensed tools in jobs that Accelerator Plus will manage. This is typically a yes, which leads to choosing how to manage the licenses - either the built-in Monitor-basic or ControlCenter.
☐	☐	Identify the license server machine. If using a keyfile, it will be the same machine as the one running Accelerator Plus.
☐	☐	Identify which desired features will be used and note the auxiliary daemons that are needed to provide each feature: • Enable optimized resource management. (`vovresourced` daemon) • Enable email notification about events. (`vovnotifyd` daemon) • Enable higher priority jobs to run first. (`vovpreemptd` daemon) • Enable reconciliation of mismatched resource requests. (`vovreconciled` daemon)

FlowTracer Installation Checklist

EVAL	PROD	Description
☐	☐	If installing in a multi-site environment, its best to ensure that the installation looks identical from a file system point of view at each site. One site may be the

△ ALTAIR

EVAL	PROD	Description
		master where all modifications/installations are made, while the other sites are mirrored from the master site.
		Identify a robust, high performance file system to host the registry directory used by all FlowTracer users. This could be a Tier 1 filer or even an SSD. It is also recommended that for multi-site installations, an independent registry file system exist at the same logical location at each site, that is, `/proj/ registry`. It is not necessary to procure a large disk, 10GB-20GB will be more than sufficient for even the largest enterprises.
		It is recommended to enable Enterprise Licensing and set it to Auto.

Download the Software

After the pre-installation checklist is complete, you are ready to download the software and prepare the files in order to do the install.

Establish an Altair Accelerator Product Administrator User Account

It is recommended that you establish an Altair Accelerator Product Administrative user, or *role account*, on your computer. For example, define the user account accel_mgr, for installing and owning the Altair Accelerator software files.

This user is a normal user with no special privileges. The files owned by this user are shared by way of having the same user set up with the same name and UID on all the machines.

> ⚠ **Important:** This user should not be the UNIX root user. Doing so would be a badly designed way to share files that conflicts with best practices of UNIX security policy.

You can share the Altair Accelerator Product Administrative user account with other tools. For example you can use Cadence's recommended user account cdsmgr to also be the Altair Accelerator Product Administrative account.

For evaluations, you can use your own personal account as the Altair Accelerator Product Administrator, and use it do the product install.

In the following sections, examples will be written as if you will perform the installation as user accel_mgr. For Accelerator production installations, one step in the procedure must be performed by the root user.

Create or Reuse a Temporary Download Staging Area

Create a staging area to hold the downloaded files. It must be different from the directory where you plan to install the Altair Accelerator software. The staging area may be removed after installation.

For example, create the staging area directory as user accel_mgr.

a)   On UNIX:

```
% mkdir /usr/tmp/altair_download
```

b)   On Windows:

```
C:\> md  c:\temp\altair_download
```

> 📝   **Note:** Location path names must not contain any spaces.

Download and Expand the Files

1.   Access the Altair Marketplace and select the product to install.

2.   Log in using the account you set up as part of the pre-installation planning steps.

3.   Click the link for the release version you want, and download the appropriate files. Because this is a first install, download both the documentation files and the main installation files.

   The documentation files includes a Release Notes file. This contains high level descriptions about what is in this release.

   The first directory level contains a file called README-1st.txt. This is a text file that describes the files in the release and gives a high level overview of performing an installation.

4.   For a UNIX installation, complete the steps below:

   a)   Download the Common.tar file into the staging directory you just created.

   b)   Download the platform-specific file(s) for each platform type that you will be using.

   c)   Expand the archives with tar extract.

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf <platform>.tar    # For each platform
```

   For example:

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf linux.tar
```

5.   For a Windows installation, complete the steps below:

   a)   When Windows is one of the platforms used in a networked multi-platform situation, you should download the win64.tar file into the staging area.

> 📝   **Note:** The name "win64" is a generic name for all versions of 64-bit Windows.

   b)   Expand the archives with tar extract:

```
% cd /usr/tmp/altair_download
% tar xf win64.tar
```

△ ALTAIR

c) Expand this zip file into the staging area using Windows Explorer (via right-clicking on the file) or any utility that works with zip files.

Install the Software

Create the Installation Directory

Because this is the first installation of the Altair Accelerator software, you need to create a directory that will hold Altair Accelerator software, including this release and future updates and upgrades. It is a convention to give the top level install directory a name based on Altair, to make it easy to understand what is in it.

In the examples here, the top level Altair Accelerator software install directory will be named `/opt/rtda`.

The recommended naming scheme is to use the `altair` name in the top level directory where the installation is, and then use the release name as the subdirectory name to hold the installed files for the particular release:

`/opt/altair/vov/1212.1.0`

where `1212.1.0` is the product release name. For example:

`/opt/altair/vov/2023.1.1`

> 📝 **Note:** Location path names must not contain any spaces.

The directory naming model uses a subdirectory under the top level for each release, named after the release. It will hold the installed files for that particular release. As other releases are installed later, they can be installed in peer subdirectories to this release, with their own release name.

See File System Layout, for a full description of the directory structure of Altair Accelerator files.

The naming convention is suggested to avoid a mistaken practice of using the platform as part of the top level path name. The platform-specific files will be placed within a release with subdirectory names related to the platform. The distinction of "platform" is at the innermost area of the directory structure, with the "release" distinction at the outer level.

The software is normally installed on a file server machine within the network, and is auto-mounted on the other machines so that the files are accessible as a local path on the other machines.

The expectation is that User and Group account names and UIDs are the same on all the machines in the network. Even though the installed files are owned by a user on the file server machine, the ownership will be shared by a user with the same name from another machine. This reduces the conflicts and problems related to shared files across the network.

Installation via the Graphical User Interface

The GUI installation procedure is implemented with a single dialog, where you enter information that tells the install program what to do.

Start the Installer Graphical User Interface for your platform as shown below. The GUI installer uses the X-Windows technology for the Graphical User Interface. If you are running on a remote UNIX system in X-windows, be sure your *DISPLAY* environment variable is set properly.
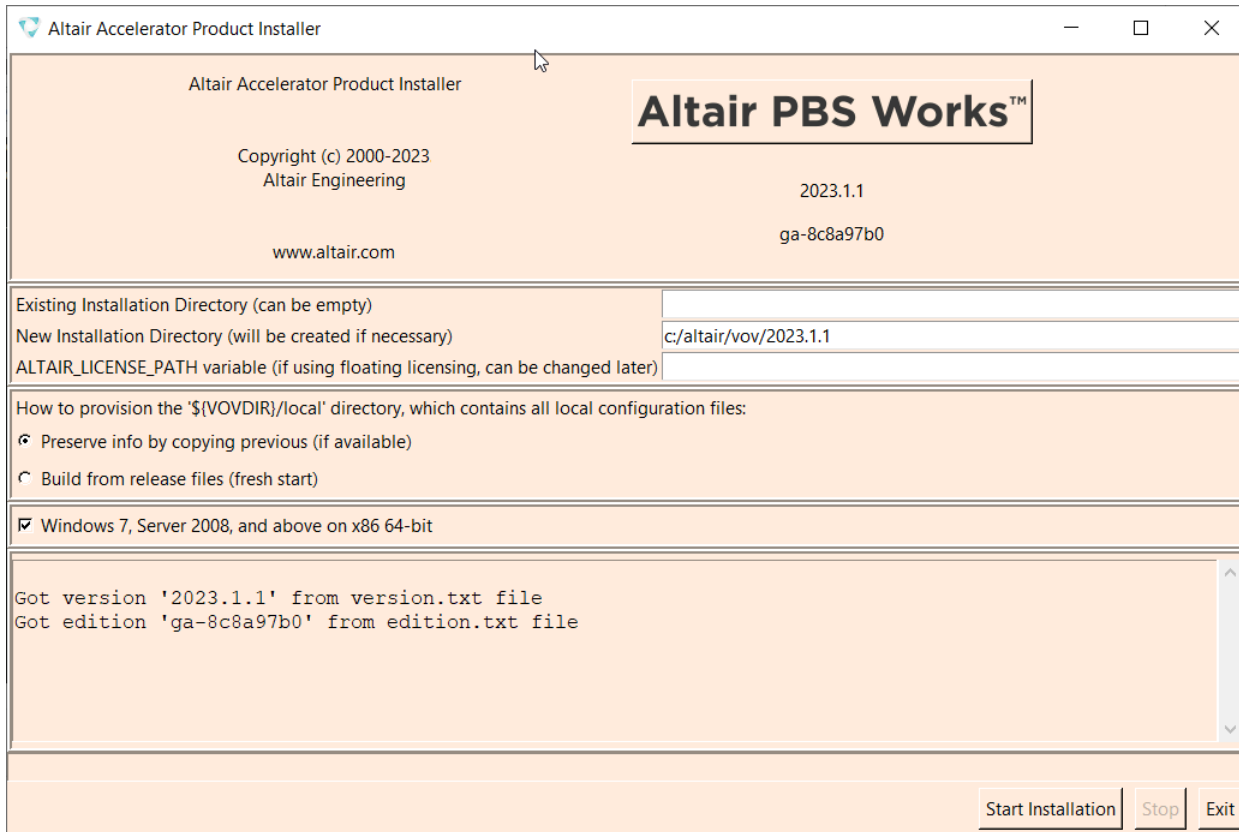
Figure 3:

1. Change the working directory to the staging directory where you downloaded the tar files and extracted them, and run the GUI install script `install.sh`.

   a) On UNIX:

   ```
   % cd /usr/tmp/altair_download
   % ./install.sh
   ```

   b) On Windows:

   ```
   C:\> cd  c:\temp\altair_download
   C:\> install.bat
   ```

   The installation dialog appears. The first field shows the path to the current release if one exists. This should be empty because this is the first time Altair Accelerator products have been installed.

2. In the **New Installation Directory** field, enter the full path to the desired location.
   The new installation directory can be created automatically, if necessary.

3. Set the ALTAIR_LICENSE_PATH variable field to the license server location.

   The value of ALTAIR_LICENSE_PATH can be specified in one of two formats:

   • As port@host (recommended): `<port>@<host>`

   • As a full path to the license file:

   ```
   $VOVDIR/local/almlicense.lic
   ```

4. In the Local Directory Option, select **Build from release files (fresh start)**.

    This tells the installer to make new files within the `common/local` directory and to initialize them with files from within the baseline release.

    Managing the local directory when doing an install is an important topic for later when you do an upgrade. You can read more about it Local Directory.

5. Select one or more platforms on which you wish to install.

    This directs the installer to place the binary executables for these machine types into the installed directory structure. The files will be placed within subdirectories named for the platform. The installed files for each platform will come from the staging area where you have downloaded and extracted them.

    The installed files for each platform will come from the staging area where you have downloaded and extracted them. You should only select platforms here that match the platform-specific install files you have downloaded from the website and extracted into the staging area.

    If you are installing on a UNIX system, you could select the Windows platform here if the Windows file (`win64.tar`) was extracted into the staging area. The Window binaries will be installed on the UNIX machine as files within the win64 platform subdirectory.

    The intention of platform selection is to provide executable files in the file system on this machine, for each platform in the network. This enables each remote machine to access the executables of their type by way of NFS, without needing to do an install of files on any other machines.

6. Click **Start installation** to start the installation. It normally takes a few minutes to run, depending mostly on the speed of your disk.

    During the install, the program will write messages into the bottom section of the dialog, which is a scrollable region. The messages are also written to a log file on disk in the new installation directory, named `install<timestamp>.log`.

7. When the installation is done, click **Exit**.

8. You can review the install log for errors and warnings. The last few sentences of the log file remind you to do something if you are installing Accelerator. This subject is explained in *For Accelerator Installation Only*.

Installation via a Batch Script

You can install the product via a batch script, if desired.

> (!) **Important:** If running on Linux, use `install.sh`. For Windows versions, use `install.bat`.

Run the command `./install.sh`

The values that control the installation are passed into the batch script as command line parameters, except for the *Existing Installation Directory*. The batch script gets the value for this from the $VOVDIR environment variable as it runs.

Only the *New Installation Directory* value is a required parameter. The batch install uses a default value for the other fields.

Because this is a new install, the option -local create should be used for specifying the handling of the "local" directory.

Here is the logical description of calling the batch install for the scenario of installing for the first time.

```
% cd /usr/tmp/rtda_download
```

△ ALTAIR

```
% ./install.sh //opt/altair/vov/1212.10 -local create -rlm <port@host> -plat
  '<platform list>'
```

Here is a specific example of installing for the first time that compares to the example used in the GUI dialog example above.

```
% cd /usr/tmp/rtda_download
% ./install.sh //opt/altair/vov/1212.10 -local create -rlm 7070@buffalo; -plat
  'linux'
```

install.sh

This script supports batch installation on Linux as well as Windows. Run from the directory where the installation files were extracted.

By default, the installer will run in graphical mode. Use the -batch option to suppress the GUI and run in batch mode.

Usage

```
./install.sh [INSTALLPATH][OPTIONS]
```

Options

Use the options below for non-interactive installation.

-alm PORT@HOST	Value to use for ALTAIR_LICENSE_PATH.
-batch	Install in batch mode. If INSTALLPATH is not passed as an argument, the default installation path for Linux is /opt/altair/vov/1212.10 and for Windows is \opt\altair \vov\1212.10.
-local HANDLING	Specify handling of the 'local' directory where HANDLING is one of: - copy (copy previous installation's local) - create (create new from installation stubs) - link (link to previous installation's local) - linkext (link to a version-independent local) The default handling for Linux is linkext, which results in the local directory being placed in the parent of the installation and a symbolic link created inside the installation to point to it. This facilitates the reuse of the local directory for all installed versions. The default handling for Windows is copy. Only copy and create are supported on Windows.
-nocommon	Used when installing additional platforms after an initial installation has been performed, to prevent overwriting of the common directory. Ignored on Windows.
-noconfirm	Do not prompt for confirmation to begin the installation process when running in batch mode.

-platforms LIST   A quoted, space-separated list of platforms to install. Supported platforms are: armv8, linux64, and win64.

-previous PATH   Path to previous installation, if applicable. Defaults to the value of $VOVDIR if set in the calling environment. Otherwise, defaults to an empty string.

Examples

```
./install.sh /opt/altair/vov/<version>
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv -batch
./install.sh /opt/altair/vov/<version> -local create -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -local link -platforms 'linux64 win64'
```

Because this is an update install to add a new platform to a previously installed release, the options -nocommon and -local copy should be used. The `copy` choice for the local parameter will cause the process to copy the local files onto themselves, creating a non-destructive no-op operation.

Here is the logical description of calling the batch install for the scenario of adding platform-specific files for a new platform.

```
% cd /opt/altair/vov/2023.1.0
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy <port@host> -platforms
  '<added-platform>'
```

Here is a specific example of doing an update by running the batch install to add the platform Linux to the release.

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy -alm 6200@almsrv; -
platforms 'linux64'
```

Sample Module - Software Environment Management

If your company uses the "Modules" software from http://modules.sourceforge.net/ to manage the environment for the installed tools, the installation script also generates a module file named `vov.modulefile` to copy into one of the directories of your MODULEPATH where other tool environments are stored.

For example:

```
% mkdir -p /tools/modules/3.2.8a/Modules/modulefiles/vov
% cp /opt/altair/2023.1.0/common/etc/vov.modulefile /tools/modules/3.2.8a/Modules/
modulefiles/vov/2023.1.0

% module avail
---------- /tools/modules/3.2.8a/Modules/modulefiles ----------
vov/2023.1.0      vov/2022.1.1
```

ALTAIR

For Accelerator Installation Only

Accelerator requires one extra step during the install. This step creates a set_UID executable that Accelerator uses so that jobs can run under the ID of the user who submits them. This step requires root capability to have it run correctly.

If you do not have root privileges, or if you are performing an evaluation, you can postpone this step. You can still run the software in single user mode, but you will not be able to run it correctly with multiple users due to permission issues.

You must be logged in as root or have *sudo* privileges to perform this step correctly to create the `vovtaskerroot` binary for each installed platform.

> **Note:** You must run the `SETTASKERUID.sh` script exactly as shown in the next example. The script will not work correctly unless it is executed from the specified directory.

The following script creates a set-UID executable called vovtaskerroot so that jobs can be run under the ID of the user who submits them. This script also sets the set-UID bit in `vovgetpasswd` for Linux.

To complete the installation, you need to login as root and install the binary for `vovtaskerroot` by means of the following commands:

```
% /bin/su -
# cd <install_path><version>
# sh ./scripts/SETTASKERUID.sh
# exit
```

For example:

```
% /bin/su -
# cd /opt/rtda/2020.1
# sh ./scripts/SETTASKERUID.sh
# exit
```

At this point, installation is complete.

Install Third Party Software: FlexNet Publisher Utilities: lmutil/lmstat

If you are installing FlowTracer, Accelerator or Monitor and need to either monitor licenses managed by FlexNet Publisher through Monitor, or use FlowTracer or Accelerator to execute applications that use FlexNet Publisher for license management, you will need to be able to execute the FlexNet Publisher `lmutil` utility, which is used to gather license utilization status.

This means you will need to install `lmutil` to provide support for this feature.

1. Get the latest version of `lmutil` for the operating system that matches that of the Monitor server.

    > **Note:** Some FlexNet Publisher license daemons may require specific versions of `lmutil`.

2. Install `lmutil` under `$VOVDIR/bin` so it can act as the default binary to use when a binary is not specified in the configuration.

△ ALTAIR

$VOVDIR is the directory within your Altair Accelerator application installation that holds the files that are executables that run on the server platform.

```
% cd $VOVDIR/bin
% cp <path>/lmutil .
% chmod a+rx lmutil
% ln -s lmutil lmstat
% ln -s lmutil lmremove
```

For example:

```
% cd $VOVDIR/bin
% cp /tools/flexlm/bin/lmutil .
% chmod a+rx lmutil
% ln -s lmutil lmstat
% ln -s lmutil lmremove
```

Set Up Environment Variables

In order to run the software for validation, you need to set up your command line environment properly.

This includes changing your PATH environment variable so you can run the installed executables, and adding two environment variables that are used by the Altair Accelerator programs.

Set your command line environment by sourcing a setup file created by the installation. You will source the setup file that matches your platform and shell. As earlier, we are using the path `/opt/rtda/<release>` in our examples as the location of the installation.

Platform Type	Shell	Command to Source File
UNIX	csh tcsh	source `/opt/rtda/2020.1/common/etc/vovrc.csh`
	bash sh ksh zsh	`. /opt/rtda/2020.1/common/etc/vovrc.sh`
Windows	DOS	`c:\rtda\2020.1\win64\bat\vovinit.bat`

Each one of these files defines the variable VOVDIR, which will be used in the rest of the installation documentation and which must be set for proper operation of Altair Accelerator software.

The environment variable VOVDIR always points to the platform specific directory, so that, for example, the executables can be found in `$VOVDIR/bin`, the documentation can be found in `$VOVDIR/doc`, and the local information can be found in `$VOVDIR/local`.

ALTAIR

Verify the Installation

Install the License

Before you can verify the installation, you will need to install the proper licensing. Follow the instructions in Licensing to enable execution of Altair Accelerator software.

Quick Verification

After completing the installation and setting up your shell environment, you can do a quick verify that the install was successful by running a program that reports on the installed version. This confirms that the installed program is accessible on your PATH and that it is the version you expect.

Use the `vovversion` program to confirm you are accessing the desired version of the product.

Running this command should give you the expected version number, and the expected location where you have installed the product.

It is also a good practice to check that your environment variables have been set up correctly:

```
> source /<install_path>/<version>/<platform>/common/etc/vovrc.csh
% vovversion -all
2020.1
VOVDIR=/opt/rtda/2020.1/linux
```

You should see the expected version number, and the expected location where you have installed the product.

vovversion

Print the VOV version. Optionally show more details.

```
vovversion: Usage Message

DESCRIPTION:
    Print the VOV version. Optionally show more details.

USAGE:
    vovversion [options]

OPTIONS:
    -help      -- Print this message
    -all       -- Print all available version information
    -clients   -- List clients and their version numbers
    -install   -- Print the VOV version with patch names, if any
    -patches   -- Print patch names, if any
    -project   -- Print the version of vovserver associated with the
                  current project
    -short     -- Print VOV version with short patch name, if any
    -verify    -- Ensure server and client versions match

    [Can abbreviate options provided they are unambiguous]
```

△ ALTAIR

Visual Verification

You can navigate around the install area to make sure that there are subdirectories for each platform that you requested to be installed.

When you set up your environment in the previous section, you set the environment variable VOVDIR to be the path to your platform specific files. You can navigate to that area of the installed files to look around to see the other platform files.

If you only specified one platform - the one you are on - then there is nothing to see other than the executable files for your platform that are in `$VOVDIR/bin`.

```
% cd $VOVDIR/bin
% ls
```

If you specified other platforms, then you can navigate to see the platform executables in the install area of the file system that will be used by remote machines of a given type, accessing them from the install area.

```
% cd $VOVDIR
% cd ..
% ls
# expect to see directory names corresponding to platform names
```

You can see the subdirectories there for each platform that was installed. Verify that there is a subdirectory there for each platform you requested to be installed.

You can also view the installation log file in `$VOVDIR/../install<datestamp>.log` that was created during the install process.

Rigorous Verification

After you have confirmed that the correct installation is accessible, you can do a rigorous check on the state of the install using a program that checks on a variety of aspects of your installation.

This will report on deeper issues with regard to the install and the configuration and the environment.

Use the `vovcheck` program as solid verification of the Altair Accelerator product installation.

You should run this program after you install a new version of Altair Accelerator software, or as needed to help with troubleshooting.

vovcheck

Basic check of configuration of VOV installation, configuration, and project setup.

```
vovcheck: Usage Message

     DESCRIPTION:
     Basic check of configuration of VOV installation,
     configuration, and project setup.
```

```
      USAGE:
      % vovcheck [options]

      OPTIONS:
      -h                          -- This help.
      -report   <file>            -- Name report file.
      -installation               -- Check the installation (default).
      -project                    -- Check the project.
      -t                          -- Turn on Tcl tracing (for debugging).
      -v                          -- Increase verbosity.

