Altair License Management System

Version 2025.0

www.altair.com

Contents

Altair License Management System

Introduction

This guide explains the concepts behind the Altair Units licensing system, and provides instructions for installation and operation on supported platforms. Useful cross references to the previous FLEXlm based system are included.

New for ALM 12.1 and above is almutilGUI. This is a graphical version of almutil. Where you see almutil used in this document, you may also use almutilGUI on Windows, Mac/OSX and Linux (64bit only).

About the Altair License Management System

The Altair License Management System (ALM) provides a units-based licensing model for all Altair software related to CAE, on-demand computing, and business intelligence. This new system has been designed to enable Altair Partner Alliance (selected third party products) to be licensed via Altair Units as well.

The Altair License Management System is comprised of two components: The License Server and the Usage Reporting System (URT). The license system is based on X-Formations LM-X license server. The URT is a java application that reports anonymous transactional logs to Altair. As part of the license agreement that enables partner software to be used, product usage must be reported to Altair.

The "on premise" version (LM-X) uses 2 way TCP/IP on the designated port.

NOTE: By default the server is installed and configured to use TCP/IP port 6200. If you have another application using that port, the server will not start. You can simply edit the altair-serv.cfg file after install to change the port used by the server.

Conventions Used in this Guide

This guide uses the following conventions:

Bold Italic	All commands from the user interface. Options, menus, buttons, and dialog box names are bolded, but not italicized. Example: On the **Welcome** screen, click ***Next***.
`Courier`	The path of a program or folder; a web address; a file name or component; text that the user is expected to enter. Example: The default path is `C:\Altair\hw13.0.`
Italic	For emphasis or for introducing a new term; for names of topics (headings) within this guide. Example: See *Minimum Requirements* for more details.

Questions regarding the software and license installation may be directed to Altair Support. See the *Contact* page for further contact information.

Chapter 1: Altair Units Licensing

Altair licensed products use one of two different licensing schemes: **units based** and **feature based**. Units based licensing requires each application to draw a certain number of license features from a central license server in order to become licensed. Each application has a different cost and is required to draw that cost from the license server. Feature based licensing simply requires one license feature to be drawn for each seat of the application. Most applications are units based, providing a unique advantage to leverage purchased Altair Units.

This chapter explains how the Altair Licensing Model works.

1.1 Altair Units Licensing

Purchasing Altair Units (AU) allows customers to access an entire suite of products as well as many partner products. AUs are represented in the license file as HyperWorks features. (See the description of a feature block in *Appendix A*.) As of HyperWorks 12.0, each purchased Altair Unit is represented by 1000 HyperWorks features in the license file. For example, a company purchases 21 Altair Units and receives 21000 HyperWorks features, the cost of running a single copy of HyperMesh.

HyperWorks features are the currency of the Altair licensing system. By sharing this common currency, the large suite of Altair applications and Altair Partner Alliance applications are able to share the purchased units.

Units based licensing is a three-feature process that occurs during a license checkout. In order for an application to be successfully licensed, all three features must be successfully executed from the same license server. These checkouts require the appropriate number of units for the following features:

- GlobalZoneXX feature (where XX is one of AM, EU or AP)
- HyperWorks feature
- Tracking feature (typically the product name prefixed with HW)

NOTE: With the 13.0 release the server now supports "Atomic Checkouts" This means that the 3 features are checked out as 1 transaction vs the old way which was 3 individual transactions. This has many advantages, primarily that an application can search multiple license servers and will only get "locked" to one if it can satisfy the entire request.

1.1a Global Zone Based Licensing

Altair Units licensing is divided into three distinct global zones. License files are then enabled for one or more of these global zones. A license server may reside anywhere in the world and serve licenses for any or all zones. When a client application requests a license from the license server, the license server must have enough of the proper GlobalZoneXX features available based on the client's location. These Global Zone features work hand-in-hand with the HyperWorks features described below to enforce overall usage.

Definitions of the Global Zones are described below.

- American Global Zone (GlobalZoneAM): This global zone covers Hawaii to Iceland, including North and South America.
  - GMT-12 < Local time < GMT
- European Global Zone (GlobalZoneEU): This global zone covers Europe, the Middle East, and Africa.
  - GMT <= Local time < GMT + 4
- Asia Pacific Global Zone (GlobalZoneAP): This global zone covers Asia Pacific (India and China) and Australia.
  - GMT +4 <= Local time <= GMT + 12

Example: If a client computer is in New York (GMT-5) and is starting an application requiring 21000 units, it would need to successfully checkout 21000 GlobalZoneAM features from the license server to proceed. This is the first of three checkouts required for the application to proceed.

1.1b HyperWorks Features

The HyperWorks feature in the license file is the *currency* of Altair Units. An application must check out the appropriate number of HyperWorks features from the same license server as the GlobalZone features in order to proceed. This is the second of the three required checkouts.

1.1c Tracking Features

Tracking features are used to enable and identify individual applications. Each application has an assigned tracking feature. This feature is typically the name of the application with an **HW** prefix. (e.g. HyperMesh is HWHyperMesh). These tracking features are the third and final checkout required to license an application.

1.1d Unit Stacking and Leveling

Unit based checkouts may be **stacked** or **leveled**. Stacked features drawn from the license server always increment the total features in use by the requested amount. Applications that stack always increase the total feature count by the requested amount. Tracking features are ALWAYS stacked.

Leveled features MAY increase the total features in use. A set of leveled features allows a group of applications to run simultaneously, but only draws the amount of features that are associated with the most costly application. The criteria for leveling are typically the same username from the same host. As a simple example, a user starts HyperMesh and draws 21000 features. Then on the same computer he starts HyperGraph. HyperGraph draws 6000 features, but is leveled against the 21000 already in use. Therefore, the total remains 21000. If the user had started an application that required 30000 features, the total draw of the leveled applications would increase to 30000, not 51000. The only features that leveling may apply to are GlobalZone and HyperWorks.

1.2 Licensing Pools

1.2a Using License Pools

A license pool is a single collection of features available on a license server. A single license pool can be constructed for use with either a high availability setup (three servers) or a single-server license. You can create multiple license pools by dividing your total licenses into two or more groups. For example: If you had a total of 6300 HyperWorks features, you could create two license pools, one with 3500 features and the other with 2800 features on two different servers. This requires obtaining the appropriate license files from Altair.

Multiple license pools provide the ability to load balance the features, allowing some units to be available if a license server(s) fails, and obtain licenses from another license resource when the primary one reaches its maximum limit.

1.2b Leveling with Multiple License Pools

Since each license pool functions independently of any other license pool, a license pool cannot communicate with any other license pool. Therefore, features may not be leveled across multiple servers. Leveling can only occur within a single pool on an individual license server.

1.3 License Decay

The licensing system contains a feature that allows a decay function (reduction of features used) to be implemented for certain applications or scenarios. The files that control this feature are in `<installation_directory>/decay`. The license server looks for files that end with `.decay` in this directory.

To install a new decay file, simply place it in the decay directory and restart the license server. To remove a decay file, move it out of the decay directory and restart the license server.

Status messages for the decay files are printed in the server's log file as defined in the altair-serv.cfg file.

The decay files are created and distributed by Altair as part of either the License Server package or as an update via the support web site.

1.4 Feature Based Licensing

Feature based licenses are used for products that are licensed outside of the Altair Units ecosystem. The features follow the naming convention <PRODUCT>Feature and may or may not draw a GlobalZone feature as well depending on how the product is licensed.

Chapter 2: Altair License Management System

The Altair License Management System provides a common unit-based licensing model for all Altair software. The key components of the Altair License Management System are: The Altair License Server (ALS), the Usage Reporting Tool (URT), and the Altair Application Server (AAS).

ALS	Altair License Server is an LM-X license server built and distributed by Altair.
URT	Usage Reporting Tool sends logs back to Altair. **This is set to run AUTOMATICALLY during installation**.
AAS	Altair Application Server references the application server that is deployed to accept usage logs from the customers.