      FANCY OPTIONS:
      -class  {NP|PD|PS|PU}       -- Specify class of tests to perform.
      -- NP = No project   (check installation)
      -- PD = Project down
      -- PS = Project sick
      -- PU = Project up   (check project)

      EXAMPLES:
      % vovcheck
      % vovcheck -installation
      % vovcheck -project
      % vovcheck -report myreport.txt
```

Comments

As mentioned at the end of the `vovcheck` output above, the program creates a more detailed report in a file in the tmp directory. You **must** look at this report in order to interpret the results of the checking.

The output shows the focus of each test, along with the products for which the test is applicable and the result of the test. Not all checks are meaningful for all products or all environments. For your specific environment, a WARN or ERROR may be the expected and correct result.

The standard output will show WARN and ERROR results as a way to alert you to look within the report file to read the details about `vovcheck` has noticed.

The Products column indicates the product for which the test applies. You can ignore a WARN or ERROR result for a test that is checking an aspect of a product you are not using.

Look at the report file to see the details about each test's results. The report file has clear and detailed explanations about was expected and what was noticed, that can help you interpret the report. It also provides a hint on what to fix or change so that the `vovcheck` program will not produce the alert when doing the test.

Deep Verification Checks

The following table gives additional detail about the function of each check performed by `vovcheck`.

N	Check	Meaning
1	BasicVariables	Checks that environment variables VOVDIR, VOVARCH, VOV_HOST_NAME, VOV_PROJECT_NAME are set. Needed by all VOV projects.

△ ALTAIR

N	Check	Meaning
2	CwdPermissions	Checks that you have read/write permission for the current directory, needed to run jobs there in Accelerator
3	EnvBase	Verifies that the BASE environment exists and can be entered, and that a few basic UNIX and Altair Accelerator programs are in the PATH in that environment.
4	GuiCustomization	Checks for an old filename for GUI customization, and if present advises of the correct location.
5	HostPortConflicts	Checks the registry and advises if more than one project on the same host uses the same TCP port. Only one of such projects may be started, since the OS only allows one process to listen on a given port at a time.
6	Installation	Verifies that the Linux OS version is compatible with the Altair Accelerator libraries for runtime tracing.
7	Check	Checks for an old filename for `Make.tcl` defaults, and if present advises of the correct location.
9	RshSsh	Tries to verify whether there is a remote shell command set up that works to the current host without a password. The owner of Accelerator must be able to remote-shell from the vovserver host to the farm hosts to start the vovtasker.
10	SecurityPermissions	Checks the permissions on `$VOVDIR/etc/cgi` and other directories and files in the installation.
11	TaskerRoot	Verifies that `$VOVDIR/bin/vovtaskerroot` exists and has the desired ownership and permissions. This is needed in Accelerator so that jobs can run in the account of their submitter, so as to have the correct privileges with respect to the submitter's files.
12	SshSetup	Verifies that the `~/.ssh` directory has correct permissions to permit ssh to operate. This directory must be owned by the user or root, and not be readable by anyone else.
13	UsrTmp	Verifies that the directory `/usr/tmp` exists and is writable. Some Linux distributions (e.g. Ubuntu) do not include this by default.
14	VovPermissions	Verifies ownership and permissions of files in the `bin` and `scripts` directory of the Altair Accelerator installation.
15	WritableLocal	Checks permissions on directories `$VOVDIR/local/`{`capsules`,`environments`,`cgi`,`scripts`} where site-specific customizations are stored. These may be writable to permit

N	Check	Meaning
		users to add items when using FlowTracer, or not writable and under admin control when only Accelerator/Monitor are in use.
16	WritableRegistry	Verifies that the Altair Accelerator registry directory `$VOVDIR/local/registry` is writable. This is needed so that the `vovproject create` command can add files there. These files contain the metadata used to locate, start, stop, etc. VOV projects.
17	vovrc	Verifies that the source-able Altair Accelerator setup files created during installation are present. These are `$VOVDIR/etc/vovrc.{csh,sh,tcl}`. The `vovrc.sh` file is especially important to Accelerator, because it is used to set the environment when starting vovtaskers on the farm hosts.

Clean Up

Access the Help Documentation

Now that you've installed the product, you will want to read the product-specific documentation. Details about various ways to read the documentation is available in the Use Altair Accelerator Package Help topic.

Remove Downloaded Files

Now that the install of the downloaded files has been completed, you can remove the downloaded and expanded files from the staging area.

The only benefit in leaving them on your disk is if you may need to update the installation with additional platform-specific files.

An update of the existing release to add platform-specific is needed if you add a machine type to your network that was not in place during this install process. In that case, the downloaded and extracted common files will need to exist to do that update. The platform-specific files that are now already installed will not be needed during that add-platform update step. You can leave the common files on disk, or delete them now and download them again when you need them.

Licensing

The software license agreement to operate the Altair Accelerator software is enforced by either a standalone license file or a floating license checked out from a license server.

There are 3 licensing modes supported by Accelerator products:

- Altair License Manager floating licenses
- Standalone license file – Altair license file
- Standalone license file – legacy key file

The software component that checks out a license is vovserver, which applies license accounting rules and requests the type of license that is appropriate for the individual Accelerator product, together with the number of taskers, users, or other relevant data. When a standalone license file is in effect, the correct license authorization is checked in that file. When a floating license is in effect a shareable license is *checked out* from the license server.

Standalone License File

A standalone license file allows operation of Accelerator products without the need to run an Altair license server to dole out floating licenses. It is most suitable for a situation where a single VOV project will run on a dedicated computer system or node. A standalone license file will typically unlock the ability to run an Accelerator product instance on a specified host and port.

Use of standalone license files are simpler to deploy since an Altair License Server is not needed. Also, standalone license files can be more reliable because they do not depend on the network for licensing validation.

Floating License Server

An Altair License server (also referred to an "LMX" server for historical reasons) is installed and run to serve floating Accelerator product licenses. This approach allows license authorization rights to float between users and hosts. A customer site will have some number of license seats that are exclusively allocated to different Accelerator product instances.

Grace Periods

If any of the license files are missing or has expired, a grace period of 5 days is offered in which the product can run normally.

Grace period is activated for all license types upon these conditions:

- License has expired.
- License cannot be obtained at any point after at least one successful license checkout has occurred.
    - # For floating licenses, this can be anything from a network/routing error to a power cord being unplugged on the license server host.
    - # For node-locked licenses, this could only be if the license was deleted or modified while vovserver is running.

At the end of the grace period, Accelerator stops dispatching jobs and Monitor obfuscates all checkout information saved to the log files, and does not load it into the DB.

> 📄 **Note:** The grace period does not apply when a new license file is installed; the newly installed license file is quickly detected should the previous license expire or otherwise go bad.

Altair License Management

This is a Quick Reference Guide covering how to install, set up and manage your licenses keys using the Altair License Management system (ALM). ALM is derived from LM-X License Manager from X-Formation.

Getting Started with Altair Licensing

In Accelerator product versions 2023.1.0 and later, Altair Licensing is activated by default. If upgrading from a Reprise-based license for FlowTracer and seat-based licensing is being utilized, the name of the seat-based license will need to be changed to an Altair Licensing feature name. See Licensing FlowTracer by Seat.

For a standalone license file situation, set ALTAIR_LICENSE_PATH to the pathname of the license file. For floating license usage, set ALTAIR_LICENSE_PATH to "PORT@HOST" for the ALM license server.

The I/O job profiling feature enables Accelerator to track and plot performance statistics over the time the job is running. This feature can only be activated with a Mistral license, which must be installed at `$VOVDIR/local/mistral.dat`. Check with your system administrator for accessing and installing the license.

The following sections will describe procedures for managing floating ALM licenses and the ALM license server.

Install the ALM License Server Software

The ALM license server software and configuration files are included with the Accelerator products, and are referred to as the "ALM bundle". The location is `$VOVDIR/alm`. The files from that location should be copied to a directory local to the server where the Altair License Manager will run. In this example, the location `/opt/alm_admin` is used. The installation and running of the license server should be done by a "role" account on the Linux machine. Here, that role account is called `almmgr`. The steps are as follows:

1. Log in to the role account `almmgr`.

2. Copy the ALM bundle to the ALM installation location:

   ```
   % cp -r $VOVDIR/alm /opt/alm_admin
   ```

3. Copy your ALM license file `ALUS_ALUS*.dat` into `/opt/alm_admin`.

ALM Bundle

The following is a detailed look into the ALM file bundle.

altair_serv.cfg

    Configuration file that can be used to start the lmx-serv license daemon.

```
#*************************************************
# TCP/UDP port number the license server will listen on.
# TCP port is used for data traffic protocol.
# UDP port is used for automatic server discovery protocol.
#
# The default TCP port is 6200.
# The UDP port is fixed to 6200 and cannot be changed.
# See http://www.iana.org/assignments/port-numbers
```

```
#***************************************************
TCP_LISTEN_PORT = 6200
#***************************************************
# Set the log file path:
# It is preferred to write out the full path.
#***************************************************
LOG_FILE = ./altair_license.log


#***************************************************
# Set the log file format.
# The following formats are valid:
# NORMAL, EXTENDED
# Setting the log file format to EXTENDED causes
# additional information to be included in the log
# file, such as license server HostIDs, whether the
# license server is a virtual machine, etc., which
# is useful for debugging purposes.
#
#***************************************************
#
LOG_FORMAT = NORMAL
#***************************************************
# Set a license file path:
#
LICENSE_FILE = /opt/alm_admin/ALUS_npaladugu1_20230812.dat
```

alus.conf

Standard global configuration file.

```
## Altair alus.conf file
#
#******************************************************************************
## Server Logging Level (Altair Specific Logging)
## Possible values are:
##
## 0 No Altair Specific Logging
## 1 Errors (Default)
## 2 Debug
## 3 Info
##
ALUS_DEBUG_LOG_LEVEL=1
#******************************************************************************
#
#******************************************************************************
## Location and settings of files for license server and reporting tools
#
ALUS_PENDING_DIRECTORY=./pending
ALUS_PROCESSING_DIRECTORY=./processing
ALUS_SCRATCH_DIRECTORY=./scratch
ALUS_DETAILED_DIRECTORY=./detailed
ALUS_LOGGER_DIRECTORY=./logs
ALUS_MANUAL_DIRECTORY=./manual
ALUS_SENT_DIRECTORY=./sent
ALUS_INVALID_DIRECTORY=./invalid
ALUS_UNDELIVERABLE_DIRECTORY=./undeliverable
ALUS_UNDELIVERABLE_DB_FILE_PATH=./undeliverable.properties
#******************************************************************************
## Location of Decay Directory
#
ALUS_DECAY_DIRECTORY=./decay
#******************************************************************************
#
```

```
#*****************************************************************************
## Setting for Altair Logging level of transactional data
#
## If you are part of the Partner Program then ALUS_LOG_LEVEL must be set to
## at least 1. Setting this to 0 is VERY UNCOMMON and should be done only
## under special circumstances.
##
## 0 = No logging
## 1 = Anonymized only
## 2 = Anonymized and Detailed
#
ALUS_LOG_LEVEL=0
#*****************************************************************************
#
#*****************************************************************************
## Location of script to run reporting tool
#
ALUS_URT_LOCATION=./urt.sh
```

liblmxvendor.so

Dynamic shared library that is needed by lmx-serv daemon.

lmx-serv

ALM License server that needs to be launched for Accelerator products.

```
% lmx-serv -h would give you usage with examples.
#
# Launch the license server below in background mode, using altair_serv.cfg and
 capture the logs into a logfile.
% ./lmx-serv -b -c /opt/alm_admin/altair-serv.cfg -lf /opt/alm_admin/
alm_license_server.log
```

urt.sh

Empty shell script used to disable telemetry services.

Default Port

The default port chosen by ALM services is 6200. It is defined in the `altair-serv.cfg` file. It is best to leave it as it is. However you are welcome to make your changes and set your ALTAIR_LICENSE_PATH accordingly.

Start and Stop the License Server

1. To start the license server:

   a) Point to the license file:

   ```
   % su - almmgr
   % cd /opt/alm_admin
   % lmx-serv -b -c /opt/alm_admin/altair-serv.cfg -lf /opt/alm_admin/
   alm_license_server.log -l  ALUS*.dat
   ```

   b) In the location where the files are installed, run the following command to start the license server:

   ```
   lmx-serv -b -c /opt/alm_admin/altair-serv.cfg -lf /opt/alm_admin/
   alm_license_server.log
   ```

2. To stop the license server, run the following:

```
% killall lmx-serv
```

Update Licenses

When new floating licenses need to be served, or when replacement licenses are to be introduced, edit or replace the `ALUS*.dat` file in `/opt/alm_admin` with a file containing all the new licenses from the new `ALUS*.dat` files received from Altair. Then, stop and start the lmx-serv process by the procedure described in the previous section.

If an Accelerator product vovserver is running when these changes are made, the expired or removed license checkouts will be cancelled, and the vovserver will check out new licenses at the next heartbeat with no interruption of service.

Troubleshooting

Is ALM enabled?

1. Run these steps below:

```
% vovproject enable <projectname>
```

where your `projectname` is one of the Accelerator Products.

2. Look for ALM status.

```
% vovshow -policy | grep -i alm
VovServerConfig alm.enable                      "1"
```

"1" means enabled. "0" means disabled.

vovbrowser Alerts

1. Type **vovbrowser** at the Unix command prompt. Chances are you will see the error message below.

```
spindiproli1 FT-alm_debug@spindiproli1 [DEFAULT] 1183 > vovbrowser
vovbrowser 11/10/2022 08:39:12: message:
 VOV license violation!!
 Thu Nov 10 08:39:12 PST 2022

 All retracing is now disabled
 in this project FT-alm_debug@spindiproli1.

 You need to eliminate the license violation
 before continuing.

 The current value of RLM_LICENSE is /projects/spindiproli/vov/NC_install/
 trunk_dev-105571_20221024_003038_231/local/rtda.lic
 The current value of ALTAIR_LICENSE_PATH is 6200@npaladugu1

 If you need a larger license,
 please file a support request at https://community.altair.com/community
```

```
http://spindiproli1:9880
```

2. Open the URL shown using your favorite internet browser, and view **Project Home** > **Admin** > **Alerts**. The example below illustrates a FlowTracer project.

Figure 4:

3. Notice that line #2 states "Bad seat license specified which says seat_ft_l" which means that the FlowTracer project by default is looking for ALM license feature seat_ft_l which is set in `$SWD/policy.tcl`.

Environment Variables

The following environment variables can be set on Unix and Windows to control certain features within LM-X and the Altair Licensing System.

Table 1:

Variable	Description
ALTAIR_LICENSE_PATH	Sets the path to the license file or address of the license server. The format is one of the following: • path_name • PORT@HOST Multiple license files can be combined using ":" on UNIX and ";" on Windows. This is a client variable.
ALM_PROJECT	Used to report project strings in the transactional logs. This string will be part of the recorded transaction. The value set should be 30 characters or less. String may contain valid characters in the portable filename character set (A-Z, a-z, 0-9, { } / \< > ( ) ` ', ? | [ ] * & ^ $ # @ ! ~ + =). Any other characters will be URL encoded in the server logs. The maximum length is 30 characters. This is a client variable. 📝 **Note:** This will NOT be included in the anonymous log files, only in the detailed logs.

△ **ALTAIR**

Variable	Description
ALUS_CONF_FILE	Sets the path to the alus.conf file. This must be set prior to starting the license server process. This is a server variable.
ALM_NO_EXPIRE_WARNING	Setting this value to any non-zero length value will prevent applications from warning users that the license is going to expire. This is a client variable.
ALM_DONGLE_ENABLED	Setting this to any value will enable a client to use a license dongle. The server is always enabled for dongles. This must be set to enable client dongle support. This is a client variable.

Licensing FlowTracer by Seat

FlowTracer can be licensed by "seat". Each seat can be considered a license to implement flows with up to a certain number of jobs and up to a certain number of slots for parallel execution. Multiple seats are checked out automatically to cover the requirements of each flow.

The seats come in 3 sizes, namely L, XL, and GXL, each corresponding to an ALM feature called `FTSeatLFeature`, `FTSeatXLFeature`, and `FTSeatGXLFeature`.

To activate seat licensing, set the parameter `seatlic` to the name of the feature to use. This parameter can take the following values:

- "-" to mean "no change"
- "" to mean "disable"
- <name_of_seat_license> to select a specific type of seat license

```
% echo "SELECT SEAT LICENSE FROM THE COMMAND LINE"
% source /<install_path>/<version>/<platform>/etc/vovrc.csh
% vovsh -x "vtk_server_config seatlic FTSeatLFeature"
```

or

```
## This is in policy.tcl
set config(seatlic) "FTSeatLFeature";# Enable seat licensing with feature
 FTSeatLFeature

# set config(seatlic) "-" ;# No change.
# set config(seatlic) ""  ;# Disable seat licensing
```

To disable seat licensing, you need to set the parameter `seatlic` to the empty string, as in:

```
% source /<install_path>/<version>/<platform>/etc/vovrc.csh

% vovproject enable <project_name>
```

```
% vovservermgr config seatlic ""
```

```
## This is in policy.tcl
set config(seatlic) ""
```

Remember that after changing the `policy.tcl` file, you need to ask for:

```
% vovproject sanity
```

Each FlowTracer server will use one or more seats, but all seats have to be of the same type. For example, if you are using the seat feature "FTSeatLFeature" and your flow has 8345 jobs, you will need 5 seats of type "FTSeatLFeature", which will also give you access to 100 parallel slots.

Under normal circumstances, it is expected that the seat licenses given to a customer are sized in such way as to allow most FlowTracer servers to use exactly 1 seat, and occasionally 2 seats. In general, however, there is no limit in the number of seats that a given FlowTracer server can use to accommodate the size of the flow, expressed by the total number of jobs in the flow.

The license seat requirement is checked about once an hour. Excessive seats are released or additional seats are checked out. To force an immediate check of the license seats, run `vovproject sanity`.

If the required number of seats cannot be obtained, the vovserver goes into "License Violation", which means that no additional jobs can be dispatched to the taskers. Everything else still works.

vovserver Licensing

Each Accelerator product has a running vovserver with an particular product identity. The license features checked out, or validated in a license file, for the different Accelerator products are as follows.

Product	Nickname	Altair Floating License Features	Altair License File Notes
Accelerator	NC	NCSlotFeature The number of NCSlotFeature license tokens consumed by an NC tasker is the lesser of the number of configured NC job slots and the number of logical CPUs on the host.	The number of slots is identified by the SLOT attribute on the NCSlotFeature feature.
Accelerator Plus	WX	WXSlotFeature WXSlotFeature tokens consumed by an Accelerator Plus queue instance = (number of WX taskers in the base queue) + (number of logical CPUs on the Linux host where	The number of slots is identified by the SLOTS attribute on the WXSlotFeature feature.

Product	Nickname	Altair Floating License Features	Altair License File Notes
		the Accelerator Plus server is running.	
Allocator	LA	LAJobFeature is the allowed jobs count.	JOB attribute on the LAJobFeature set to the job count
FlowTracer	FT	FTSlotFeature, FTSeatMFeature, FTSeatLFeature, FTSeatXLFeature, FTSeatGXLFeature, FTSeatUFeature	For FTSlotFeature, the slot count is the SLOT attribute
Hero	HE	HELeafFeature	The SLOT and LEAF attributes are specified
Monitor Basic	LMS	(none)	
Monitor	LM	LMUserFeature One token per user.	The USER attribute is specified.

To find out the product supported by a vovserver, use:

```
% source /<install_path>/<version>/<platform>/etc/vovrc.csh
% vovsh -x 'vtk_generic_get project info; puts $info(product)'
```

If vovserver loses a feature, it will keep running for a grace period of five days.

If the vovserver fails to get the appropriate license upon startup, or if the grace period expires, it will go into "License Violation", which essentially means that dispatching to remote taskers is disabled. Every other behavior remains fully functional.

To check the current licensing status, visit /cgi/license.cgi.

License Keys

License Key Files

License key files can be installed either in the local directory of the installation tree or in the product-specific server working directory (SWD). License key files installed in the local directory must be in the format of `license.productcode.key`, where `productcode` is one of `"lm"` or `"nc"`.

License key files that are installed in the SWD can use this format or the name `license.key`. For example, for Monitor, use the file `licmon.swd/license.key` and for Accelerator use `vnc.swd/license.key`.

> 📝 **Note:** The SWD is created upon product start up, so ensure the product has been started before attempting to put a SWD license key file in place.

The vovserver searches for the keyfile license as follows:

1. The file named by the environment variable VOV_LICENSE_KEY, if set. This should be defined in the project's `setup.tcl` file.

2. `SWD/license.<productcode>.key`

3. `SWD/license.key`

4. `$VOVDIR/local/license.<productcode>.key`

The license key files encodes the host IDs of the hosts where the application is allowed to execute and the maximum allowed number of tokens provided by the license.

The following is an example of one such license key file, to enable an Accelerator project (product: nc) on port 6271 on a machine called mac05 for up to 100 slots (license for: 100 slots).

Example license key file:

```
#      -----------------------------------
#      -- VOV License  Key  Information --
#      License S/N: 20161110a
#      VovVersion:  2023.1.0
#      Product:     nc
#      Licensed to: altair
#      For host id: MAC:00:25:00:9f:76:68
#      For host:    mac05
#      For port:    6271
#      License for: 100 slots
#      StartDate:   Tue Nov 10 10:29:25 2022
#      Expiration:  Thu Dec 31 21:00:00 2023
#      ----------------------------------
#
EFIFFF`gjdCEceFDdd`ekgMDEDMEBGheee`ejcGEiadeifogJA
iel`neC@`eca@EHAMDFAgeD@LENFedfdCEceJE`angFE`dn`AA
L@AAADNGkfA@IADEBAkaB@o`KEceADEFJGlfnbJEd`bdFBeeod
CDeehgidbaOEjgMENE`ejcOEhdeeieedadceMEGEcc@@AECEle
bdHDCEfbDEEBEDICIEIFAFEAG@oeLACDndLEDENAg`KFhgfdAA
BDddodCEbeJBddne@DFFaejeODjdD@AEAALDAAKGk`BDofCGl`
dek`@@OFidd`aeedadheheoeACGEee`eieedadeeceLEAEIGNG
`e
```

Multiple license key files can be concatenated into the same file and the combined value of all the license keys will be used by the application. The serial numbers of such licenses must be distinct. You must leave the comment lines that precede the encrypted section.

This licensing mode is activated if the `license.key` file is present, even if it is empty or invalid:

If the software is using the license key, vovserver manages the licenses for any vovtasker that connects it. The vovtaskers do not read the license key files.

Checking the lLcense Key File

An easy way to verify a license key file is with the utility `vovreadlicutil`.

△ ALTAIR

vovreadlicutil

```
usage: vovreadlicutil [-f filename] [-P port] [-p product] [-V version]
                      [-I hostid] [-tv]
   -f:           Specify a license file.
   -P:           Specify port number.
   -p:           Product name to test (nc,lm,ft,lam,wa,...)
   -V:           Release date of version to test, in YYYY.MM format
   -I:           HostId to test
   -t:           Generate Tcl array with license info
   -v:           Print version.
```

Example

The typical use is the following:

```
% source /<install_path>/<version>/<platform>/common/etc/vovrc.csh
                  % vovreadlicutil -p nc -f vnc.swd/license.key
                  ----------------------------------
                      -- VOV License  Key  Information --
                      License S/N: 20160909T153538
                      VovVersion:  2020.1
                      Product:     nc
                      Licensed to: Altair
                      For host id: MAC:00:25:00:9f:76:68
                      For host:    mac05
                      For port:    6271
                      License for: 40 slots
                      StartDate:   Thu Sep  9 06:35:38 2019
                      Expiration:  Fri Dec 31 12:00:00 2021
                      ----------------------------------
                  vovreadlicutil Nov 14 14:11:27 Valid for product nc
                  vovreadlicutil Nov 14 14:11:27 Valid for this host
 MAC:00:25:00:9f:76:68
                  vovreadlicutil Nov 14 14:11:27 Valid for port 6271
                  vovreadlicutil Nov 14 14:11:27 Valid for version 2019.01
                  vovreadlicutil Nov 14 14:11:27 Valid expiration (still 112d21h to
 go)
                  vovreadlicutil Nov 14 14:11:27 The combined license is:
                  ----------------------------------
                  -- VOV License  Key  Information --
                  License S/N: 20160909T153538
                  VovVersion:  2020.1
                  Product:     nc
                  Licensed to: Altair
                  For host id: MAC:00:25:00:9f:76:68
                  For host:    mac05
                  For port:    6271
                  License for: 40 slots
                  StartDate:   Thu Sep  9 06:35:38 2019
                  Expiration:  Fri Dec 31 12:00:00 2021
                  ----------------------------------
```

Add New Platform to the Current Release

The Add Platform installation scenario updates the set of installed files to add more files that are the executables for the added platform.

You will need to perform this Altair Accelerator software update installation if you add a new machine type to your network and did not install platform-specific files for that machine type when doing the original install of the current version.

When you installed an Altair Accelerator product, you requested the install of platform-specific files to support a group of machines on your network. If you later add a new machine type to your network, you will need to update your Altair Accelerator installation to add the platform-specific files for the new machine type.

The activity of doing the download and install for the incremental installation to add a new platform is similar to doing a regular install either for the first time or to upgrade to a new release. However, during this process, you will request an install for **only the added platform** and not for any of the platforms that are already installed.

Overview of the steps to follow:

1. Download the software to install from the Altair website, with guidance from the download information contained in a later section.
2. Perform the update software installation, which should require between 5 and 10 minutes depending on disk and network performance.
3. Validate the update installation by setting your command line environment and validating the installation.

The following sections will lead you through the steps to update your current installation to add support for additional platforms.

Plan to Add a New Platform

This section contains a discussion of what to investigate as you consider adding support for a new platform in your network. This will influence what steps you will follow to update your current release by installing platform-specific files for the added platform.

1. To begin, identify if there are software updates published for both your existing platform, as well as the new platform. Log onto the Altair website, and navigate to the download page to determine the most recent update package.
2. Select the version you have installed, and expand the link to see all available updates.
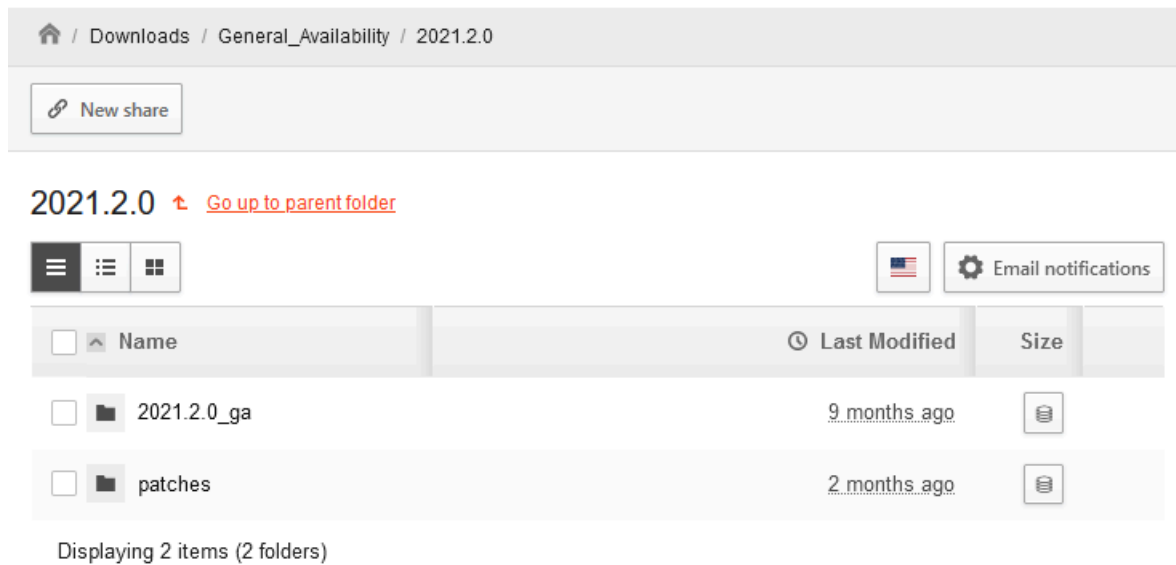
3. Look through the list to see if there are any updates for either your current or target platform. Note that updates are recommended, but not required. If desired, you can download the release notes, which explains the changes in the update files.

4. Patches are published between recommended software updates. Each patch contains a fix to a specific issue, which is then rolled into the next update. If there are no software updates for your target platform, then you only need to add the platform for the current release. If you need to install an updated version, use the two stage install process:

    a) Download and install the update package, including your original platforms and the one you'd like to add.

    b) Check for any patches for your newly installed version.

5. Continue with the steps in this chapter.

Download the Software

To enable support for a new machine type, you need to download the platform-specific files for the machine type and prepare the files to do the install.

Use the Altair Accelerator Product Administrator Account

Log in to your computer, into your Altair Accelerator Administrator account. This is the user that was set up to do the standard install, and who owns all the files in the release.

In the following sections, examples will be written as if the user account named rtdamgr is the Altair Accelerator Administrator. If you are installing Accelerator, there is one step in the procedure that must be performed by the root user.

Create or Reuse a Temporary Download Staging Area

Create a staging area to hold the downloaded files. It must be different from the directory where you plan to install the Altair Accelerator software. The staging area may be removed after installation.

For example, create the staging area directory as user accel_mgr.

    a) On UNIX:

```
% mkdir /usr/tmp/altair_download
```

    b) On Windows:

```
C:\> md  c:\temp\altair_download
```

> **Note:** Location path names must not contain any spaces.

Download and Expand the Files

1. Access the Altair Marketplace and select the product to install.

2. Log in using the account you set up as part of the pre-installation planning steps.

3. Click the link for the release version you want, and download the appropriate files.

4. For a UNIX installation, complete the steps below:

    a) Download the `Common.tar` file into the staging directory you just created.

    b) Download the platform-specific file(s) for each platform type that you will be using.

    c) Expand the archives with tar extract.

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf <platform>.tar    # For each platform
```

       For example:

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf linux.tar
```

5. For a Windows installation, complete the steps below:

    a) When Windows is one of the platforms used in a networked multi-platform situation, you should download the `win64.tar` file into the staging area.

> **Note:** The name "win64" is a generic name for all versions of 64-bit Windows.

    b) Expand the archives with tar extract:

```
% cd /usr/tmp/altair_download
% tar xf win64.tar
```

c)   Expand this zip file into the staging area using Windows Explorer (via right-clicking on the file) or any utility that works with zip files.

Install the Software

Install Added Platform to Current Release

You may need to know the path to the current release in order to perform the operations described here. The path that is needed is the one that ends with the release number.

In the examples here, the current installation path will be shown as the `/opt/rtda/2020.1` directory. When you see this in examples, you should understand that it is referring to the path to your current installation.

Installation via the Graphical User Interface

Start the Installer Graphical User Interface for your platform as shown below. The GUI installer uses the X-Windows technology for the Graphical User Interface. If you are running on a remote UNIX system in X-windows, be sure your *DISPLAY* environment variable is set properly.
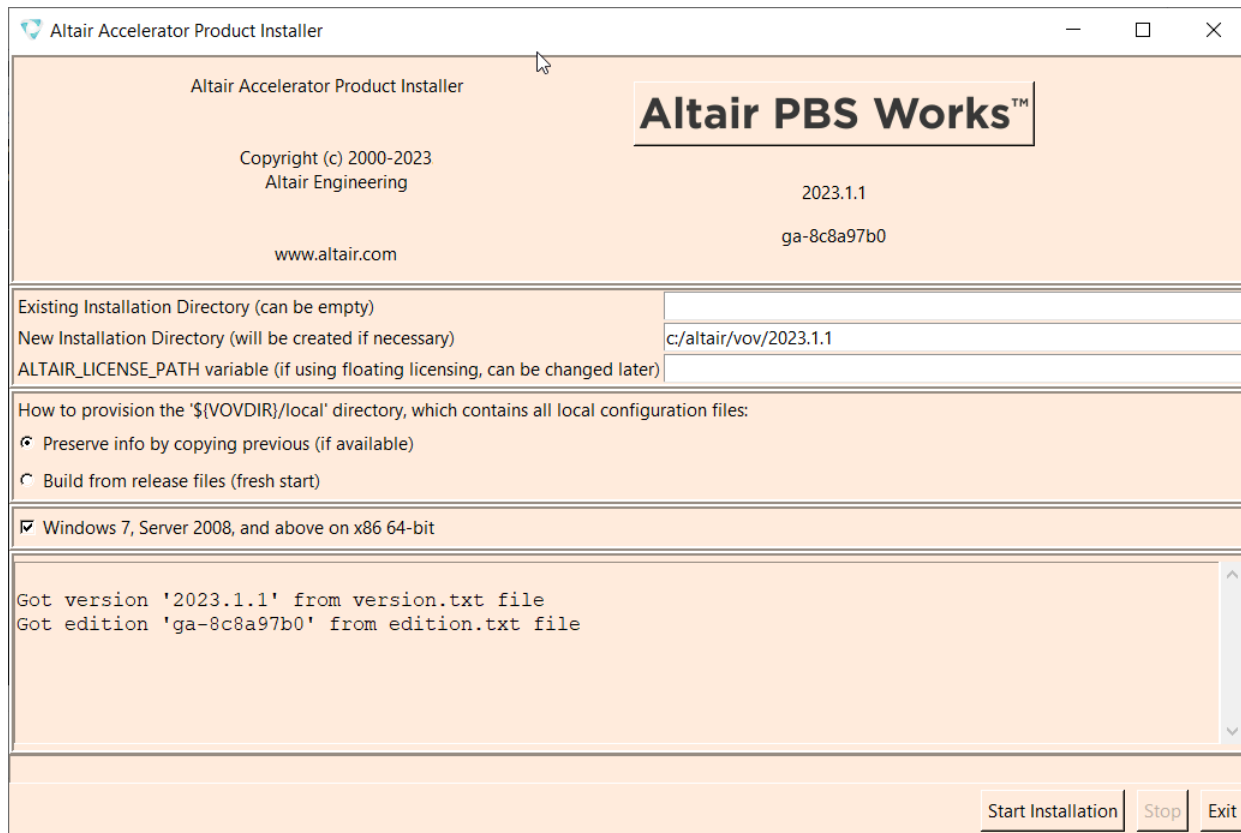

Figure 5:

1. Change the working directory to the staging directory where you downloaded the tar files and extracted them, and run the GUI install script `install.sh`.

   a) On UNIX:

   ```
   % cd /usr/tmp/altair_download
   % ./install.sh
   ```

   b) On Windows:

   ```
   C:\> cd  c:\temp\altair_download
   C:\> install.bat
   ```

   The installation dialog appears. Since this is an update to an existing release, the **Existing Installation Directory** and **New Installation Directory** fields should be the same value, and that value should be the location of the current installation. The value of the ALTAIR_LICENSE_PATH variable field will be obtained from the current installation and does not need to be changed.

2. Use the first section to tell the installer where to install the Altair Accelerator software and where it will find the Altair Accelerator license server.

3. Use the second section to setup the new release so that locally changed files are placed into the new release. Because this is an update of an existing release, you want the install to do nothing with regard to your local directory. You should choose **Preserve info by copying previous**. This tells the installer to copy the "local" directory file in the current release to itself, which is effectively to do nothing.

4. In the Platforms section, indicate which platform you are adding during this install. Clear all the other platforms that have already been installed.

   This directs the installer to place the binary executables for these machine types into the installed directory structure. The files will be placed within subdirectories named for the platform. The installed files for each platform will come from the staging area where you have downloaded and extracted them.

   If you are installing on a UNIX system, you could select the Windows platform here if the Windows file (`win64.tar`) was extracted into the staging area. The Window binaries will be installed on the UNIX machine as files within the win64 platform subdirectory.

   The intention of platform selection is to provide executable files in the file system on this machine, for the added platform in the network. This enables the remote machine to access the executables of their type by way of NFS, without needing to do an install of files on any other machines.

5. Click **Start installation** to start the installation. It normally takes a few minutes to run, depending mostly on the speed of your disk.

   During the install, the program will write messages into the bottom section of the dialog, which is a scrollable region. The messages are also written to a log file on disk in the new installation directory, named `install<timestamp>.log`.

6. When the installation is done, click **Exit**.

7. You can review the install log for errors and warnings. The last few sentences of the log file remind you to do something if you are installing Accelerator. This subject is explained in *For Accelerator Installation Only*.

Installation via a Batch Script

You can install the product via a batch script, if desired.

To perform a batch install on Linux, use the script `install.sh`. For Windows, run `install.bat`.

Run the command `./install.sh`

The values that control the installation are passed into the batch script as command line parameters, except for the *Existing Installation Directory*. The batch script gets the value for this from the $VOVDIR environment variable as it runs.

Because this is an update install to add a new platform to a previously installed release, the options `-nocommon` and `-local copy` should be used. The "copy" choice for the local parameter will cause the process to copy the local files onto themselves, creating a non-destructive no-op operation.

Here is the logical description of calling the batch install for the scenario of adding platform-specific files for a new platform.

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/<version> -nocommon -local copy -alm 6200@almsrv -
platforms '<added-platform>'
```

Here is a specific example of doing an update by running the batch install to add the platform Linux64 to the release.

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/<version> -nocommon -local copy -alm 6200@almsrv; -
platforms 'linux64'
```

install.sh

This script supports batch installation on Linux as well as Windows. Run from the directory where the installation files were extracted.

By default, the installer will run in graphical mode. Use the -batch option to suppress the GUI and run in batch mode.

Usage

`./install.sh [INSTALLPATH][OPTIONS]`

Options

Use the options below for non-interactive installation.

-alm PORT@HOST	Value to use for ALTAIR_LICENSE_PATH.
-batch	Install in batch mode. If INSTALLPATH is not passed as an argument, the default installation path for Linux is /opt/altair/vov/1212.10 and for Windows is \opt\altair\vov\1212.10.
-local HANDLING	Specify handling of the 'local' directory where HANDLING is one of:

- copy (copy previous installation's local)
- create (create new from installation stubs)
- link (link to previous installation's local)
- linkext (link to a version-independent local)

The default handling for Linux is linkext, which results in the local directory being placed in the parent of the installation and a symbolic link created inside the installation to point to it. This facilitates the reuse of the local directory for all installed versions.

	The default handling for Windows is copy. Only copy and create are supported on Windows.
-nocommon	Used when installing additional platforms after an initial installation has been performed, to prevent overwriting of the common directory. Ignored on Windows.
-noconfirm	Do not prompt for confirmation to begin the installation process when running in batch mode.
-platforms LIST	A quoted, space-separated list of platforms to install. Supported platforms are: armv8, linux64, and win64.
-previous PATH	Path to previous installation, if applicable. Defaults to the value of $VOVDIR if set in the calling environment. Otherwise, defaults to an empty string.

Examples

```
./install.sh /opt/altair/vov/<version>
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv -batch
./install.sh /opt/altair/vov/<version> -local create -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -local link -platforms 'linux64 win64'
```

Because this is an update install to add a new platform to a previously installed release, the options -nocommon and -local copy should be used. The `copy` choice for the local parameter will cause the process to copy the local files onto themselves, creating a non-destructive no-op operation.

Here is the logical description of calling the batch install for the scenario of adding platform-specific files for a new platform.

```
% cd /opt/altair/vov/2023.1.0
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy <port@host> -platforms
  '<added-platform>'
```

Here is a specific example of doing an update by running the batch install to add the platform Linux to the release.

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy -alm 6200@almsrv; -
platforms 'linux64'
```

For Accelerator Installation Only

Accelerator requires one extra step during the install. This step creates a set_UID executable that Accelerator uses so that jobs can run under the ID of the user who submits them. This step requires root capability to have it run correctly.

If you do not have root privileges, or if you are performing an evaluation, you can postpone this step. You can still run the software in single user mode, but you will not be able to run it correctly with multiple users due to permission issues.

You must be logged in as root or have *sudo* privileges to perform this step correctly to create the `vovtaskerroot` binary for each installed platform.

ALTAIR

> 📝 **Note:** You must run the `SETTASKERUID.sh` script exactly as shown in the next example. The script will not work correctly unless it is executed from the specified directory.

The following script creates a set-UID executable called vovtaskerroot so that jobs can be run under the ID of the user who submits them. This script also sets the set-UID bit in `vovgetpasswd` for Linux.

To complete the installation, you need to login as root and install the binary for `vovtaskerroot` by means of the following commands:

```
% /bin/su -
# cd <install_path><version>
# sh ./scripts/SETTASKERUID.sh
# exit
```

For example:

```
% /bin/su -
# cd /opt/rtda/2020.1
# sh ./scripts/SETTASKERUID.sh
# exit
```

At this point, installation is complete.

Set Up Environment Variables

In order to run the software for validation, you need to set up your command line environment properly.

This includes changing your PATH environment variable so you can run the installed executables, and adding two environment variables that are used by the Altair Accelerator programs.

Set your command line environment by sourcing a setup file created by the installation. You will source the setup file that matches your platform and shell. As earlier, we are using the path `/opt/rtda/<release>` in our examples as the location of the installation.

Platform Type	Shell	Command to Source File
UNIX	csh tcsh	source `/opt/rtda/2020.1/common/etc/vovrc.csh`
	bash sh ksh zsh	`. /opt/rtda/2020.1/common/etc/vovrc.sh`
Windows	DOS	`c:\rtda\2020.1\win64\bat\vovinit.bat`

Each one of these files defines the variable VOVDIR, which will be used in the rest of the installation documentation and which must be set for proper operation of Altair Accelerator software.

△ ALTAIR

The environment variable VOVDIR always points to the platform specific directory, so that, for example, the executables can be found in `$VOVDIR/bin`, the documentation can be found in `$VOVDIR/doc`, and the local information can be found in `$VOVDIR/local`.

Verify the Installation

Install the License

Before you can verify the installation, you will need to install the proper licensing. Follow the instructions in Licensing to enable execution of Altair Accelerator software.

Quick Verification

After completing the installation and setting up your shell environment, you can do a quick verify that the install was successful by running a program that reports on the installed version. This confirms that the installed program is accessible on your PATH and that it is the version you expect.

Use the `vovversion` program to confirm you are accessing the desired version of the product.

Running this command should give you the expected version number, and the expected location where you have installed the product.

It is also a good practice to check that your environment variables have been set up correctly:

```
> source /<install_path>/<version>/<platform>/common/etc/vovrc.csh
% vovversion -all
2020.1
VOVDIR=/opt/rtda/2020.1/linux
```

You should see the expected version number, and the expected location where you have installed the product.

vovversion

Print the VOV version. Optionally show more details.

```
vovversion: Usage Message

DESCRIPTION:
    Print the VOV version. Optionally show more details.

USAGE:
    vovversion [options]

OPTIONS:
    -help      -- Print this message
    -all       -- Print all available version information
    -clients   -- List clients and their version numbers
    -install   -- Print the VOV version with patch names, if any
    -patches   -- Print patch names, if any
    -project   -- Print the version of vovserver associated with the
```

```
                    current project
    -short      -- Print VOV version with short patch name, if any
    -verify     -- Ensure server and client versions match

    [Can abbreviate options provided they are unambiguous]
```

Visual Verification

You can navigate around the install area to make sure that there are subdirectories for each platform that you requested to be installed.

When you set up your environment in the previous section, you set the environment variable VOVDIR to be the path to your platform specific files. You can navigate to that area of the installed files to look around to see the other platform files.

You can navigate to see the platform executables in the install area of the file system that were just finished.

```
% cd $VOVDIR
% cd ..
% ls
# expect to see directory named for the added platform
```

You can see the subdirectories there for each platform that was installed. Verify that there is a subdirectory there for each platform you requested to be installed.