The License Server is an application that runs on supported platforms and serves licenses to Altair Licensing System enabled clients. This includes a large suite of Altair products as well as a collection of third party applications via the Altair Partner Alliance.

The Usage Reporting System contains the Usage Reporting Tool (URT) and the Altair Application Server (AAS). The reporting system gathers and sends the anonymous transaction logs from the license server and sends them to Altair for processing. The URT resides on the customer's license server and the AAS resides on the Altair hosted server.

When a customer signs a new license agreement, they agree to provide certain software usage data to Altair. The Altair License Management System can be installed to use an automated method or one of several manual methods for delivering usage data. Either way, any data delivered to Altair is stripped of all personal or company data (e.g. usernames, hostnames, etc.).

2.1 Altair License Server (ALS)

The ALS is built on X-Formation's LM-X license manager suite. The main components of the system are:

- License server (`lmx-serv` and `liblmxvendor.so/dll`)
- Configuration files (`alus.conf` and optional `.cfg` file)
- License file (`altair_lic.dat` or equiv.)
- Application programs (HyperWorks, PBS Professional, FEKO…)
- Supplied utilities (almutil, almutilGUI…)

2.1a License Server Daemon (lmx-serv)

The Altair License Server (ALS) serves network licenses. It runs on a license server machine that can be reached by network clients. It is typically setup to automatically restart on machine restart and to communicate via a specific TCP/IP port. The default port is 6200. The server process requires one (optionally, two) configuration file to run. These are `alus.conf` (required) and a command line specified `.cfg` file, if used. The `alus.conf` file contains Altair specific configuration information and the `.cfg` file contains generic LM-X configuration data.

Multiple LM-X servers from different vendors may run on a single machine by specifying different ports. However, only a single server from any specific vendor may run on a machine at one time. A LM-X based server may also run side-by-side with the licensing systems of other licensing system vendors such as FLEXlm and others.

The ALS also creates transaction logs that provide detailed license usage information. These logs are plain readable text and documented (*Chapter 8, Usage Logging System*) for ease of use. Two versions of these logs are available: anonymous and detailed. By default, only the anonymous logs are written. This can be changed in the `alus.conf` file. Command line options for the ALS are listed in *Section 7.1*.

2.1b Configuration Files

The `alus.conf` file stores configuration information required by the ALS and the URT to function properly. The ALS needs to find the `alus.conf` file in order to start. By default, it is located in the current working directory or you can specify the location (the preferred method) by setting the environment variable ALUS_CONF_FILE on all platforms or by specifying the following registry key on Windows.

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Altair Engineering, Inc.\Licensing\, set ALUS_CONF_FILE to the full pathname of the `alus.conf` file.

The example `alus.conf` files for Windows and Unix are listed in *Appendix C: Example of alus.conf File*.

By default, the installer also creates `altair-serv.cfg`. This file contains information that designates the location of the license file, the location in which to place the log files, and which TCP/IP port to use. It also provides other information used to configure the server. A `sample_config.cfg` is also placed in the installation directory that contains all the options available in the config file. This file has much of the same functionality as the FLEXlm options file. It is also used to configure Highly Available (HAL)

server setups. This setup uses three servers to provide redundant license servers that continue to work in the event of the primary server going down. See *Section 7.2, Setting up a High Availability License (HAL) System and/or Multiple Servers*, for more details.

2.1c License File (altair_lic.dat)

The license file is a plain text document that contains license information. Each block of text in the file is known as a feature block. It contains all the details for that feature including its name, quantity, sharing, extra details, and whether it is server or nodelocked. A server based license file can be identified by its feature blocks containing the keyword COUNT=value.

The default name that Altair products use for this license file is `altair_lic.dat`.

Detailed information on the license file contents is provided in *Appendix A*.

2.1d Using the Altair License Server

In order for applications to use the ALS, the environment variable, ALTAIR_LICENSE_PATH, must point to the appropriate location. Multiple license paths are allowed and should be separated by a semicolon (;) on Windows and a colon (:) on Unix/Linux. The proper format for a network license location is port@hostname. If you are using a local license file, simply set the value to the full pathname of the file.