You can also view the installation log file in `$VOVDIR/../install<datestamp>.log` that was created during the install process.

Rigorous Verification

After you have confirmed that the correct installation is accessible, you can do a rigorous check on the state of the install using a program that checks on a variety of aspects of your installation.

This will report on deeper issues with regard to the install and the configuration and the environment.

Use the `vovcheck` program as solid verification of the Altair Accelerator product installation.

You should run this program after you install a new version of Altair Accelerator software, or as needed to help with troubleshooting.

vovcheck

Basic check of configuration of VOV installation, configuration, and project setup.

```
vovcheck: Usage Message

    DESCRIPTION:
    Basic check of configuration of VOV installation,
    configuration, and project setup.
```

```
        USAGE:
        % vovcheck [options]

        OPTIONS:
        -h                              -- This help.
        -report  <file>                 -- Name report file.
        -installation                   -- Check the installation (default).
        -project                        -- Check the project.
        -t                              -- Turn on Tcl tracing (for debugging).
        -v                              -- Increase verbosity.

        FANCY OPTIONS:
        -class  {NP|PD|PS|PU}        -- Specify class of tests to perform.
        -- NP = No project   (check installation)
        -- PD = Project down
        -- PS = Project sick
        -- PU = Project up   (check project)

        EXAMPLES:
        % vovcheck
        % vovcheck -installation
        % vovcheck -project
        % vovcheck -report myreport.txt
```

Comments

As mentioned at the end of the `vovcheck` output above, the program creates a more detailed report in a file in the tmp directory. You **must** look at this report in order to interpret the results of the checking.

The output shows the focus of each test, along with the products for which the test is applicable and the result of the test. Not all checks are meaningful for all products or all environments. For your specific environment, a WARN or ERROR may be the expected and correct result.

The standard output will show WARN and ERROR results as a way to alert you to look within the report file to read the details about `vovcheck` has noticed.

The Products column indicates the product for which the test applies. You can ignore a WARN or ERROR result for a test that is checking an aspect of a product you are not using.

Look at the report file to see the details about each test's results. The report file has clear and detailed explanations about was expected and what was noticed, that can help you interpret the report. It also provides a hint on what to fix or change so that the `vovcheck` program will not produce the alert when doing the test.

Deep Verification Checks

The following table gives additional detail about the function of each check performed by `vovcheck`.

N	Check	Meaning
1	BasicVariables	Checks that environment variables VOVDIR, VOVARCH, VOV_HOST_NAME, VOV_PROJECT_NAME are set. Needed by all VOV projects.

N	Check	Meaning
2	CwdPermissions	Checks that you have read/write permission for the current directory, needed to run jobs there in Accelerator
3	EnvBase	Verifies that the BASE environment exists and can be entered, and that a few basic UNIX and Altair Accelerator programs are in the PATH in that environment.
4	GuiCustomization	Checks for an old filename for GUI customization, and if present advises of the correct location.
5	HostPortConflicts	Checks the registry and advises if more than one project on the same host uses the same TCP port. Only one of such projects may be started, since the OS only allows one process to listen on a given port at a time.
6	Installation	Verifies that the Linux OS version is compatible with the Altair Accelerator libraries for runtime tracing.
7	Check	Checks for an old filename for `Make.tcl` defaults, and if present advises of the correct location.
9	RshSsh	Tries to verify whether there is a remote shell command set up that works to the current host without a password. The owner of Accelerator must be able to remote-shell from the vovserver host to the farm hosts to start the vovtasker.
10	SecurityPermissions	Checks the permissions on `$VOVDIR/etc/cgi` and other directories and files in the installation.
11	TaskerRoot	Verifies that `$VOVDIR/bin/vovtaskerroot` exists and has the desired ownership and permissions. This is needed in Accelerator so that jobs can run in the account of their submitter, so as to have the correct privileges with respect to the submitter's files.
12	SshSetup	Verifies that the `~/.ssh` directory has correct permissions to permit ssh to operate. This directory must be owned by the user or root, and not be readable by anyone else.
13	UsrTmp	Verifies that the directory `/usr/tmp` exists and is writable. Some Linux distributions (e.g. Ubuntu) do not include this by default.
14	VovPermissions	Verifies ownership and permissions of files in the `bin` and `scripts` directory of the Altair Accelerator installation.
15	WritableLocal	Checks permissions on directories `$VOVDIR/local/` `{capsules,environments,cgi,scripts}` where site-specific customizations are stored. These may be writable to permit

ALTAIR

N	Check	Meaning
		users to add items when using FlowTracer, or not writable and under admin control when only Accelerator/Monitor are in use.
16	WritableRegistry	Verifies that the Altair Accelerator registry directory `$VOVDIR/local/registry` is writable. This is needed so that the `vovproject create` command can add files there. These files contain the metadata used to locate, start, stop, etc. VOV projects.
17	vovrc	Verifies that the source-able Altair Accelerator setup files created during installation are present. These are `$VOVDIR/etc/vovrc.{csh,sh,tcl}`. The `vovrc.sh` file is especially important to Accelerator, because it is used to set the environment when starting vovtaskers on the farm hosts.

Clean Up

Now that you have performed a verified install of the platform-specific files for an added platform, you may be done.

However, if your early checking showed that there has been a software hotfix published in this release, for the added platform, then you are not done. You will need to go on to the chapter on how to "*Install Patch to Current Release*" and install the software hotfix for the platform.

Remember, investigate the state of the current release with regard to the published software patch level.

Has the software patch level already been applied to the current release? If so, then you will be installing the software patch for just the added platform. If not, they you will want to consider installing the software patch level to the entire release and have the software hotfix for the added platform be covered by doing the full software patch update.

Install a Patch to a Current Release

This Patch Installation scenario updates the set of installed files to replace existing files, with new versions that contain changes that fix known problems.

Software patches may be made available between releases in order to correct problems found in a version.

> ⚠ **Important:** A patch is always related to a specific released version, and should only be applied to that version.

There are two types of patch update files: **Required Updates** and **Recommended Updates**.

Required Updates are patches that should be applied by everyone as soon as they become available.

Recommended Updates contain a fix that is aimed at a particular feature and platform. It is published between patch releases to provide a fix to a specific issue for those who can't wait for the publication of the next Required Update patch. The Recommended Updates are ones that are optional and need only be applied if a problem that is fixed is a problem that is one that is affecting you.

Recommended Updates are only useful between patch releases. A patch release published after a Recommended Update will contain the fix in the Recommended Update.

Published software patches are cumulative. They are given a name to indicate their increasing level or position in a sequence. Only the most recent patch level is published on the website. Each new level replaces the older one and contains all the earlier software fixes, plus the new ones.

Installing a later software patch level onto a release with an earlier software patch level installed is an expected case. There is no problem with doing this.

These software update patches are relative to a major release. They are available for downloading in the same directory as the original download files for the release. If a software patch has been published for a release, then the software patch files will be listed on the download page in the Patches directory.

The download and install steps are different for a software patch than for a regular install. One difference is that the files are not downloaded into a temporary staging area - they are downloaded into a specific Patches directory within the release. A second difference is that the normal install script is not used to install them. Instead, a special patch install script is called once for each software patch file.

Here is the general description of what you will do when updating your release with software update patches.

1. Follow the software patch planning directions contained in the next section.
2. Download the needed software patches from the Altair website, with guidance from the download information contained in a later section.
3. Perform the patch software installation, which should require between 15 and 30 minutes depending on disk performance.
4. Validate the installation by setting your command line environment and validating the applied patch.

The following sections will lead you through the steps to update your current installation by installing software patches.

Plan to Install a Patch

This section contains a description of what to investigate as you consider installing software patches onto your current release. At the end of this activity, you will have a worklist of software patch files to install.

Check Available Patches

You can check the Altair website download page for the version of your current release to see if there are any published software patches.

1. From the Altair website, navigate to the desired product and click **Download Now**.

2. Log in using the account you set up as part of the pre-installation planning steps.

3. Click the link for the release version you have installed, such as **2019.01**.
    This opens a page listing all available updates, as well as any patches for that release.

    Any files in the directory labeled **Patches** are software patches you might consider installing. If no patches are available for the selected release, this directory will be empty.

4. Read the `README.1st.txt` to learn what is in the patch, and whether it is considered required or recommended. To install the patch, proceed with the following instructions.

Build a Worklist of Software Patches to Install

First, verify your current software patch level to ensure that you haven't already installed the latest patch. Complete the steps below:

1. From the command line, set up a working environment to access your current release, then run the `vovversion` command with the `-all` option.
    This will display the list of patches that have been installed.

    ```
    % vovversion -all
    2019.01
    VOVDIR=/opt/rtda/2019.01/linux
    Patch: p1 (linux)
    Patch: p2
    ```

    If this shows that your current release has the most recent software updates applied, then you can stop. Otherwise, continue with the steps in this chapter to create a worklist of software updates and install them to your release.

2. Identify the supported platforms in your release by running the following:

    ```
    % cd $VOVDIR
    % cd ..
    % ls -l
    ```

    The table below shows the platform-specific directory names and corresponding description used for the software patch files for these platforms. The exact descriptions used for the software patch files on the file download page may differ slightly.

3. Look for directories with these names and understand that when the directory exists, you should use the corresponding software patch file description in the table as a guide.

4. Look at the list on the download page to identify the exact file description that is similar.

5. Add this file to your worklist of patch files to download and install.

Directory	File Description
linux64	2019.01-patch-1:64-bit Linux
win64	2019.01-patch-1:32-bit Windows (for multi-platform installations)

    Each platform that is installed generates a potential software patch file to download and install.

6. Start building your worklist by adding the "Platform independent files" patch file to it, from the "Required update files" group.

7. For each platform supported in your release, look in the list of "Required update files" for a file description that refers to it. If found, add that file to your worklist.

8. To install the recommended patch, then for each platform supported in your release, look in the list of "Recommended update files" for a file description that refers to it. If found, add that file to your worklist.

At the end of this activity, you will have added the patch files to your worklist that relate to all the platforms supported in your release, plus the platform independent patch file.

You are now done building the worklist. This is the list of patch files that you you need to download and install.

Download the Software

To install the set of software update patches to your current release, you will need to download the patch files that you put into your worklist during the checklist step.

Use the Altair Accelerator Product Administrator Account

Log in to your computer, into your Altair Accelerator Administrator account. This is the user that was set up to do the standard install, and who owns all the files in the release.

In the following sections, examples will be written as if the user account named rtdamgr is the Altair Accelerator Administrator. If you are installing Accelerator, there is one step in the procedure that must be performed by the root user.

Create or Reuse a Special Patches Directory

1. Create a new directory or use an existing one to hold the downloaded software update patch files. This directory is in a special location within the current release and must be named patches. It must exist as a subdirectory at the top level of the current release, as a peer directory of the platform specific directories.

2. To create a new directory on UNIX:

a)   Assume that Altair Accelerator is installed at `/opt/rtda/` with the current release being 2019.01. The `/opt/rtda/2019.01/patches` directory is the one that is needed. If `/opt/rtda/2019.01/patches` directory exists, then use it. If the `patches` directory needs to be created, complete the following:

```
% cd /opt/rtda/2019.01
% mkdir patches
```

3.   To create a new directory on Windows:

a)   Assume that Altair Accelerator is installed at `D:\rtda` with the current release being 2019.01. The `D:\rtda\2019.01\patches` directory is the one that is needed. If `D:\rtda\2019.01\patches` directory exists, then use it. If the patches directory needs to be created, complete the following:

```
c:\> D:
d:\> cd \rtda\2019.01
d:\> mkdir  patches
```

Download the Files

1.   Access the Altair Marketplace and select the product to install.

2.   Log in using the account you set up as part of the pre-installation planning steps.

3.   Click the link for the release version you want, and download the appropriate files.

4.   Using your worklist, download each patch file on the list from the "patch" directory of the web page, and place that file into the patches directory. Note the exact file name of each downloaded software update patch file and write it down next to the descriptive name. You will need that precise file name for the next step in the process.

You do not need to expand the downloaded patch file you just placed within the patches directory. The install-patch process, which you will do next, expects the patch files to be those that were downloaded.

Install the Software

You may need to know the path to the current release in order to perform the operations described here. The path that is needed is the one that ends with the release number.

In the examples here, the current installation path will be shown as the `/opt/rtda/2019.01` directory. When you see this in examples, you should understand that it is referring to the path to your current release.

In the previous step, you downloaded all the needed software patch files and placed them into the special patches directory within the current release.

For each software patch file, you need to run the install-patch script, passing it the name of the patch file to install.

You must be in the current release when you run the install-patch script from its location within the scripts directory. You must give it the relative name of the patch file in the patches directory. The processing depends on this arrangement for it to work correctly.

1.   For a UNIX system:

a) Assume that Altair Accelerator is installed at `/opt/rtda/` with the current release in the subdirectory `2019.01`. You must be in the top level release directory when you run the install-patch script, `./scripts/PATCH.tcl`, for each software patch file.

2. UNIX: For Altair Accelerator Installations Only

After installing the patches to the release, you may have installed fresh versions of the `vovtaskerroot` executable for supported platforms.

If you are running Accelerator, you need to run a special process to fix these executables so that they work properly. This process changes the executables to have set_UID turned on so that Accelerator can deploy a job and have it run under the ID of the user who submits them.

You must be logged in as root or have *sudo* privileges to perform this step correctly in order to create the `vovtaskerroot` binary for each installed platform.

> 📄 **Note:** You must run the `SETTASKERUID.sh` script exactly as shown in the next example. The script will not work correctly unless it is executed from the specified directory.

The following script creates a set-UID executable called `vovtaskerroot` so that jobs can be run under the ID of the user who submits them. This script also sets the set-UID bit in `vovgetpasswd` for Linux.

a) To complete the update of software patches, you need to login as root and create the binary for `vovtaskerroot` by means of the following commands:

```
% /bin/su -
# cd <install_path>
# sh ./scripts/SETTASKERUID.sh
# exit
```

For example:

```
% /bin/su -
# cd /opt/rtda/2019.01
# sh ./scripts/SETTASKERUID.sh
# exit
```

3. For a Windows-only System:

It is possible to install software patches to a Windows-only installation.

If you have this installation, then your earlier steps identified the software patches files and you downloaded them into the sub folder patches within the release.

For this discussion, assume that Altair Accelerator is installed at `D:\rtda` with a subfolder `2019.01` holding the current release. Then, the `D:\rtda\2019.01\patches` folder has been created and contains the downloaded patch files.

a) The install-patch command for the Windows-only installation requires that you set up your command line environment for calling it. Set it up by running the `vovinit.bat` file in the release. It is in the `win64\bat` folder.

```
c:\> D:
d:\> cd \rtda\2019.01
d:\> win64\bat\vovinit
```

The command line environment needs to be set so that folders within the current release are on the PATH so that the programs in them will be found during the install-patch processing.

△ **ALTAIR**

In particular, the main program is named `PATCH.bat`, which is in the `win/bat` folder. By setting up the environment with `vovinit.bat` call, it will be found when called.

It in turn calls the command `Patch.tcl`, which is in the `scripts/` folder. This folder is added to the PATH by the `vovinit.bat` script, so that the command will be found.

The Windows-only install-patch technology depends on the `tar.exe` and `gunzip.exe` files being found on the PATH. During the original installation, those programs are placed into the `win64/instbin` folder, so they will be found by the install-patch process. Older install programs did not do this. If those files are absent from the `win64/instbin` folder, you can copy them there, or specify the path to where they are, using the -tar and -gunzip options when you start the install-patch.

With all those dependencies taken care of, you can run the install-patch command (`PATCH`) while you are in the top folder of the release.

This example will repeat the command to run `vovinit.bat` to set up the command line environment. Assume that the patches folder exists and that it contains the downloaded patch file.

```
c:\> D:
d:\> cd \rtda\2019.01
d:\> win64\bat\vovinit
d:\> PATCH patches\p1.common.patch.tar.gz
```

PATCH.tcl

```
Usage: ./scripts/PATCH.tcl [options] patch_file1 [patch_file 2 ...]

    Updates files in a Altair Engineering software hierarchy from patch files,
    archiving the modified files and updating the version file
    to indicate the applied patches.

  Options:
    -h       Show brief usage information
    -v       Increase verbosity

    -force   Force installation of a patch that is already installed.
    -tar     Specify path to tar executable
    -gunzip  Specify path to gunzip executable

    -undo <PATCHNAME>   Uninstall a named patch

  Examples:
    % ./scripts/PATCH.tcl patches/rollup1.common.patch.tar.gz
    % ./scripts/PATCH.tcl -force patches/rollup1.common.patch.tar.gz
    % ./scripts/PATCH.tcl patches/rollup2.*gz

    % ./scripts/PATCH.tcl -undo patches/save/rollup1.common.undo.tar.gz

  To show installed patches, use:
    % vovversion -install
```

Set Up Environment Variables

In order to run the software for validation, you need to set up your command line environment properly.

This includes changing your PATH environment variable so you can run the installed executables, and adding two environment variables that are used by the Altair Accelerator programs.

Set your command line environment by sourcing a setup file created by the installation. You will source the setup file that matches your platform and shell. As earlier, we are using the path `/opt/rtda/<release>` in our examples as the location of the installation.

Platform Type	Shell	Command to Source File
UNIX	csh tcsh	source `/opt/rtda/2020.1/ common/etc/vovrc.csh`
	bash sh ksh zsh	`. /opt/rtda/2020.1/common/ etc/vovrc.sh`
Windows	DOS	`c:\rtda\2020.1\win64\bat \vovinit.bat`

Each one of these files defines the variable VOVDIR, which will be used in the rest of the installation documentation and which must be set for proper operation of Altair Accelerator software.

The environment variable VOVDIR always points to the platform specific directory, so that, for example, the executables can be found in `$VOVDIR/bin`, the documentation can be found in `$VOVDIR/doc`, and the local information can be found in `$VOVDIR/local`.

Verify the Installation

Quick Verification

After completing the installation and setting up your shell environment, you can do a quick verify that the install was successful by running a program that reports on the installed version. This confirms that the installed program is accessible on your PATH and that it is the version you expect.

Use the `vovversion` program to confirm you are accessing the desired version of the product.

Running this command should give you the expected version number, and the expected location where you have installed the product.

It is also a good practice to check that your environment variables have been set up correctly:

```
> source /<install_path>/<version>/<platform>/common/etc/vovrc.csh
% vovversion -all
2016.09
VOVDIR=/opt/rtda/2016.09/linux
Patch: p1 (linux)
```

ALTAIR

```
Patch: p2
```

You should see the expected version number, and the expected location where you have installed the product. You should also see the list of installed patches that matches the patches you've just applied.

You can also view the installation log file in `$VOVDIR/..` that was created during the install process.

vovversion

Print the VOV version. Optionally show more details.

```
vovversion: Usage Message

DESCRIPTION:
    Print the VOV version. Optionally show more details.

USAGE:
    vovversion [options]

OPTIONS:
    -help      -- Print this message
    -all       -- Print all available version information
    -clients   -- List clients and their version numbers
    -install   -- Print the VOV version with patch names, if any
    -patches   -- Print patch names, if any
    -project   -- Print the version of vovserver associated with the
                  current project
    -short     -- Print VOV version with short patch name, if any
    -verify    -- Ensure server and client versions match

    [Can abbreviate options provided they are unambiguous]
```

Visual Verification

You can navigate around the install area to make sure that there are subdirectories for each platform that you requested to be installed.

When you set up your environment in the previous section, you set the environment variable VOVDIR to be the path to your platform specific files. You can navigate to that area of the installed files to look around to see the other platform files.

You can navigate to see the platform executables in the install area of the file system that were just finished.

```
% cd $VOVDIR
% cd ..
% ls
# expect to see directory named for the added platform
```

You can see the subdirectories there for each platform that was installed. Verify that there is a subdirectory there for each platform you requested to be installed.

You can also view the installation log file in `$VOVDIR/../install<datestamp>.log` that was created during the install process.

Rigorous Verification

After you have confirmed that the correct installation is accessible, you can do a rigorous check on the state of the install using a program that checks on a variety of aspects of your installation.

This will report on deeper issues with regard to the install and the configuration and the environment.

Use the `vovcheck` program as solid verification of the Altair Accelerator product installation.

You should run this program after you install a new version of Altair Accelerator software, or as needed to help with troubleshooting.

vovcheck

Basic check of configuration of VOV installation, configuration, and project setup.

```
vovcheck: Usage Message

      DESCRIPTION:
      Basic check of configuration of VOV installation,
      configuration, and project setup.

      USAGE:
      % vovcheck [options]

      OPTIONS:
      -h                               -- This help.
      -report  <file>                  -- Name report file.
      -installation                    -- Check the installation (default).
      -project                         -- Check the project.
      -t                               -- Turn on Tcl tracing (for debugging).
      -v                               -- Increase verbosity.

      FANCY OPTIONS:
      -class   {NP|PD|PS|PU}          -- Specify class of tests to perform.
      -- NP = No project    (check installation)
      -- PD = Project down
      -- PS = Project sick
      -- PU = Project up    (check project)

      EXAMPLES:
      % vovcheck
      % vovcheck -installation
      % vovcheck -project
      % vovcheck -report myreport.txt
```

Comments

As mentioned at the end of the `vovcheck` output above, the program creates a more detailed report in a file in the tmp directory. You **must** look at this report in order to interpret the results of the checking.

The output shows the focus of each test, along with the products for which the test is applicable and the result of the test. Not all checks are meaningful for all products or all environments. For your specific environment, a WARN or ERROR may be the expected and correct result.

△ ALTAIR

The standard output will show WARN and ERROR results as a way to alert you to look within the report file to read the details about `vovcheck` has noticed.

The Products column indicates the product for which the test applies. You can ignore a WARN or ERROR result for a test that is checking an aspect of a product you are not using.

Look at the report file to see the details about each test's results. The report file has clear and detailed explanations about was expected and what was noticed, that can help you interpret the report. It also provides a hint on what to fix or change so that the `vovcheck` program will not produce the alert when doing the test.

Deep Verification Checks

The following table gives additional detail about the function of each check performed by `vovcheck`.

N	Check	Meaning
1	BasicVariables	Checks that environment variables VOVDIR, VOVARCH, VOV_HOST_NAME, VOV_PROJECT_NAME are set. Needed by all VOV projects.
2	CwdPermissions	Checks that you have read/write permission for the current directory, needed to run jobs there in Accelerator
3	EnvBase	Verifies that the BASE environment exists and can be entered, and that a few basic UNIX and Altair Accelerator programs are in the PATH in that environment.
4	GuiCustomization	Checks for an old filename for GUI customization, and if present advises of the correct location.
5	HostPortConflicts	Checks the registry and advises if more than one project on the same host uses the same TCP port. Only one of such projects may be started, since the OS only allows one process to listen on a given port at a time.
6	Installation	Verifies that the Linux OS version is compatible with the Altair Accelerator libraries for runtime tracing.
7	Check	Checks for an old filename for `Make.tcl` defaults, and if present advises of the correct location.
9	RshSsh	Tries to verify whether there is a remote shell command set up that works to the current host without a password. The owner of Accelerator must be able to remote-shell from the vovserver host to the farm hosts to start the vovtasker.
10	SecurityPermissions	Checks the permissions on `$VOVDIR/etc/cgi` and other directories and files in the installation.

N	Check	Meaning
11	TaskerRoot	Verifies that `$VOVDIR/bin/vovtaskerroot` exists and has the desired ownership and permissions. This is needed in Accelerator so that jobs can run in the account of their submitter, so as to have the correct privileges with respect to the submitter's files.
12	SshSetup	Verifies that the `~/.ssh` directory has correct permissions to permit ssh to operate. This directory must be owned by the user or root, and not be readable by anyone else.
13	UsrTmp	Verifies that the directory `/usr/tmp` exists and is writable. Some Linux distributions (e.g. Ubuntu) do not include this by default.
14	VovPermissions	Verifies ownership and permissions of files in the `bin` and `scripts` directory of the Altair Accelerator installation.
15	WritableLocal	Checks permissions on directories `$VOVDIR/local/ {capsules,environments,cgi,scripts}` where site-specific customizations are stored. These may be writable to permit users to add items when using FlowTracer, or not writable and under admin control when only Accelerator/Monitor are in use.
16	WritableRegistry	Verifies that the Altair Accelerator registry directory `$VOVDIR/ local/registry` is writable. This is needed so that the `vovproject create` command can add files there. These files contain the metadata used to locate, start, stop, etc. VOV projects.
17	vovrc	Verifies that the source-able Altair Accelerator setup files created during installation are present. These are `$VOVDIR/etc/ vovrc.{csh,sh,tcl}`. The `vovrc.sh` file is especially important to Accelerator, because it is used to set the environment when starting vovtaskers on the farm hosts.

Uninstall a Software Update Patch

During the install-patch processing, the original files were saved under the directory `patches/save/` in a backup file that has `.undo` in its name.

There is currently no automatic process to uninstall a patch using those saved files. The files are there, and you can use them to restore to the previous version that was there when the software update patch was applied. This step is done manually, copying the files back into their location.

△ ALTAIR

Upgrade to a New Release

This upgrade installation scenario expands an existing release by installing a separate set of files that implement a new release of the product.

When you upgrade to a new release, you will need to perform a full install of the new software, and carry forward the files you have changed locally in the current release.

You will do a non-destructive installation of the new release by installing it into a different area than the current release. This allows you to let the current release continue to run until the new release has been validated to be ready for production.

1. Follow the upgrade planning directions contained in the next section.

2. Download the software to install from the Altair Community website, with guidance from the download information contained in a later section.

3. Perform the upgrade software installation, which should require between 15 and 30 minutes depending on disk performance.

4. Validate the upgrade by setting your command line environment and running the validation steps.

The following sections will lead you through the steps listed above.

Plan the Upgrade

You should read the Release Notes for the version you are upgrading to in order to understand the new features in the version, and to identify any special activity that might be needed in order to do the upgrade.

It is best practice to install the upgrade and to use it on small projects before moving it into production.

This installation guide describes placing the upgrade files into a separate location from the current release to support the ability to have both the current version and the upgrade version available at the same time.

Download the Software

After the pre-upgrade checklist is complete, you are ready to download the software and prepare the downloaded files in order to do the upgrade to a new release.

Use the Altair Accelerator Product Administrator Account

Log in to your computer, into your Altair Accelerator Administrator account. This is the user that was set up to do the standard install, and who owns all the files in the release.

In the following sections, examples will be written as if the user account named rtdamgr is the Altair Accelerator Administrator. If you are installing Accelerator, there is one step in the procedure that must be performed by the root user.

Create or Reuse a Temporary Download Staging Area

Create a staging area to hold the downloaded files. It must be different from the directory where you plan to install the Altair Accelerator software. The staging area may be removed after installation.

For example, create the staging area directory as user accel_mgr.

   a)   On UNIX:

```
% mkdir /usr/tmp/altair_download
```

   b)   On Windows:

```
C:\> md  c:\temp\altair_download
```

> 📝   **Note:** Location path names must not contain any spaces.

Download the Files

1.   Access the Altair Marketplace and select the product to install.

2.   Log in using the account you set up as part of the pre-installation planning steps.

3.   Click the link for the release version you want, and download the appropriate files.

4.   For a UNIX installation, complete the steps below:

   a)   Download the `Common.tar` file into the staging directory you just created.

   b)   Download the platform-specific file(s) for each platform type that you will be using.

   c)   Expand the archives with tar extract.

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf <platform>.tar    # For each platform
```

   For example:

```
% cd /usr/tmp/altair_download
% tar xf common.tar
% tar xf linux.tar
```

5.   Access the Altair Marketplace and select the product to install.

6.   Log in using the account you set up as part of the pre-installation planning steps.

7.   Click the link for the release version you want, and download the appropriate files.

8.   For a Windows installation, complete the steps below:

   a)   When Windows is one of the platforms used in a networked multi-platform situation, you should download the `win64.tar` file into the staging area.

> 📝   **Note:** The name "win64" is a generic name for all versions of 64-bit Windows.

△ALTAIR

b)   Expand the archives with tar extract:

```
% cd /usr/tmp/altair_download
% tar xf win64.tar
```

c)   Expand this zip file into the staging area using Windows Explorer (via right-clicking on the file) or any utility that works with zip files.

Install the Software

You will need to know the path to the current release in order to perform the operations described here. The path that is needed is the one that ends with the release number.

In the examples here, the current installation path will be shown as the `/opt/rtda/2019.01` directory. When you see this in examples, you should understand that it is referring to the path to your current installation. The upgrade examples will be upgrading the 2019.01 release to the 2020.1 release. The upgrade install will be installing files into the `/opt/rtda/2019.01` directory.

Installation via the Graphical User Interface

Start the Installer Graphical User Interface for your platform as shown below. The GUI installer uses the X-Windows technology for the Graphical User Interface. If you are running on a remote UNIX system in X-windows, be sure your *DISPLAY* environment variable is set properly.
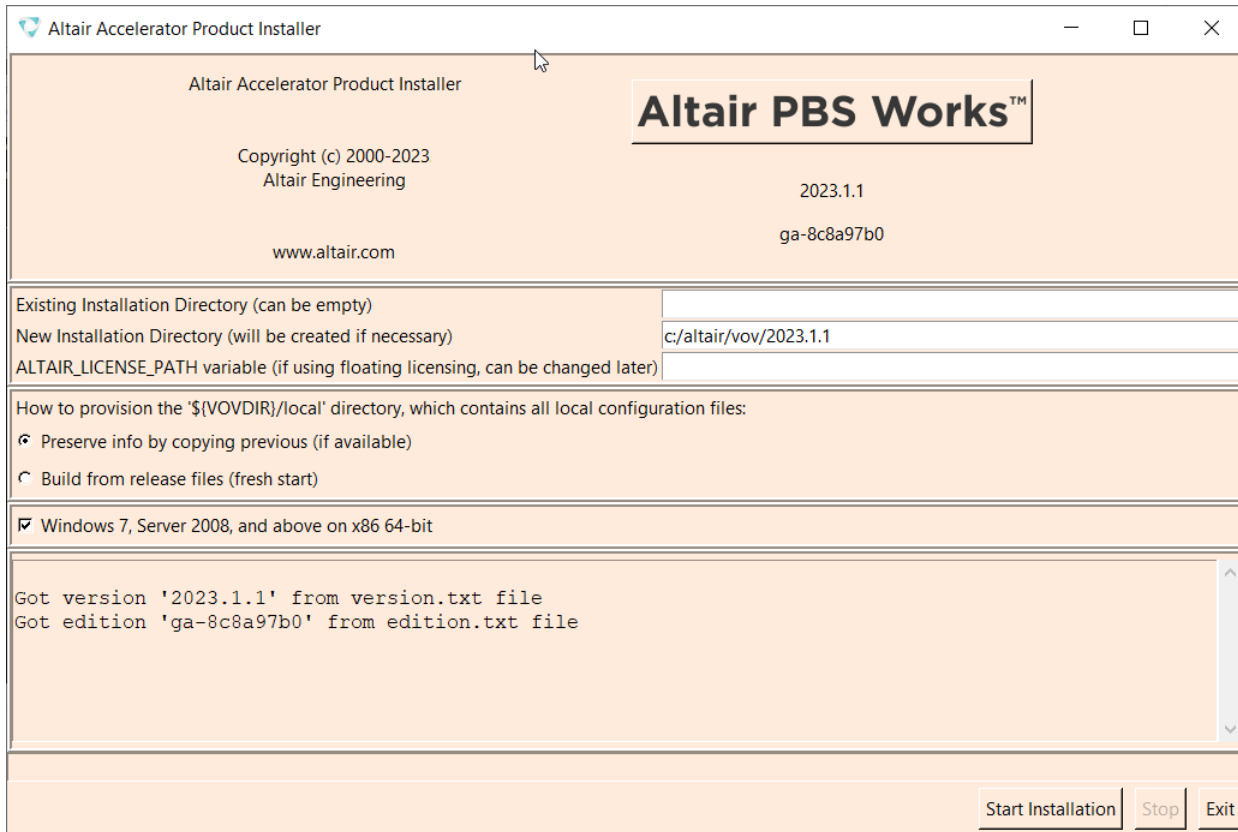
Figure 6:

1. Change the working directory to the staging directory where you downloaded the tar files and extracted them, and run the GUI install script `install.sh`.

    a) On UNIX:

    ```
    % cd /usr/tmp/altair_download
    % ./install.sh
    ```

    b) On Windows:

    ```
    C:\> cd  c:\temp\altair_download
    C:\> install.bat
    ```

    The installation dialog appears. Since this is an update to an existing release, the **Existing Installation Directory** and **New Installation Directory** fields should be the same value, and that value should be the location of the current installation. The value of the ALTAIR_LICENSE_PATH variable field will be obtained from the current installation and does not need to be changed.

2. Use the first section to tell the installer where to install the Altair Accelerator software and where it will find the Altair Accelerator license server.

3. Use the second section to setup the new release so that locally changed files are placed into the new release. Because this is an update of an existing release, you want the install to do nothing with regard to your local directory. You should choose **Preserve info by copying previous**. This tells the installer to copy the "local" directory file in the current release to itself, which is effectively to do nothing.

4. In the Platforms section, indicate which platform you are adding during this install. Clear all the other platforms that have already been installed.

    This directs the installer to place the binary executables for these machine types into the installed directory structure. The files will be placed within subdirectories named for the platform. The installed files for each platform will come from the staging area where you have downloaded and extracted them.

    If you are installing on a UNIX system, you could select the Windows platform here if the Windows file (`win64.tar`) was extracted into the staging area. The Window binaries will be installed on the UNIX machine as files within the win64 platform subdirectory.

    The intention of platform selection is to provide executable files in the file system on this machine, for the added platform in the network. This enables the remote machine to access the executables of their type by way of NFS, without needing to do an install of files on any other machines.

5. Click **Start installation** to start the installation. It normally takes a few minutes to run, depending mostly on the speed of your disk.

    During the install, the program will write messages into the bottom section of the dialog, which is a scrollable region. The messages are also written to a log file on disk in the new installation directory, named `install<timestamp>.log`.

6. When the installation is done, click **Exit**.

7. You can review the install log for errors and warnings. The last few sentences of the log file remind you to do something if you are installing Accelerator. This subject is explained in *For Accelerator Installation Only*.

Installation via a Batch Script

On UNIX, you can use a batch install script instead of the GUI version to install the files.

Run the command `./install.sh`

The values that control the installation are passed into the batch script as command line parameters, except for the "Existing Installation Directory". It gets the value for this from the *$VOVDIR* environment variable as it runs.

Because this is an update install to add a new platform to a previously installed release, the option -local copy should be used for specifying the handling of the "local" directory. Give it a value that corresponds to your policy on how the install should carry forward the local files from your current release into the upgrade.

*   `create`: Don't carry files forward. Create new starting state for files.
*   `copy`: Copy files from current release into upgrade.
*   `link`: Create symbolic link in upgrade to release-independent local directory.

> 📝 **Note:** There is not an option value to cause a symbolic link in upgrade to the local directory in the current release.

Here is the logical description of calling the batch install for the scenario of installing an upgrade:

```
% # VOVDIR is set for current release
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/<version> -local link -alm 6200@almsrv -platforms
  '<platform list>'
```

Here is a specific example of installing for the first time that compares to the example used in the GUI dialog example above.

```
%  source /opt/altair/vov/<version>/etc/vovrc.csh
```

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/<version> -local link -alm 6200@almsrv –platforms
  'linux64 linux'
```

install.sh

This script supports batch installation on Linux as well as Windows. Run from the directory where the installation files were extracted.

By default, the installer will run in graphical mode. Use the -batch option to suppress the GUI and run in batch mode.

Usage

```
./install.sh [INSTALLPATH][OPTIONS]
```

Options

Use the options below for non-interactive installation.

-alm PORT@HOST	Value to use for ALTAIR_LICENSE_PATH.
-batch	Install in batch mode. If INSTALLPATH is not passed as an argument, the default installation path for Linux is /opt/altair/vov/1212.10 and for Windows is \opt\altair \vov\1212.10.
-local HANDLING	Specify handling of the 'local' directory where HANDLING is one of: • copy (copy previous installation's local) • create (create new from installation stubs) • link (link to previous installation's local) • linkext (link to a version-independent local) The default handling for Linux is linkext, which results in the local directory being placed in the parent of the installation and a symbolic link created inside the installation to point to it. This facilitates the reuse of the local directory for all installed versions. The default handling for Windows is copy. Only copy and create are supported on Windows.
-nocommon	Used when installing additional platforms after an initial installation has been performed, to prevent overwriting of the common directory. Ignored on Windows.
-noconfirm	Do not prompt for confirmation to begin the installation process when running in batch mode.
-platforms LIST	A quoted, space-separated list of platforms to install. Supported platforms are: armv8, linux64, and win64.
-previous PATH	Path to previous installation, if applicable. Defaults to the value of $VOVDIR if set in the calling environment. Otherwise, defaults to an empty string.

Examples

```
./install.sh /opt/altair/vov/<version>
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -alm 6200@almsrv -batch
./install.sh /opt/altair/vov/<version> -local create -alm 6200@almsrv
./install.sh /opt/altair/vov/<version> -local link -platforms 'linux64 win64'
```

Because this is an update install to add a new platform to a previously installed release, the options -nocommon and -local copy should be used. The `copy` choice for the local parameter will cause the process to copy the local files onto themselves, creating a non-destructive no-op operation.

Here is the logical description of calling the batch install for the scenario of adding platform-specific files for a new platform.

```
% cd /opt/altair/vov/2023.1.0
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy <port@host> -platforms
  '<added-platform>'
```

Here is a specific example of doing an update by running the batch install to add the platform Linux to the release.

```
% cd /usr/tmp/altair_download
% ./install.sh /opt/altair/vov/2023.1.0 -nocommon -local copy -alm 6200@almsrv; -
platforms 'linux64'
```

For Accelerator Installation Only

Accelerator requires one extra step during the install. This step creates a set_UID executable that Accelerator uses so that jobs can run under the ID of the user who submits them. This step requires root capability to have it run correctly.

If you do not have root privileges, or if you are performing an evaluation, you can postpone this step. You can still run the software in single user mode, but you will not be able to run it correctly with multiple users due to permission issues.

You must be logged in as root or have *sudo* privileges to perform this step correctly to create the `vovtaskerroot` binary for each installed platform.

> 📝 **Note:** You must run the `SETTASKERUID.sh` script exactly as shown in the next example. The script will not work correctly unless it is executed from the specified directory.

The following script creates a set-UID executable called vovtaskerroot so that jobs can be run under the ID of the user who submits them. This script also sets the set-UID bit in `vovgetpasswd` for Linux.

To complete the installation, you need to login as root and install the binary for `vovtaskerroot` by means of the following commands:

```
% /bin/su -
# cd <install_path><version>
# sh ./scripts/SETTASKERUID.sh
# exit
```

For example:

```
% /bin/su -
# cd /opt/rtda/2020.1
# sh ./scripts/SETTASKERUID.sh
```

△ **ALTAIR**

```
# exit
```

At this point, installation is complete.

Set Up Environment Variables

In order to run the software for validation, you need to set up your command line environment properly.

This includes changing your PATH environment variable so you can run the installed executables, and adding two environment variables that are used by the Altair Accelerator programs.

Set your command line environment by sourcing a setup file created by the installation. You will source the setup file that matches your platform and shell. As earlier, we are using the path `/opt/rtda/<release>` in our examples as the location of the installation.

Platform Type	Shell	Command to Source File
UNIX	csh tcsh	source `/opt/rtda/2020.1/common/etc/vovrc.csh`
	bash sh ksh zsh	. `/opt/rtda/2020.1/common/etc/vovrc.sh`
Windows	DOS	`c:\rtda\2020.1\win64\bat\vovinit.bat`

Each one of these files defines the variable VOVDIR, which will be used in the rest of the installation documentation and which must be set for proper operation of Altair Accelerator software.

The environment variable VOVDIR always points to the platform specific directory, so that, for example, the executables can be found in `$VOVDIR/bin`, the documentation can be found in `$VOVDIR/doc`, and the local information can be found in `$VOVDIR/local`.

Verify the Installation

Before you can verify the installation, you will need to configure the proper licensing for the new release. Follow the instructions in the "Licensing" chapter to enable execution of the newly installed upgrade version of Altair Accelerator.

Quick Verification

After completing the installation and setting up your shell environment, you can do a quick verify that the install was successful by running a program that reports on the installed version. This confirms that the installed program is accessible on your PATH and that it is the version you expect.

Use the `vovversion` program to confirm you are accessing the desired version of the product.

Running this command should give you the expected version number, and the expected location where you have installed the product.

It is also a good practice to check that your environment variables have been set up correctly:

```
> source /<install_path>/<version>/<platform>/common/etc/vovrc.csh
% vovversion -all
2020.1
VOVDIR=/opt/rtda/2020.1/linux
```

You should see the expected version number, and the expected location where you have installed the product.

vovversion

Print the VOV version. Optionally show more details.

```
vovversion: Usage Message

DESCRIPTION:
    Print the VOV version. Optionally show more details.

USAGE:
    vovversion [options]

OPTIONS:
    -help       -- Print this message
    -all        -- Print all available version information
    -clients    -- List clients and their version numbers
    -install    -- Print the VOV version with patch names, if any
    -patches    -- Print patch names, if any
    -project    -- Print the version of vovserver associated with the
                   current project
    -short      -- Print VOV version with short patch name, if any
    -verify     -- Ensure server and client versions match

    [Can abbreviate options provided they are unambiguous]
```

Visual Verification

You can navigate around the install area to make sure that there are subdirectories for each platform that you requested to be installed.

When you set up your environment in the previous section, you set the environment variable VOVDIR to be the path to your platform specific files. You can navigate to that area of the installed files to look around to see the other platform files.

You can navigate to see the platform executables in the install area of the file system that were just finished.

```
% cd $VOVDIR
% cd ..
% ls
# expect to see directory named for the added platform
```

You can see the subdirectories there for each platform that was installed. Verify that there is a subdirectory there for each platform you requested to be installed.

You can also view the installation log file in `$VOVDIR/../install<datestamp>.log` that was created during the install process.

Rigorous Verification

After you have confirmed that the correct installation is accessible, you can do a rigorous check on the state of the install using a program that checks on a variety of aspects of your installation.

This will report on deeper issues with regard to the install and the configuration and the environment.

Use the `vovcheck` program as solid verification of the Altair Accelerator product installation.

You should run this program after you install a new version of Altair Accelerator software, or as needed to help with troubleshooting.

vovcheck

Basic check of configuration of VOV installation, configuration, and project setup.

```
vovcheck: Usage Message

     DESCRIPTION:
     Basic check of configuration of VOV installation,
     configuration, and project setup.

     USAGE:
     % vovcheck [options]

     OPTIONS:
     -h                          -- This help.
     -report  <file>             -- Name report file.
     -installation               -- Check the installation (default).
     -project                    -- Check the project.
     -t                          -- Turn on Tcl tracing (for debugging).
     -v                          -- Increase verbosity.

     FANCY OPTIONS:
     -class  {NP|PD|PS|PU}       -- Specify class of tests to perform.
     -- NP = No project   (check installation)
     -- PD = Project down
     -- PS = Project sick
     -- PU = Project up   (check project)

     EXAMPLES:
     % vovcheck
     % vovcheck -installation
     % vovcheck -project
     % vovcheck -report myreport.txt
```

Comments

As mentioned at the end of the `vovcheck` output above, the program creates a more detailed report in a file in the tmp directory. You **must** look at this report in order to interpret the results of the checking.

The output shows the focus of each test, along with the products for which the test is applicable and the result of the test. Not all checks are meaningful for all products or all environments. For your specific environment, a WARN or ERROR may be the expected and correct result.