A few examples of license paths

On Windows:

ALTAIR_LICENSE_PATH=6200@server.foo.bar.com

ALTAIR_LICENSE_PATH=6200@srv1;6200@srv2;6200@srv3

ALTAIR_LICENSE_PATH=c:\Program Files\Altair\Licensing12.0\altair_lic.dat

On Linux/Unix:

ALTAIR_LICENSE_PATH=6200@server.foo.bar.com

ALTAIR_LICENSE_PATH=6200@srv1:6200@srv2:6200@srv3

ALTAIR_LICENSE_PATH=/usr/local/altair/licensing12.0/altair_lic.dat

For HAL setups list the three servers in the order: primary;secondary;tertiary.

2.2 Usage Reporting Tool

The Usage Reporting Tool (URT) is a tool that is launched by the license server once per week. When it is initiated, it reads the anonymous log files recorded by the ALS, validates them, and transmits the data back to Altair. In order to transmit valid files from the customer site to the Altair site, the URT must be configured by the customer to contact Altair through their proxy server if applicable.

The URT is a one-way communication initiated at the client's machine. It simply communicates via HTTP or HTTPS to the Altair server and delivers the anonymized transactional logs. Several security steps ensure that only authorized clients can connect to the Altair Server.

If the connection does not go through, the URT will make several attempts before moving the files to the undeliverable directory.

If the user chooses to disable the automatic reporting system, they must send the log files to Altair manually per the Altair license agreement using one of the manual methods provided in *Appendix E, Sending Reports Manually*.

During installation, the URT is configured and set to run automatically. See *Appendix E* to change this once the installation is complete.

2.3 Transactional Log File(s)

The ALM contains a transactional logging system. As long as the server is running, it logs all transactions to the location specified in the `alus.conf` file. The level of logging is also specified and can be anonymous (default) or detailed. The anonymous logs make sure that no private data is disclosed. All private data such as username, host, and share strings are one-way hashed to make the original values irretrievable. These anonymous logs are sent to Altair for usage reporting. If detailed logging is enabled, a second set of transaction logs is also produced, keeping the private data in a legible format. These are NOT sent to Altair, and are stored in a separate directory named detailed. For details on the log file contents, see *Chapter 8, Usage Logging System*.

Chapter 3: Preparing to Install Altair License Management System

This chapter summarizes the items and procedures needed to install and successfully run the Altair License Management System on Linux, Unix, or Windows.

The general procedure for installing and activating the license management system are:

- Determine the server(s) for the license manager and the Usage Reporting System
- Download the proper package(s) from Altair One
- Install the Altair License Management System on the designated machine(s) and, if required, install the hardware key (dongle) on the appropriate machine(s)
- Obtain a valid license file for the machine
- Activate the license manager and the usage reporting system

Requirements:

- Access to the Altair web site or Altair One
- Altair License Management System Software
- A valid HostID of the server machine (use almutil –hostid or almtuil GUI)
- License file

3.1 Installation Media and Supported Platforms

The Altair License Management System software is available for download by registered customers through Altair One https://altairone.com

Download the images for the platforms on which you expect to run the license manager and usage report system. If you plan to run the license manager and the usage report system on multiple servers, download the proper packages for all servers.

3.2 Compatibility

You cannot run two versions of the same type of Altair license managers on the same server. However, you may run older FLEXlm based versions (10.0 and prior) alongside the 11.0+ system. Be sure to specify different ports for each server.

3.3 Selecting Licenses Server Systems

3.3a Resources used by License Servers

When you select a license server, take the following resources into consideration:

Sockets

Each licensed application connected to a license server uses one or more sockets. The number of sockets available to the license server is defined by the per-process system limit for file descriptors. The total number of sockets used by the license server is slightly larger than the total number needed by the licensed applications that are served by it. If the number of sockets required by the license server on a single machine becomes excessive, the license file can be split into more than one file, on multiple servers.