The standard output will show WARN and ERROR results as a way to alert you to look within the report file to read the details about `vovcheck` has noticed.

The Products column indicates the product for which the test applies. You can ignore a WARN or ERROR result for a test that is checking an aspect of a product you are not using.

Look at the report file to see the details about each test's results. The report file has clear and detailed explanations about was expected and what was noticed, that can help you interpret the report. It also provides a hint on what to fix or change so that the `vovcheck` program will not produce the alert when doing the test.

Deep Verification Checks

The following table gives additional detail about the function of each check performed by `vovcheck`.

N	Check	Meaning
1	BasicVariables	Checks that environment variables VOVDIR, VOVARCH, VOV_HOST_NAME, VOV_PROJECT_NAME are set. Needed by all VOV projects.
2	CwdPermissions	Checks that you have read/write permission for the current directory, needed to run jobs there in Accelerator
3	EnvBase	Verifies that the BASE environment exists and can be entered, and that a few basic UNIX and Altair Accelerator programs are in the PATH in that environment.
4	GuiCustomization	Checks for an old filename for GUI customization, and if present advises of the correct location.
5	HostPortConflicts	Checks the registry and advises if more than one project on the same host uses the same TCP port. Only one of such projects may be started, since the OS only allows one process to listen on a given port at a time.
6	Installation	Verifies that the Linux OS version is compatible with the Altair Accelerator libraries for runtime tracing.
7	Check	Checks for an old filename for `Make.tcl` defaults, and if present advises of the correct location.

N	Check	Meaning
9	RshSsh	Tries to verify whether there is a remote shell command set up that works to the current host without a password. The owner of Accelerator must be able to remote-shell from the vovserver host to the farm hosts to start the vovtasker.
10	SecurityPermissions	Checks the permissions on `$VOVDIR/etc/cgi` and other directories and files in the installation.
11	TaskerRoot	Verifies that `$VOVDIR/bin/vovtaskerroot` exists and has the desired ownership and permissions. This is needed in Accelerator so that jobs can run in the account of their submitter, so as to have the correct privileges with respect to the submitter's files.
12	SshSetup	Verifies that the `~/.ssh` directory has correct permissions to permit ssh to operate. This directory must be owned by the user or root, and not be readable by anyone else.
13	UsrTmp	Verifies that the directory `/usr/tmp` exists and is writable. Some Linux distributions (e.g. Ubuntu) do not include this by default.
14	VovPermissions	Verifies ownership and permissions of files in the `bin` and `scripts` directory of the Altair Accelerator installation.
15	WritableLocal	Checks permissions on directories `$VOVDIR/local/``{capsules,environments,cgi,scripts}` where site-specific customizations are stored. These may be writable to permit users to add items when using FlowTracer, or not writable and under admin control when only Accelerator/Monitor are in use.
16	WritableRegistry	Verifies that the Altair Accelerator registry directory `$VOVDIR/``local/registry` is writable. This is needed so that the `vovproject create` command can add files there. These files contain the metadata used to locate, start, stop, etc. VOV projects.
17	vovrc	Verifies that the source-able Altair Accelerator setup files created during installation are present. These are `$VOVDIR/etc/``vovrc.{csh,sh,tcl}`. The `vovrc.sh` file is especially important to Accelerator, because it is used to set the environment when starting vovtaskers on the farm hosts.

△ALTAIR

Clean Up

Access the Help Documentation

Now that you've installed the product, you will want to read the product-specific documentation. Details about various ways to read the documentation is available in the Use Altair Accelerator Package Help topic.

Remove Downloaded Files

Now that the install of the downloaded files has been completed, you can remove the downloaded and expanded files from the staging area.

The only benefit in leaving them on your disk is if you may need to update the installation with additional platform-specific files.

An update of the existing release to add platform-specific is needed if you add a machine type to your network that was not in place during this install process. In that case, the downloaded and extracted common files will need to exist to do that update. The platform-specific files that are now already installed will not be needed during that add-platform update step. You can leave the common files on disk, or delete them now and download them again when you need them.

Special Considerations for Accelerator

This section provides instructions to change the software version on which Accelerator runs.

Often you will do this to switch all the components (vovserver, vovtasker, vovsh) to a newer version, but you may also switch the vovserver and vovtasker versions separately. For the discussion below, we use 'current' to refer to the version before the change, and 'new' for the version after it.

You will find instructions for three common scenarios below.

- **Cold Upgrade**, used when you are also shutting down the Accelerator computers, as for making operating system or hardware upgrades. All components will be on the new version.
- **Hot Upgrade**, used to replace the vovserver while allowing jobs to continue to run on the current vovtasker version. vovserver will be on the new version, but vovtaskers stay on the current version.
- **Rolling Hot Upgrade**, used to allow the current software version to run jobs after the new version is installed and starts running jobs. This method is recommended when the normal workload consists of large-scale projects.

> ⚠ **Important:** The upgrade process includes downloading software archive files from the Altair website, which are then unpacked or untarred into a temporary installation directory. The installer will ask for a destination directory.

When choosing the destination for the downloaded software, be careful to select a non-destructive installation path. This ensures the current Altair Accelerator software is not overwritten or otherwise damaged before the new software is installed and activated. Usually the new version will be installed as a sibling of the current version, that is, `/tools/rtda/2019.01` for the current and `/tools/rtda/2020.1` for the new.

Cold Upgrade

This method is usually implemented with a physical overhaul, a part of a major IT event. This kills all running jobs, which can be highly disruptive.

To reduce the impact, Accelerator can be instructed to stop accepting and dispatching new jobs for a period prior to the shutdown event, enabling some of the running jobs to complete. How many jobs will complete depends on the jobs' duration and the time allowed before shutdown.

1. Download the Accelerator upgrade software.

2. Install the new software.

3. Using the new version, create a separate, temporary test queue (to validate the new version while production continues).

4. Validate the installation using the test queue that you created.

5. Schedule and announce the upgrade.

6. If you have multiple Accelerator queues, it is recommended that you set thNC_QUEUE to the name of the queue that is undergoing maintenance. This helps prevents accidentally shutting down the wrong queue. Use the command:

   ```
   setenv NC_QUEUE vncNameOfQueue
   ```

7. Optional: Suspend the vovtaskers with the command below. This command puts the vovtaskers in the SUSP state; running jobs will continue, but vovtasker will not accept new jobs. When vovtasker completes its current set of jobs, it will exit.

   ```
   nc cmd vovtaskermgr stop
   ```

8. Optional: At the point of the scheduled downtime, document the IDs of running jobs as those jobs will be terminated forcefully. This list can be used to inform users that their jobs were terminated by the maintenance event.

   ```
   nc list -r -a -O @ID@ @USER@ @COMMAND@
   ```

9. Optional: To automatically identify jobs when the queue is restarted, place the jobs in a special set.
   Example:

   ```
   nc cmd vovset create "ImpactedByQueueRestart" "isjob status==RETRACING"
   ```

10. Optional: Terminate these running jobs with the -force option, which should terminate the remaining taskers within a few minutes.
    Example:

    ```
    vovtaskermgr stop -force -all
    ```

11. Stop the vovserver of the queue with the command `ncmgr stop`

12. Proceed with any necessary infrastructure maintenance.

13. Restart the queue.

14. Ensure that your shell is configured to support the correct number of file descriptors.

    > 📝 **Note:** This value cannot be changed after starting the queue.

15. Ensure that you are pointing to the appropriate version of Accelerator with the command `which nc`.

16. Start the queue with the command `ncmgr start`.
    A confirmation dialog will open.

17. Review the parameters carefully (especially number of file descriptors) before replying 'yes'. (Starting the queue will automatically start the taskers but this will take some time, be patient.)

18. Validate that jobs are dispatching normally.

> 📝 **Note:** Restarting a large compute farm will take several minutes.

19. Optional: Re-queue the jobs that were impacted by the shutdown. Use the following command:

```
nc rerun -f -set ImpactedByQueueRestart
```

Hot Upgrade

This method reduces upgrade impact -- less downtime for productivity, less obtrusive for the users. The vovserver and vovtaskers are moved separately to the new version.

With this method, vovtaskers with running jobs are temporarily renamed, by appending `_stopped_<timestamp>` to their regular names. This allows active jobs to finish, and restarted vovtaskers to use the regular names.

> 📝 **Note:** If you are using a large value of VOV_RELIABLE_TIMEOUT you should suspend job dispatch (see Suspend Accelerator Job Dispatch) for some time, 30-60s, before initiating the cutover.

Follow the steps below for the preparation portion of the procedure:

1. Download the Accelerator upgrade software.

2. Install the upgrade software.

3. Create a separate, temporary test queue (to validate installation).

4. Validate the installation by starting the temporary test queue with a few vovtaskers on the new version and running test jobs.

Cutover process:

5. Notify your Accelerator users of the scheduled upgrade.

6. Get a shell as the Accelerator owner on the Accelerator vovserver host with current-version Altair Accelerator commands in the PATH.

7. Suspend job dispatching as in Suspend Accelerator Job Dispatch.

8. Stop the vovserver with the following command:

```
ncmgr -q nc-queue stop -freeze
```

> 📝 **Note:** The vovtaskers with jobs will continue to run and will be renamed. The ones that have no jobs will exit.

9. Ensure that the shell you are using has a sufficiently high limit for file descriptors.

10. Source the Altair Accelerator setup file for the new version, or use a separate shell set up for the new version.

11. Restart the vovserver with the new software version.

```
ncmgr -queue nc-queue start
```

△ ALTAIR

12. Optional: Restart a subset of the taskers. Use the following command:

```
nc cmd vovtaskermgr stop tasker1...taskerN
```

13. After the vovtaskers have finished their jobs and exited, run the following command:

```
nc cmd vovtaskermgr start tasker1...taskerN
```

Rolling Hot Upgrade

In this method, a new queue is brought up with the new version of the software; the previously existing queue remains fully functional.

This approach permits both administrators and users to thoroughly test the new version of the software and selectively move workloads to the new queue. The vovtasker machines are progressively moved from one queue to another by suspending them on the old queue, allowing jobs to *drain off* (complete) and then restarting the tasker on the new queue. The advantage of this approach is reducing risk: The old queue remains available during transition. Some disadvantages: The queue host:port and possibly queue name changes is not transparent to users; the entire upgrade process may take many days; you may need additional licenses during the transition interval.

1. Download the Accelerator upgrade software.
2. Install the new upgrade software.
3. Create a new queue name.
4. Start the queue (created in step 1) on either the existing host machine for Accelerator or a new host.
5. Configure the new queue with the various policy and job class settings.
6. Suspend a number of the taskers on the old queue and when they have terminated (when existing jobs have completed), add these taskers to the new queue.
7. Test and verify the software upgrade; transfer other remaining taskers after verifying new Accelerator.
8. After all taskers have been transferred, stop the old queue.

Appendix

This section describes how to make Accelerator stop accepting new jobs, and also to suspend Accelerator job dispatch.

Stop Accelerator Job Acceptance

When preparing to stop Accelerator, you may wish to stop accepting new jobs so the queued jobs can drain before the shutdown.

You can do this by implementing the `vnc_policy.tcl` file shown below. Since `vnc_policy.tcl` is interpreted inline during the `nc run` command, this policy refuses new jobs with an informative message.

```
proc VncPolicyValidateResources { resList } {
#
# This policy prevents submission of new jobs with a message
#
VovWarning "Job not not submitted; NC closed by admin"
exit
return $resList; # not reached
}
```

Suspending Accelerator Job Dispatch

You can suspend job scheduling without stopping the Accelerator vovserver.

```
% nc [-q qname]  vovservermgr scheduler suspend
```

This will suspend the dispatching of jobs.

To resume, enter

```
% nc [-q qname] cmd vovservermgr scheduler resume
```

Special Considerations for Monitor Upgrades

Upgrade the Monitor Software Version

This section provides instructions for changing the software version on which Monitor runs.

The upgrade process includes downloading software archive files from the Altair website, which are then unpacked or untarred into a temporary installation directory. The installer will ask for a destination directory.

When choosing the destination for the downloaded software, be careful to select a non-destructive installation path. This ensures the current Accelerator software is not overwritten or otherwise damaged before the new software is installed and activated. Usually the new version will be installed in a directory as a sibling of the current version, such as `/tools/aap/2023.1.0` for the current version, and `/tools/aap/2023.1.1` for the new version.

Database Upgrades

Most Monitor installations configure a PostgreSQL database to permit historical data reporting about license usage and trends. The upgrade of a Monitor project to a new Monitor software version may require a concurrent upgrade of the PostgreSQL database to a newer format.

Overview

If a Monitor upgrade is required, determine whether a database upgrade is required, and which method is best suited for your needs.

- If no database upgrade is needed, choose the Direct Upgrade method and follow steps 1 – 8.
- If a database upgrade is needed, read the following section to decide which Monitor Upgrade is best for your situation. The subsequent sections describe the details of upgrading Monitor with the "Direct (Offline) Database Upgrade Method" or "Online Database Upgrade" methods.

Methods to Upgrade the PostgreSQL Database

Accelerator queues and Monitor projects can enable a PostgreSQL database during operation. Monitor stores historical license checkouts in a database to support usage report generation from the web UI. Accelerator stores some minimal long-term job historical data in an optional database. When an upgrade to a new Accelerator or Monitor version is planned, version 9.6 and prior PostgreSQL databases require upgrades of pre-existing database files to a new format.

△ **ALTAIR**

There are two database upgrade methods, with the following characteristics and advantages/disadvantages. Choose the method that best fits your situation.

Direct (Offline) Database Upgrade Method

- This method has fewer steps.
- The database is converted to version 14.4 after the software upgrade has been done, while the Monitor project is up and running.
- While the database upgrade (conversion) is taking place, historical usage reporting via the Monitor web UI will be offline. This period of down time could be several hours or days, depending on the size of the database. (Depending on the speed of the filesystem, a 1 TB database upgrade may take a full day.)
- Free disk space 1.5x the size of the database is required so that a new copy of the database can be made.

Online Database Upgrade Method

- The live database and the Monitor project continue to operate while the database upgrade process operates on a copy of the database. Thus, there is no long down time during which history usage report generation cannot be performed.
- The database is converted to version 14.4 before the software upgrade is done.
- The database conversion is performed on a copy of the live database.
- The Monitor software upgrade is performed after the new version 14.4 database is created.
- Once the new database goes online, Monitor will bring the historical content up to date quickly.
- Free disk space 2.5x the size of the database is required to make two copies of the database during the upgrade.

Upgrade Monitor using the Direct Database Method on Linux

Administrators are best served by this upgrade method if a few hours or less of license usage reporting downtime is acceptable, or if you are not currently running a database in your Monitor project.

To upgrade Monitor with this method, the steps are as follows:

1. Source the `vovrc.sh` (or `vovinit.bat`) file for the old software version and set LM_NAME to the name of the Monitor project:

    ```
    % LM_NAME=monitor_project_name
    % source /opt/OLDVERSION/etc/vovrc.sh
    ```

2. View the Monitor web UI and check the **Admin** > **System** > **Database UI** page to see if the latest version 14.4 PostgreSQL database is running, or if no database is configured. In either of these cases, no database upgrade is needed.

3. Set OLDVOVDIR to the value of $VOVDIR for later use in the PostgreSQL database upgrade steps that will be needed later.

    ```
    % OLDVOVDIR=$VOVDIR
    ```

4. Stop the Monitor project.

    ```
    % lmmgr stop -name $LM_NAME -force
    ```

5. Source the `vovrc.sh` (or `vovinit.bat`) file to access the newly installed software.

    ```
    % source /opt/NEWVERSION/etc/vovrc.sh
    ```

6. Start the Monitor project.

```
% lmmgr start -name $LM_NAME
```

7. Enable the shell for a Monitor project and verify that the database has not started, because of the version change:

```
% vovproject enable $LM_NAME
% vovdb_util showcfg
```

8. If a database upgrade is not needed, and the database is running, then **you are done.** Otherwise continue with step 9 .

9. Prepare for a database file format upgrade. The following steps will create an upgraded copy of the Monitor database which will be stored in the same directory as the old one. The new database file directory created will have a new versioned directory name like "dbdataNN_M" where "NN_M" is the PostgreSQL software version. Before doing the conversion, check that the filesystem has enough space for another directory of about the same size. The old database directory will be left in place.

10. Before the database file conversion command can be issued, assign shell variable OLDPG to the `bin` directory containing PostgreSQL software in the old Monitor version installation. This uses the OLDVOVDIR variable set in Step 3.

    Here is a Linux shell command that would accomplish this:

```
% OLDPG=$(ls -d $OLDVOVDIR/postgresql*/bin)
```

11. Set the shell variable DBPATH to the path to the `dbdata9_6` top-level DB directory. See the "Data Path" shown by `vovdb_util showcfg`:

```
% vovdb_util showcfg | grep 'Data Path'
vovdb_util 11/09/2022 18:39:05:    Data Path:   /DBDIR
```

12. Use this example to set DBPATH to `/DBDIR/dbdata9_6` or `DBDIR/dbdata9_4` as appropriate.

13. You are ready to initiate the database file conversion. This command should be run from a Linux shell or Windows command prompt window that can remain running for the hours needed to do the full conversion:

```
% vovdb_util upgrade -sdb $DBPATH -spgsw $OLDPG
```

14. After the database file upgrade completes, the PostgreSQL database may be started by:

```
% vovdb_util startdb
```

15. Ensure that all checkouts that occurred while the database was offline get loaded into the database:

```
% lmmgr loaddb
```

You have now completed the Monitor upgrade. If for some reason you need to downgrade Monitor back to the old version with an old version of PostgreSQL as well, the original database directory remains available for use (see Step 2 text above).

Upgrade Monitor using the Direct Database Method on Windows

Administrators are best served by this upgrade method if a few hours or less of license usage reporting downtime is acceptable, or if you are not currently running a database in your Monitor project.

Before starting, check if a license update if needed. Consider the following:

- If using a keyfile license, no license upgrade is needed.

- If using an RLM license, switch to an ALM license model

> 📝 **Note:** If setting the environment variable is confusing, just capture the folder path and location and make a note of it.

> ℹ️ **Tip:** Reference the Database Upgrade Process for Linux if you need further detailed explanations.

1. Run a database backup from the web portal by clicking **Admin** > **System** > **Database Information**.

2. Set the project name environment variable:

```
% set LM_NAME=<Monitor_project_name>
```

   For example:

```
% set LM_NAME=licmon3
```

3. Set the project environment:

```
% $OLDVOVDIR\bat\vovinit.bat
```

   For example:

```
C:\altair\lm_2210\win64\bat\vovinit.bat
```

4. Log on to the Monitor web portal and check the DB engine version by clicking **Admin** > **System** > **Database Information**. Scroll to the bottom of the page and check the **Engine Version**. If Engine Version is 14.4, you do not need to upgrade.

5. Verify and take note of the current VOVDIR. For example:

```
set | findstr VOVDIR
VOVDIR=C:/altair/lm_2210/win64
```

6. Run the following command:

```
% set OLDVOVDIR=C:/altair/lm_2210/win64
```

7. Stop the Monitor project:

```
% lmmgr stop -name %LM_NAME% -force
```

   For example:

```
lmmgr stop -name licmon3 -force
```

8. Source the vovinit.bat file to access the newly installed software:

```
% $NEWVOVDIR\bat\vovinit.bat
```

   For example:

```
C:\altair\lm_2311_2\win64\bat\vovinit.bat
vovinit: VOV has been initialized
        VOVDIR=C:/altair/lm_2311_2/win64
```

```
vovinit: Checking for C:/altair/lm_2311_2/win64/local/vovinit.bat
```

9. Start the Monitor project:

```
% lmmgr start -name $LM_NAME
```

For example:

```
lmmgr start -name licmon3
```

10. Enable the shell for a Monitor project and verify that the database has not started, because of the version change:

```
% vovproject enable $LM_NAME
```

For example:

```
vovproject enable licmon3
vovproject 08/15/2023 16:41:11: message: Enabling project 'licmon3'...
```

11. Run the following command:

```
vovdb_util showcfg
```

For example:

```
vovdb_util showcfg
vovdb_util 08/15/2023 16:41:33: message: The VOV database configuration is:
vovdb_util 08/15/2023 16:41:33: Host: localhost
vovdb_util 08/15/2023 16:41:33: Port: 14991
vovdb_util 08/15/2023 16:41:33: Data Path: C:/altair/db/lm_2210_33
vovdb_util 08/15/2023 16:41:33: Status: error
vovdb_util 08/15/2023 16:41:33: AutoBackup: enable 0 period 604800 root {}
 startFrom 1 startTo 4 keep 3
```

If a database upgrade is not needed, and the database is running, you are done. Otherwise, continue with the procedure.

12. Prepare for a database file format upgrade. The following steps will create an upgraded copy of the Monitor database which will be stored in the same directory as the old one. The new database file directory created will have a new versioned directory name like dbdataNN_M where NN_M is the PostgreSQL software version. Before doing the conversion, check that the filesystem has enough space for another directory of about the same size. The old database directory will be left in place.

13. Capture the old Postgres bin path:

```
% set OLDPG=C:\altair\lm_2210\win64\postgresql9_6\bin
```

14. Capture the old data path folder:

```
vovdb_util showcfg | findstr 'Data Path'
```

For example:

```
vovdb_util showcfg | findstr "Data Path"
vovdb_util 08/15/2023 16:45:46: Data Path: C:/altair/db/lm_2210_33
```

15. Use this example to set DBPATH to `/DBDIR/dbdata9_6` or `DBDIR/dbdata9_4` as appropriate.

```
% vovdb_util upgrade -sdb %DBPATH% -spgsw %OLDPG%
```

For example:

```
vovdb_util upgrade -sdb C:\altair\db\lm_2210_33\dbdata9_6 -spgsw C:\altair
\lm_2210\win64\postgresql9_6\bin
vovdb_util 08/15/2023 16:55:29: message: sourceVer: 9_6
vovdb_util 08/15/2023 16:55:29: message: targetVer: 14_4
vovdb_util 08/15/2023 16:55:29: message: Preparing to upgrade VOV database to
 version 14.4:
vovdb_util 08/15/2023 16:55:29: Source Dir: C:\altair\db\lm_2210_33\dbdata9_6
vovdb_util 08/15/2023 16:55:29: Version: 9.6
vovdb_util 08/15/2023 16:55:29: Target Dir: C:/altair/db/lm_2210_33/dbdata14_4
vovdb_util 08/15/2023 16:55:29: Version: 14.4
vovdb_util 08/15/2023 16:55:29:
vovdb_util 08/15/2023 16:55:29:
The upgrade process creates a copy of the database for use with the new version
 of PostgreSQL.
There must be free disk space greater than the size of the current database.
The original files will remain after the upgrade, but may be deleted at any time
 to reclaim disk space.
vovdb_util 08/15/2023 16:55:29:
Upgrade database (yes/no)? yes
vovdb_util 08/15/2023 16:55:49: message: Proceeding with upgrade.
vovdb_util 08/15/2023 16:55:49: message: Configuration of upgrade
vovdb_util 08/15/2023 16:55:49: message: ::PGBINOLD (Source Bin): C:\altair
\lm_2210\win64\postgresql9_6\bin
vovdb_util 08/15/2023 16:55:49: message: ::PGBINNEW (Destination Bin): C:/altair/
lm_2311_2/win64/postgresql14_4/bin
vovdb_util 08/15/2023 16:55:49: message: ::PGDATAOLD (Source Data): C:\altair\db
\lm_2210_33\dbdata9_6
vovdb_util 08/15/2023 16:55:49: message: ::PGDATANEW (Destination Data): C:/
altair/db/lm_2210_33/dbdata14_4
vovdb_util 08/15/2023 16:55:49: message: Checking DB state at location: C:\altair
\db\lm_2210_33\dbdata9_6
vovdb_util 08/15/2023 16:55:49: message: Creating new database instance.
The files belonging to this database system will be owned by user "jlin".
This user must also own the server process.

The database cluster will be initialized with locale "en-US".
The default text search configuration will be set to "english".

Data page checksums are disabled. ...
```

16. After the database file upgrade completes, the PostgresSQL database may be started by:

```
% vovdb_util startdb
```

For example:

```
vovdb_util startdb
vovdb_util 08/15/2023 16:57:51: message: Starting database...
vovdb_util 08/15/2023 16:57:51: message: Checking user privileges
vovdb_util 08/15/2023 16:57:51: message: User is not admin or service.
vovdb_util 08/15/2023 16:57:51: message: Starting database
...
```

17. Ensure that all checkouts that occurred while the database was offline get loaded into the database:

```
% lmmgr loaddb -v
```

For example:

```
lmmgr loaddb -v
lmmgr 08/15/2023 16:59:59: message: Loading all data files into the database...
lmmgr 08/15/2023 16:59:59: message: Load all checkouts that come from sampling
...
```

Upgrade Monitor Using the Online Database Method on Linux

This upgrade method will allow license usage reporting to be available from a live database while conversion of a backup database executes in parallel.

1. Set LM_NAME to the Monitor project name and enable the shell.

```
% export LM_NAME=lm_project
% vovproject enable $LM_NAME
% SWD=$(vovserverdir -p .)
```

2. Set more environment variables that will be used later. OLDVOVDIR is the value of VOVDIR from the current software version, and NEWVOVDIR is the value of VOVDIR from the new software version. For example:

```
% OLDVOVDIR=/path/to/vov/install/2022.1.0_ga/linux64
% NEWVOVDIR=/path/to/vov/install/2023.1.0_ga/linux64
```

3. Record the location of the database for later removal. Store the path in ORIGPATH.

```
% vovdb_util showcfg
% ORIGPATH=<the path shown>
```

4. Create PG 9.X database backup. Choose the location and set the BACKUP variable. This step runs at disk copy speed.

```
% BACKUP=/big/DBBACKUP
% vovdb_util backup $BACKUP
```

5. Export the PG 9.X database passwords using a version of the Accelerator software (2023.1.0 or higher) that supports the `exportpasswords` subcommand.

```
% source $NEWVOVDIR/etc/vovrc.sh
% vovdb_util exportpasswords $SWD/PASSWORD_FILE
```

6. Create a new Monitor converter project that will be used to convert the older PG database to a version 14.4 database. Configure the backed-up database for use in this new project.

```
% export LM_CONVERTER=lm_temp_project
% source $NEWVOVDIR/etc/vovrc.sh
% lmmgr start -name $LM_CONVERTER -port any:1 -webport any:2 \
-roport 0 -inport any:3 -upport any:4
% vovproject enable $LM_CONVERTER
% vovdb_util configure -noconfirm `hostname -s` $BACKUP
```

> 📝 **Note:** Steps 7 - 9 will target the Monitor $LM_CONVERTER project.

7. Find the `bin` directory containing the PG 9.X software.

```
% OLDPG=$(ls -d $OLDVOVDIR/postgresql*/bin)
```

8. Convert database PG 9.X to PG 14.4. This will operate at about 1/2 of disk-copy speed.

```
% vovdb_util importpasswords $SWD/PASSWORD_FILE
% vovdb_util upgrade -sdb $BACKUP/dbdata9_6 -spgsw $OLDPG
```

9. After the database is converted, shut down the Monitor converter project.

```
% lmmgr stop -name $LM_CONVERTER -force
% sleep 5 ; vovproject destroy -force $LM_CONVERTER
```

> 📝 **Note:** Steps 10 and beyond will target the production Monitor project - $LM_NAME.

10. Shut down the Monitor project and disable DB so it does not come up initally after restart.

```
% source $OLDVOVDIR/etc/vovrc.sh
% vovproject enable $LM_NAME
% vovdb_util stopdb
% lmmgr stop -name $LM_NAME -force
```

11. Upgrade the production Monitor project with start and new Monitor software version.

```
% sleep 5
% source $NEWVOVDIR/etc/vovrc.sh
% lmmgr start -name $LM_NAME
% vovproject enable $LM_NAME
```

12. Configure the location of the PG 14.4 database, then start the database.

```
% vovdb_util configure -reset -noconfirm `hostname -s` $BACKUP
% sleep 3 ; vovdb_util startdb
```

13. Verify the status of the PG 14.4 live database by viewing the web UI page **Admin** > **System** > **Database**.

14. Load checkouts that occurred since the time of the backup.

```
% lmmgr loaddb
```

15. Remove the two extra DB copies:

    a) Remove the 9.x backup DB from `/big/DBBACKUP/pgdata9*`

    b) Remove the 9.x project DB from $ORIGPATH

Upgrade Monitor Using the Online Database Method on Windows

Before starting, check if a license update if needed. Consider the following:

- If using a keyfile license, no license upgrade is needed.

△ **ALTAIR**

- If using an RLM license, switch to an ALM license model

> 📝 **Note:** If setting the environment variable is confusing, just capture the folder path and location and make a note of it.

An overview of this process:

1. Backup the current database.
2. Upgrade the backup database.
3. Switch the backup database as the production database.

Be sure to provide a meaningful folder name where the backup database resides. For example:

```
set BACKUP=/big/DBBACKUP
```

For example, `/big/DB_14`

1. Set LM_NAME to the Monitor project name and enable the shell.

   ```
   % set LM_NAME=licmon
   % vovproject enable %LM_NAME%
   % vovserverdir -p
   % set SWD=%<_the_return_value_of_above_command%
   ```

   For example:

   ```
   % set SWD=C:\altair\swd\licmon.swd
   ```

2. Set more environment variables that will be used later. OLDVOVDIR is the value of VOVDIR from the current software version, and NEWVOVDIR is the value of VOVDIR from the new software version. For example:

   ```
   % set OLDVOVDIR=C:\altair\lm_2210\win64
   % set NEWVOVDIR=C:\altair\lm_2311\win64   (upgrade to v2023.1.1)
   ```

3. Record the location of the database for later removal.

   ```
   vovdb_util showcfg
   ```

   For example:

   ```
   vovdb_util 08/14/2023 15:27:15: message: The VOV database configuration is:
   vovdb_util 08/14/2023 15:27:15:   Host:       localhost
   vovdb_util 08/14/2023 15:27:15:   Port:       37204
   vovdb_util 08/14/2023 15:27:15:   Data Path:  C:/altair/db/lm_2210_3
    <<<<<<<<<<<<< used this path in next step
   vovdb_util 08/14/2023 15:27:15:   Status:     ready
   vovdb_util 08/14/2023 15:27:15:   AutoBackup: enable 0 period 604800 root {}
    startFrom 1 startTo 4 keep 3
   ```

4. Store the path in ORIGPATH.

   ```
   % set ORIGPATH=C:/altair/db/lm_2210_3
   ```

△ **ALTAIR**

5.   Create PG 9.X database backup.

```
% set BACKUP=/big/DBBACKUP
```

For example:

```
% set BACKUP=C:\altair\DBBACKUP
```

6.   Choose the location and set the BACKUP variable. This step runs at disk copy speed. (Be sure to use a meaningful folder name.)

```
% vovdb_util backup %BACKUP%
```

For example:

```
vovdb_util backup C:\altair\DBBACKUP
vovdb_util 08/14/2023 15:32:58: message: Backing up the database to 'C:\altair
\DBBACKUP'...
pg_basebackup: initiating base backup, waiting for checkpoint to complete
pg_basebackup: checkpoint completed
transaction log start point: 0/3000028 on timeline 1
pg_basebackup: starting background WAL receiver
39861/39861 kB (100%), 1/1 tablespace
transaction log end point: 0/3007148
pg_basebackup: waiting for background process to finish streaming ...
pg_basebackup: base backup completed
vovdb_util 08/14/2023 15:33:06: message: Database backed up.
```

7.   Source the vovinit.bat file to access the newly installed software:

```
% $NEWVOVDIR\bat\vovinit.bat
```

For example:

```
C:\altair\lm_2311\win64\bat\vovinit.bat
```

8.   Verify the environment version:

```
% vovversion
```

For example:

```
%  C:\Users\jlin>vovversion
2023.1.1
```

9.   Export the PG 9.X database passwords using a version of the Accelerator software (2023.1.0 or higher) that supports the `exportpasswords` subcommand.

```
vovdb_util exportpasswords $SWD\PASSWORD_FILE
```

For example:

```
% vovdb_util exportpasswords C:\altair\licmon\licmon2.swd\PASSWORD_FILE
vovdb_util 08/14/2023 16:18:48: message: Database passwords exported to: C:
\altair\licmon\licmon2.swd\PASSWORD_FILE
```

10. Create a new Monitor converter project that will be used to convert the older PG database to a version 14.4 database.

```
% set LM_CONVERTER=lm_temp_project (may skip)

% $NEWVOVDIR\bat\vovinit.bat
```

For example:

```
C:\altair\lm_2311\win64\bat\vovinit.bat
```

Steps 11 - 17 will target the Monitor $LM_CONVERTER project.

11. Configure the backed-up database for use in this new project.

```
% lmmgr start -name lm_temp_project -port any:1 -webport any:2 -roport 0 -inport
  any:3 -upport any:4
```

For example:

```
lmmgr start -name lm_temp_project -port any:1 -webport any:2 -roport 0 -inport
  any:3 -upport any:4
lmmgr 08/14/2023 16:40:57: message: Starting Monitor instance lm_temp_project
...
```

12. Run the following commands:

```
% vovproject enable lm_temp_project

% hostname
% vovdb_util configure -noconfirm `hostname -s` %BACKUP%
```

For example:

```
% vovdb_util configure -noconfirm sdc-win19csg-1 C:\altair\DBBACKUP
```

13. Check the path, which is the `postgres bin\` folder of the current release. For example:

```
C:\altair\lm_2210\win64\postgresql9_6\bin
```

14. Convert database PG 9.X to PG 14.4. This will operate at about 1/2 of disk-copy speed.

```
% vovdb_util importpasswords $SWD/PASSWORD_FILE
```

For example:

```
vovdb_util importpasswords C:\altair\licmon\licmon2.swd\PASSWORD_FILE
```

15. Run the following commands:

```
vovdb_util upgrade -sdb "C:\altair\DBBACKUP\dbdata9_6" -spgsw "C:\altair
\lm_2210\win64\postgresql9_6\bin"
```

For example:

```
vovdb_util upgrade -sdb C:\altair\DBBACKUP\dbdata9_6 -spgsw C:\altair
\lm_2210\win64\postgresql9_6\bin
vovdb_util 08/15/2023 10:38:19: message:       sourceVer: 9_6
vovdb_util 08/15/2023 10:38:19: message:       targetVer: 14_4
```

```
vovdb_util 08/15/2023 10:38:19: message: Preparing to upgrade VOV database to
 version 14.4:
vovdb_util 08/15/2023 10:38:19:                Source Dir: C:\altair\DBBACKUP
\dbdata9_6
vovdb_util 08/15/2023 10:38:19:                   Version: 9.6
vovdb_util 08/15/2023 10:38:19:                Target Dir: C:/altair/DBBACKUP/
dbdata14_4
vovdb_util 08/15/2023 10:38:19:                   Version: 14.4
vovdb_util 08/15/2023 10:38:19:
vovdb_util 08/15/2023 10:38:19: The upgrade process creates a copy of the
 database for use with the new
version of PostgreSQL.  There must be free disk space greater than the size of
the current database.  The original files will remain after the upgrade, but
may be deleted at any time to reclaim disk space.
vovdb_util 08/15/2023 10:38:19:
Upgrade database (yes/no)? yes
vovdb_util 08/15/2023 10:38:35: message: Proceeding with upgrade.
vovdb_util 08/15/2023 10:38:35: message: Configuration of upgrade
...
```

16. After the database is converted, shut down the Monitor converter project:

```
% lmmgr stop -name $LM_CONVERTER -force
```

For example:

```
lmmgr stop -name lm_temp_project
```

17. Wait for at least 5 seconds before shutting down the database. Then run:

```
% vovproject destroy -force $LM_CONVERTER
```

For example:

```
vovproject destroy -force lm_temp_project
vovproject 08/15/2023 10:55:22: message: Deleting c:/altair/licmon/
lm_temp_project.swd/
vovproject 08/15/2023 10:55:22: message: Deleting registry entry C:/altair/
lm_2311/win64/local/registry/jlin/lm_temp_project@sdc-win19csg-1
vovproject 08/15/2023 10:55:22: message: Deleting C:/altair/lm_2311/win64/local/
registry/system-lm/lm_temp_project.reg
```

The remainder of the procedure will target the production Monitor project - $LM_NAME.

18. Shut down the Monitor project and disable the database so it does not come up initially after restart:

   a) Run:

```
% $OLDVOVDIR\bat\vovinit.bat
```

   For example:

```
C:\altair\lm_2210\win64\bat\vovinit.bat
```

   b) Run:

```
% vovproject enable $LM_NAME
```

△ ALTAIR

For example:

```
vovproject enable licmon2
```

c) Run:

```
% vovdb_util stopdb
```

For example:

```
vovdb_util stopdb
vovdb_util 08/15/2023 11:15:09: message: Stopping database...
vovdb_util 08/15/2023 11:15:10: message: Database stopped.
```

d) Run:

```
% lmmgr stop -name $LM_NAME -force
```

For example:

```
lmmgr stop -name licmon2 -force
lmmgr 08/15/2023 11:16:33: message: Stopping LicenseMonitor project
 licmon2...
vovproject 08/15/2023 11:16:34: message: Checking privilege to stop project
 'licmon2'
vovproject 08/15/2023 11:16:34: message: Stopping all retracing
vovstop 08/15/2023 11:16:34: msg-1: No retrace to stop
vovproject 08/15/2023 11:16:34: message: Retracing stopped
vovproject 08/15/2023 11:16:35: message: Stopping all daemons in licmon2
vovresourced               STOPPING ...
vovdbd                     STOPPING ...
...
```

19. Upgrade the production Monitor project with start and new Monitor software version by running the following commands:

a) Wait for at least 5 seconds, and then initialize the new directory:

```
% $NEWVOVDIR\bat\vovinit.bat
```

For example:

```
C:\altair\lm_2311\win64\bat\vovinit.bat
```

b) Verify the version:

```
% vovversion
```

For example:

```
vovversion
2023.1.1
```

c) Start a Monitor instance:

```
% lmmgr start -name $LM_NAME
```

For example:

```
lmmgr start -name licmon2
lmmgr 08/15/2023 11:19:45: message: Starting Monitor instance licmon2
lmmgr 08/15/2023 11:19:45: message: Updating autostart/
start_vovnotifyd.tcl...
lmmgr 08/15/2023 11:19:45: message: Updating autostart/start_vovlmd.tcl...
...
```

d) Enable the Monitor project:

```
% vovproject enable $LM_NAME
```

For example:

```
vovproject enable licmon2
vovproject 08/15/2023 11:20:55: message: Enabling project 'licmon2'...
```

20. Configure the location of the PG 14.4 database, then start the database.

```
vovdb_util configure -reset -noconfirm `hostname -s` $BACKUP
```

For example:

```
vovdb_util configure -reset -noconfirm sdc-win19csg-1 C:\altair\DBBACKUP
vovdb_util 08/15/2023 11:24:02: message: Configuration saved.
```

21. Wait for a few seconds, then restart the database:

```
% vovdb_util startdb
```

For example:

```
vovdb_util startdb
vovdb_util 08/15/2023 11:25:03: message: Starting database...
vovdb_util 08/15/2023 11:25:04: message: Checking user privileges
vovdb_util 08/15/2023 11:25:04: message: User is not admin or service.
vovdb_util 08/15/2023 11:25:04: message: Starting database
vovdb_util 08/15/2023 11:25:04: message: LOG:  starting PostgreSQL 14.4, compiled
 by Visual C++ build 1914, 64-bit
vovdb_util 08/15/2023 11:25:04: message: LOG:  listening on IPv6 address "::",
 port 37204
...
```

22. In the Monitor web portal, click **Admin** > **System** > **Database Information**. Scroll to the bottom of the page and verify the Engine Version displays 14.4.

23. Load checkouts that occurred since the time of the backup.

```
% lmmgr loaddb -v
```

For example:

```
lmmgr loaddb -v
lmmgr 08/15/2023 11:30:11: message: Loading all data files into the database...
lmmgr 08/15/2023 11:30:12: message: Load all checkouts that come from sampling
...
```

24. Remove the two extra database copies; for example backup `db /big/DBBACKUP/pgdata9_6` and the original `C:/altair/db/lm_2210_3/datadb9_6 project db`.

Windows Installation Overview

VOV supports Windows and many UNIX variants. This document goes over issues that you may encounter in setting up a VOV environment in Windows machines.

> 📝 **Note:** PowerShell is not supported. When installing or upgrading Altair Accelerator products in a Windows environment, do not use PowerShell.

Windows is Single-User (mostly)

Unlike UNIX, the Windows operating system is oriented toward single-user computers. Windows is multi-tasking: the CPU is shared among many processes, each of which may have one or more threads of execution. When you are logged in to a Windows machine, your user ID owns the resources of that machine. In UNIX-like operating systems, multiple users may be logged in to a machine concurrently, and each user may use whatever resources according to the permissions (rwx) assigned to them. Processes that run as Windows services run in the local SYSTEM account by default, but may also run using a domain account, which can be different from the one logged on to the machine.

Windows NT has a significantly different security model from UNIX systems. The Windows security for domains is based on a centralized user authentication model, where each user's credentials are verified by the primary domain controller when that user logs in. Each machine may also have local machine accounts, but those accounts have no access to network resources. Windows NT object access is based on access-control lists (ACLs), rather than permission bits as in UNIX. The ACL model is more powerful and flexible than the UNIX model.

Windows Networking

Windows networking is very different and, by default, limited compared to UNIX-like operating systems. By design, Windows ships without any remote shell facilities, and remote execution is limited to client-server using remote procedure calls. These limitations have several significant consequences for VOV, which needs a way to remotely start `vovtasker` processes on the vovtasker machines. Once the `vovtasker` process is running, it communicates with the server directly via TCP/IP.

The Windows network filesystem model also differs significantly from UNIX-like systems. Most UNIX-like systems use Network File System (NFS), originally developed by Sun Microsystems, and now a public-domain standard. Windows uses the SMB (Server Message Block) protocol, which is also now being called CIFS (Common Internet File System) by Microsoft.

Organize Data into VOV Projects

A basic concern is how to organize your data into VOV projects. You will need to do this whether or not you have any Windows machines. This is a trade-off between the number of projects and servers you will need to maintain, and the complications of setting up operating system permissions to allow all users the read/write access needed to do their work.

Some files are 'primary inputs' to a project, those which appear at the top of the flow graph and are not generated by any job. Examples of such files are source and header files for software projects, or Verilog HDL source files for chip design projects. The files at the bottom of the flow graph, which are not used by any job in the project, are the primary outputs of the project. An example of this might be the executable file built by a VOV software project. These primary outputs could then be primary inputs of downstream projects.

The same principles of modularization and information hiding which apply to software design are also applicable to project and flow design. Judicious choice of project boundaries allows you to hide unnecessary detail and enforce correct information propagation between projects.

Choose VOV Project Server Machine

Each VOV project is managed by a server process, which maintains the project database and interacts with the client processes, such as VOV tasker machines, vovconsole, web browsers, etc.

In a mixed UNIX/Windows environment, you could run project servers either on UNIX or Windows. Running the server on UNIX, and using Windows machines only as taskers is usually a better choice, because of the relative ease of remote administration of UNIX machines. It is, however, possible to create a pure Windows VOV installation.

Choose Windows Tasker Machine Mount Points

The filesystems containing the VOV Windows executables and the project data directories must be mounted on all Windows machines that serve as VOV taskers.

You will need some capability for cross-platform mounting filesystems. A common solution is to run the public-domain Samba software on a UNIX file server, which enables setting up SMB shares of UNIX filesystems. These may be mounted via the 'net use' command from Windows. It is also possible to use NFS client software on Windows machines, but experience has shown this to be an inferior solution, for cost and administrative overhead reasons. To our knowledge, there is not a good public-domain NFS client implementation for Windows.

Another approach is to place the user and project data on a purpose-built fileserver machine which supports both NFS and CIFS, such as a NetApp filer. These machines have a custom-designed operating system which supports serving the same file data using both NFS, CIFS and HTTP. This solution offers high performance and low administrative overhead, but is also costly.

Site-specific vovinit.bat

This application note provides several small example batch scripts that make it easy to mount the filesystems and set up the environment for VOV taskers. You need to decide where your project's is are to be stored, and adapt the `.bat` files to match your mount point setup. In many cases, you can do it all in one script by customizing your local `vovinit.bat` file.

The command to initialize a Windows cmd shell for use with VOV is called `vovinit`. This batch script adds the VOV bin and bat directories to the path, and calls the script `$VOVDIR/local/vovinit.bat` if it exists. The local `vovinit.bat` is a good place to put site-wide initialization and variable setup. The following example shows some typical site-specific customizations.

Example script: vovinit.bat

```
REM mount tools area containing flowtracer WinNT .exe files
if not exist V:\NUL net use V: \\appsrv\tools

set VNCSWD=r:/release/VOV/vnc

REM user Windows home directories
if not exist U:\NUL net use U: \\file01\users

REM -- Optional: update the local disk cache of useful executables.
REM vtclsh %VOVDIR%/tcl/vtcl/vovupdatecache.tcl

REM Synchronize with the timeserver
REM Do this last because it fails if not holding admin rights
net time \\timeserver /set /y
```

> 📝 **Note:** This script first tries to mount the VOV software and user directories. The commented-out `vovupdatecache.tcl` command updates a local cache directory of executables. This may be used to reduce the load on the fileserver and network, at the cost of local disk space. At the end, the script synchronizes the local system clock with the time server.

Windows Tasker Accessibility Policy and Practices

Many of the Windows machines at most companies are users' desktop machines. If these are to serve as vovtaskers, you need to develop policies and practices about times of availability to avoid the degradation of interactive response time.

If your project needs Windows tasker machines, you have two main choices:

- VOV supports time-dependent resources, which is one approach to this issue. The resources of a machine may be static or defined by a Tcl procedure in the namespace VovResources. Some example procedures are found in `$VOVDIR/etc/tasker_scripts/taskerRes.tcl`.

- Another approach is to dedicate machines other than users' desktop machines to act as VOV Windows tasker machines. This is often done where there is a large Windows component to the workload, such as SmartSpice simulations.

Add Resource Specs to Jobs

If you are reading this Application Note, it is likely you have jobs which can *only* run on Windows in your project. You must add a resource specification of 'win64' to those jobs, so that VOV will dispatch the jobs to the correct type of machine.

Resources should be specified in the Flow Description file (`Flow.tcl`) for the project, using the 'R' statement. You may also modify jobs in your project interactively using the Node Editor in the vovconsole. This latter is good for interactive testing, but those customizations will be lost when you rebuild the project unless the resource specifications are in the Flow Definition File.

User IDs that Run Jobs on Windows Tasker Machines

Windows NT does not provide a means to easily switch to the ID of another user. When a user is logged in to a Windows machine, that user owns all the machine's resources. That user must have appropriate permissions in the data directories to access and create or modify files.

The most common solution to this issue is to create a specific UID to own the data for each project, and put that UID and all the users who also need access to the data into a netgroup, giving group write permission.

Start the VOV Tasker Machines

Because Windows does not supply a remote shell capability, VOV provides the `vovtsd` (Vov Tasker Service Daemon) program to enable Windows machines to act as vovtaskers. This program, once started, listens on a TCP/IP port for incoming tasker start requests. When `vovtsd` receives a tasker start request, it starts a tasker process, which then communicates directly with the vovserver to receive jobs and return status information.

Usually the TCP/IP port number is derived from the name of the logged-in user, but it may also be specified explicitly. If you use a non-default port, you must also specify that port in the project's taskers.tcl file. Note the use of -vovtsdport in the example below.

After you have started the `vovtsd` on the Windows tasker machine and configured the taskers.tcl file, you can then start and stop the taskers.tcl process on that machine using the `vovtaskermgr` command, from the console GUI, or the browser.

Periodic Automated Run and Data Reduction

Once you have set up Windows tasker machines, you will probably wish to run regression tests and other projects periodically and post the results to your internal website. This can be handled using the UNIX 'cron' facility or a Windows 'at' job.

The cron job typically wakes up and runs a vovbuild script which adds a new set to the trace of the project, and then retraces the jobs in that set.

VOV has the **vovcrontab** command for UNIX, which sets up crontab entries for hourly, daily, weekly, and monthly jobs. These jobs run scripts which are located in the scripts subdirectory of the project configuration directory.

vovserver Configuration

The following section explains the details of how to configure the vovserver to enable Windows tasker machines.

This example shows some selected parts of the server configuration file `taskers.tcl`, which describes the tasker machine setup. The key difference is that Windows uses drive letters in the names of mount points.

Windows Machines in `taskers.tcl` File

Because Windows paths include a drive letter, it is not possible to refer to a filesystem using the same path on Windows and UNIX. This means that it is necessary to explicitly specify the path to the VOV software and to the server working directory in the `taskers.tcl` file when you set up a project.

```
# -- lists of various kinds of host platforms
set linHosts { lin1 lin2 lin3 }
set macHosts { mac1 mac2 mac3 }
set winHosts { win1 win2 win3 }

# -- Simple loop to declare UNIX taskers with default parameters
foreach host [concat $linHosts $macHosts] {
    vtk_tasker_define $host
}

# set different defaults for the Windows taskers to follow
# FINE POINT: -serverdir specifies the _parent_
#             of the dir containing configuration files
# NOTE: paths specified with forward slashes, used by Tcl
#       do _not_ use backslashes as hierarchy separator here
vtk_tasker_set_default -vovdir    R:/rtda/2016.09/win64  -serverdir  H:/vov  -
vovtsdport 16666

foreach host $winHosts {
    vtk_tasker_define $host
}
```

Starting Tasker Processes for your Project

1. Enable your shell for VOV:

   ```
   vovproject enable <project_name)
   ```

2. Start the tasker processes:

   ```
   vovtaskermgr start <tasker_machine_name>
   ```

   If you supply <tasker_machine_name>, then only that machine will be started. Otherwise, all taskers named in the configuration file which are not yet running are started.

Set Up Altair Monitor Services

This section provides instructions to set up services on Windows with PostgreSQL.

To run PostgreSQL in Windows, you cannot set this up as an administrator. However, to start the Monitor services, you need to be an administrator. To start up Monitor services in an unprivileged account -- which is required to run postgreSQL -- follow these steps:

1. Go to the directory where the `lm-winNt.exe` file was downloaded.

2. Right-click on the `lm-winNt.exe` file and choose **Run as Admin**.
   A window will open.

3. At the bottom of that window, enter the credentials for an Admin level account.

4. Click **submit**.

   If an error message appears stating the user does not have logon privileges, you will need to add those privileges to that account.

5. Go to **Computer Management** > **Services and Applications** > **Services**.

6. Find and open the Monitor service.

7. Go to the **Logon** tab.

8. Add the username and the password. Be sure to add ".\" before the username.

9. Save and exit.

10. Close out of the `lm-winNt.exe` window and start it back up as Admin, and then click **Start** at the bottom of the page.

File System Layout

Directory Hierarchy

The Altair Accelerator product suite follows a non-intrusive model of file installation. All the files installed for the product are within the single branch of the file system where the product is installed. No files are installed into other areas of the file system.

This leads to a convention that you should follow for managing the installed files. Establish a single directory to hold all Altair Accelerator products and only install Altair Accelerator releases into this area.

The intention is that this single Altair Accelerator directory will contain sub-directories for each release version of the products, named after the release. All the files installed for a given release version will be placed into the file system branch below their release directory.

Within the release directory, along with subdirectories holding commonly shared files, there are sub-directories for each platform (machine type) that is used within the network. These subdirectories hold the executables for those platforms. The expectation is that each machine in the network can access the installed files by mounting this area onto their system as a Network File System (NFS) mount. This allows them to use a local path on their system to access their executables in the installation area.

```
Top Level Installation Directory Structure

/remote/tools/
    altair/
        /VOV/
         current/ --> 2023.1.1 <-- symbolic link
         2023.1.0/
             arvmv8/
                 bin/
                    ...
             linux64/
                 ...
         2022.1.1/
                 ...
```

For instance, if the top level product installation directory is `/remote/tools/altair/VOV`, then it is recommended that you install the 2023.1.1 version into a subfolder with the path `/remote/tools/altair/VOV/2023.1.1`.

Later, when you install the 2023.1.2 version, you will install it into the path `/remote/tools/altair/VOV/2023.1.2`.

There are a set of files, such as configuration files, within each release that are expected to be changed locally. These *site specific information* and *customization* files are expected to be found under the release's `common/local` subdirectory, for example: `/remote/tools/altair/VOV/common/local/*`. Since these files are intended to be used with across all versions of the software, the default behavior of the installer is to create a local directory as a sibling to the release directory and use a symbolic in the common directory to route lookups to the release-agnostic location.

```
Top Level Installation Directory Structure (showing release-independent /local
 directory)

    /remote/tools/altair/VOV/
        local/ <-- Holds changed files outside of a release.
            capsules/
            registry/
            ...
        current --> 2023.1.1
```

```
        ...         2022.1.0/
    armv8/
        bin/
        local/ --> ../common/local <-- symbolic link
        ...
    common/
        local/ --> ../../local <-- symbolic link
        ...
    linux64/
        ...
  2022.1.1/
    ...
```

Note that there is also a `/local` directory within each platform area that, by default, is also linked to the `common/local` directory. Although rare, this allows you to have different local directories for each platform by removing the symbolic link and creating an actual local directory at this location.

Similar to the local directory, Accelerator specifically creates a release-agnostic directory named `/vnc` that contains the files associated with Accelerator queues.

Similarly, Monitor creates a release-agnostic directory named `/licmon` that is the default location for the Altair Monitor User GuideMonitor configuration and data files for all instances of Monitor

Use Altair Accelerator Package Help

Altair Accelerator Package documentation is available in HTML and PDF format.

Access the Help when Altair Accelerator Package is Running

When Altair Accelerator Package is running, it displays the documentation through its browser interface. To access it from browser, you need to know which host and port Altair Accelerator Package is running on. Ask your administrator, or find the URL for Altair Accelerator Package with the following command:

```
% Altair Accelerator Package cmd vovbrowser
http://comet:6271/project
```

In the example below, assume Altair Accelerator Package is running on host comet, port 6271. The URL for Altair Accelerator Package is:

```
http://comet:6271
```

To get the entire suite of Altair Accelerator documents, including FlowTracer™, Accelerator™, Monitor™ and the VOV subsystem, use the following URL:

```
http://comet:6271/doc/html/bookshelf/index.htm
```

Access the Help when Altair Accelerator Package is not Running

All the documentation files are in the Altair Accelerator install directory, so you can access them even if vovserver is not running. To do this, open /installation_directory/common/doc/html/bookshelf/index.htm in your browser.

> ℹ️ **Tip:** Bookmark the above URL for future reference.

Access the Help PDF Files

Altair Accelerator also provides PDF files for each of the guides. All the PDF files are in the directory /installation_directory/common/doc/pdf

Access the Help via the Command Line

The main commands of Accelerator are nc and ncmgr, with some subcommands and options. You can get usage help, descriptions and examples of the commands by running the command without any options, or with the -h option. For example,

```
% nc info -h
nc:
nc:  NC INFO:
nc:  Get information about a specific job or list of jobs.
nc:  USAGE:
nc:  % nc info <jobId> [options]...
nc:   -h      -- Show this message
nc:   -l      -- Show the log file
nc:
```

Access the Help via the vovshow Command

Another source of live information is using the command vovshow. The following options are often useful:

vovshow -env RX	Displays the environment variables that match the regular expression RX provided.
vovshow -fields	Shows the fields known to the version of VOV in use.
vovshow -failcodes	Shows the table of known failure codes.

For example, to find a variable that controls the name of the stdout/stderr files, without knowing the exact name of that variable, the following command can be used:

```
% vovshow -env STD
VOV_STDOUT_SPEC          Control the names of file used to save stdout and
                         stderr. The value is computed by substituting
                         the substrings @OUT@ and @UNIQUE@ and @ID@.
                         Examples: % setenv VOV_STDOUT_SPEC
                         .std@OUT@.@UNIQUE@ % setenv VOV_STDOUT_SPEC
                         .std@OUT@.@ID@
```

The output provides a description of all the variables used by the FlowTracer system that include the substring "STD". In this example, the output resultVOV_STDOUT_SPEC.

Legal Notices

Intellectual Property Rights Notice

Copyrights, trademarks, trade secrets, patents and third party software licenses.

Copyright © 1986-2023 Altair Engineering Inc. All Rights Reserved.

This Intellectual Property Rights Notice is exemplary, and therefore not exhaustive, of intellectual property rights held by Altair Engineering Inc. or its affiliates. Software, other products, and materials of Altair Engineering Inc. or its affiliates are protected under laws of the United States and laws of other jurisdictions. In addition to intellectual property rights indicated herein, such software, other products, and materials of Altair Engineering Inc. or its affiliates may be further protected by patents, additional copyrights, additional trademarks, trade secrets, and additional other intellectual property rights. For avoidance of doubt, copyright notice does not imply publication. Copyrights in the below are held by Altair Engineering Inc. or its affiliates. Additionally, all non-Altair marks are the property of their respective owners.

This Intellectual Property Rights Notice does not give you any right to any product, such as software, or underlying intellectual property rights of Altair Engineering Inc. or its affiliates. Usage, for example, of software of Altair Engineering Inc. or its affiliates is governed by and dependent on a valid license agreement.

Altair Simulation Products

Altair® AcuSolve® ©1997-2023

Altair Activate®©1989-2023

Altair® Battery Designer™ ©2019-2023

Altair Compose®©2007-2023

Altair® ConnectMe™ ©2014-2023

Altair® EDEM™ © 2005-2023

Altair® ElectroFlo™ ©1992-2023

Altair Embed® ©1989-2023

Altair Embed® SE ©1989-2023

Altair Embed®/Digital Power Designer ©2012-2023

Altair Embed® Viewer ©1996-2023

Altair® ESAComp® ©1992-2023

Altair® Feko® ©1999-2023

Altair® Flow Simulator™ ©2016-2023

Altair® Flux® ©1983-2023

Altair® FluxMotor® ©2017-2023

Altair® HyperCrash® ©2001-2023

Altair® HyperGraph® ©1995-2023

Altair® HyperLife® ©1990-2023

Altair® HyperMesh® ©1990-2023

Altair® HyperSpice™ ©2017-2023

Altair® HyperStudy® ©1999-2023

Altair® HyperView® ©1999-2023

Altair® HyperViewPlayer® © 2022-2023

Altair® HyperWorks® ©1990-2023

Altair® HyperXtrude® ©1999-2023

Altair® Inspire™ ©2009-2023

Altair® Inspire™ Cast ©2011-2023

Altair® Inspire™ Extrude Metal ©1996-2023

Altair® Inspire™ Extrude Polymer ©1996-2023

Altair® Inspire™ Form ©1998-2023

Altair® Inspire™ Mold ©2009-2023

Altair® Inspire™ PolyFoam ©2009-2023

Altair® Inspire™ Print3D ©2021-2023

Altair® Inspire™ Render©1993-2023

Altair® Inspire™ Studio ©1993-2023

Altair® Material Data Center™ ©2019-2023

Altair® MotionSolve® ©2002-2023

Altair® MotionView® ©1993-2023

Altair® Multiscale Designer® ©2011-2023

Altair® nanoFluidX® ©2013-2023

Altair® OptiStruct® ©1996-2023

Altair® PollEx™ ©2003-2023

Altair® PSIM™ © 2022-2023

Altair® Pulse™ ©2020-2023

Altair® Radioss® ©1986-2023

Altair® romAI™ © 2022-2023

Altair® S-FRAME® © 1995-2023

Altair® S-STEEL™ © 1995-2023

Altair® S-PAD™ © 1995-2023

Altair® S-CONCRETE™ © 1995-2023

Altair® S-LINE™ © 1995-2023

Altair® S-TIMBER™ © 1995-2023

Altair® S-FOUNDATION™ © 1995-2023

Altair® S-CALC™ © 1995-2023

Altair® S-VIEW™ © 1995-2023

Altair® Structural Office™ © 2022-2023

Altair® SEAM® © 1985-2023

Altair® SimLab® © 2004-2023

Altair® SimLab® ST © 2019-2023

Altair SimSolid® © 2015-2023

Altair® ultraFluidX® © 2010-2023

Altair® Virtual Wind Tunnel™ © 2012-2023

Altair® WinProp™ © 2000-2023

Altair® WRAP™ © 1998-2023

Altair® GateVision PRO™ © 2002-2023

Altair® RTLvision PRO™ © 2002-2023

Altair® SpiceVision PRO™ © 2002-2023

Altair® StarVision PRO™ © 2002-2023

Altair® EEvision™ © 2018-2023

Altair Packaged Solution Offerings (PSOs)

Altair® Automated Reporting Director™ © 2008-2022

Altair® e-Motor Director™ © 2019-2023

Altair® Geomechanics Director™ © 2011-2022

Altair® Impact Simulation Director™ © 2010-2022

Altair® Model Mesher Director™ © 2010-2023

Altair® NVH Director™ © 2010-2023

Altair® NVH Full Vehicle™ © 2022-2023

Altair® NVH Standard™ © 2022-2023

Altair® Squeak and Rattle Director™ © 2012-2023

Altair® Virtual Gauge Director™ © 2012-2023

Altair® Weld Certification Director™ © 2014-2023

Altair® Multi-Disciplinary Optimization Director™ © 2012-2023

Altair HPC & Cloud Products

Altair® PBS Professional® © 1994-2023

Altair® PBS Works™ © 2022-2023

Altair® Control™ ©2008-2023

Altair® Access™ ©2008-2023

Altair® Accelerator™ ©1995-2023

Altair® Accelerator™ Plus ©1995-2023

Altair® FlowTracer™ ©1995-2023

Altair® Allocator™ ©1995-2023

Altair® Monitor™ ©1995-2023

Altair® Hero™ ©1995-2023

Altair® Software Asset Optimization (SAO) ©2007-2023

Altair Mistral™ ©2022-2023

Altair® Grid Engine® ©2001, 2011-2023

Altair® DesignAI™ ©2022-2023

Altair Breeze™ ©2022-2023

Altair® NavOps® © 2022-2023

Altair® Unlimited™ © 2022-2023

Altair Data Analytics Products

Altair Analytics Workbench™ © 2002-2023

Altair® Knowledge Studio® © 1994-2023

Altair® Knowledge Studio®for Apache Spark © 1994-2023

Altair® Knowledge Seeker™ © 1994-2023

Altair® Knowledge Hub™ © 2017-2023

Altair® Monarch® © 1996-2023

Altair® Panopticon™ © 2004-2023

Altair® SmartWorks™ © 2021-2023

Altair SLC™ ©2002-2023

Altair SmartWorks Hub™ ©2002-2023

Altair® RapidMiner® © 2001-2023

Altair One™ ©1994-2023

Third Party Software Licenses

AcuConsole contains material licensed from Intelligent Light (www.ilight.com) and used by permission.

For a complete list of Altair Accelerator Third Party Software Licenses, please click here.

Technical Support

Altair provides comprehensive software support via web FAQs, tutorials, training classes, telephone and e-mail.

Altair One Customer Portal

Altair One (https://altairone.com/) is Altair's customer portal giving you access to product downloads, Knowledge Base and customer support. We strongly recommend that all users create an Altair One account and use it as their primary means of requesting technical support.

Once your customer portal account is set up, you can directly get to your support page via this link: www.altair.com/customer-support/.

Altair Training Classes

Altair training courses provide a hands-on introduction to our products, focusing on overall functionality. Courses are conducted at our main and regional offices or at your facility. If you are interested in training at your facility, please contact your account manager for more details. If you do not know who your account manager is, e-mail your local support office and your account manager will contact you

Telephone and E-mail

If you are unable to contact Altair support via the customer portal, you may reach out to the technical support desk via phone or e-mail. You can use the following table as a reference to locate the support office for your region.

When contacting Altair support, please specify the product and version number you are using along with a detailed description of the problem. It is beneficial for the support engineer to know what type of workstation, operating system, RAM, and graphics board you have, so please include that in your communication.

Location	Telephone	E-mail
Australia	+61 3 9866 5557 +61 4 1486 0829	anz-pbssupport@altair.com
China	+86 21 6117 1666	pbs@altair.com.cn
France	+33 (0)1 4133 0992	pbssupport@europe.altair.com
Germany	+49 (0)7031 6208 22	pbssupport@europe.altair.com
India	+91 80 66 29 4500 +1 800 208 9234 (Toll Free)	pbs-support@india.altair.com
Italy	+39 800 905595	pbssupport@europe.altair.com
Japan	+81 3 6225 5821	pbs@altairjp.co.jp
Korea	+82 70 4050 9200	support@altair.co.kr

◢◣ ALTAIR

Location	Telephone	E-mail
Malaysia	+91 80 66 29 4500 +1 800 208 9234 (Toll Free)	pbs-support@india.altair.com
North America	+1 248 614 2425	pbssupport@altair.com
Russia	+49 7031 6208 22	pbssupport@europe.altair.com
Scandinavia	+46 (0) 46 460 2828	pbssupport@europe.altair.com
Singapore	+91 80 66 29 4500 +1 800 208 9234 (Toll Free)	pbs-support@india.altair.com
South Africa	+27 21 831 1500	pbssupport@europe.altair.com
South America	+55 11 3884 0414	br_support@altair.com
United Kingdom	+44 (0)1926 468 600	pbssupport@europe.altair.com

See www.altair.com for complete information on Altair, our team and our products.

Index

A

B

C

D

E

F

G

W