CPU Time

For small groups of clients, the license server uses very little CPU time, perhaps only a few seconds of CPU time over many days. For a large number of clients, or for high check-out/check-in activity levels, the amount of CPU time consumed by the server may become significant. If such a situation occurs, you may need to ensure that the server machine you select has an adequate number of CPU cycles.

Disk Space

The license package requires about 100 MB of disk space to install. The log files created by the system vary in size, depending on activity. It is recommended that the file system where the log files are to be written has sufficient space to store many days or months worth of log files. There should be a minimum of 500 MB of space for this purpose.

Memory

The license manager daemon uses a variable amount of system RAM. Typical memory usage is in the range of 15-50 MB resident and a virtual set size of 90-150 MB. For very busy servers with a lot of clients the Resident set size could grow to several 100MB

Network Bandwidth

The licensing process requires a fairly small amount of network bandwidth. Typical networks running over Ethernet should not produce any problems. The system uses TCP/IP for all communications on the configured port, and uses this one port for all communications.

Remote Mounted Disks

Altair highly recommends that the software and all logs reside on disks local to the license server. Testing is only done on local disks/directories. Although remote ones may work, they are not tested

Redundant License Servers

The system supports both HAL (High Availability) servers and server lists. HAL servers act as a quorum to serve up one pool of licenses, allowing one machine to go down without affecting license availability. Multiple independent servers can be used to split up the total license pool and clients can access them via a server list.

Redundant licenses are covered in *Chapter 7, Section 2*.

3.4 Hostid and Dongle Support

Altair supports a subset of the available Hostid values from X-Formation as well as hardware dongles (USB keys) on some platforms. In order to use a dongle, you must have an open USB port.

3.4a Hostid

The acceptable values of Hostid are (all may not be available on any given platform):

- ETHERNET
- DONGLE
- HARDDISK (this is NOT the same HARDDISK value as used by FLEXlm)
- LONG
- BIOS
- WIN_INSTALL_ID

When requesting a license file, use one of the above Hostid values, and the Altair provided utility, `almutil`, to obtain your Hostid.

Additionally certain cloud instance hostids may be used by setting the environment variable ALM_CLOUD_HOSTID_ENABLED to one of the following values:

- all
- aws
- azure
- gce
- oci

3.4b USB Hardware Keys (Dongles)

In order to use a dongle, you must have a free/open USB port on the machine that will be your license server.

If you have acquired a hardware dongle to use for licensing, you must obtain the dongle ID via the software before requesting your license file. Hardware dongles are supported on the following platforms:

- Windows
- OS X
- Linux

In order for the dongle to function properly, you must first install the correct drivers for your platform. These drivers are available from X-Formation. Download and install the driver from the following URL:

http://www.x-formation.com/lm-x_license_manager/enduser_downloads.html

If that is not available, try

http://www.safenet-inc.com/support-downloads/sentinel-drivers/

and look for the "Sentinel Drivers" for your platform

Once the driver is installed and the dongle is in place, use the `almutil -hostid` command to query for all the Hostids on the machine and report the DONGLE Hostid.

NOTE: Due to a change in the way DONGLES were handled in newer versions of LM-X you will need to set the environment variable ALM_DONGLE_ENABLED=t in order to use a dongle with client applications. This is NOT true for the license server itself, but is true for almutil and client applications.

3.4c Other Notes

The user that the license server runs as must have a home directory.

Chapter 4: Installing Altair License Management System on Linux

Consult the table listed in *Chapter 3* to download the package that is appropriate for the machine that will be your license server. You can download the package from the Altair web site (www.altair.com), or from the CD media sent by Altair.

In order to install the software with automatic startup during reboot, you will need to be root (super-user) to run the installer. If you do not have root/admin access or do not want to install the startup scripts, you may add the following option to the command line at when running the installer: -DNO_ROOT=t Also note that with this version you must set LD_LIBRARY_PATH (or appropriate setting for your OS) to include the installdir/bin directory in order for lmx-serv to find liblmxvendor.so/dll if you don't use the Altair supplied start/stop scripts.

It is also important to allow traffic on the port used by the Licensing systems. By default, this is port 6200. You may have to enable traffic on this port for both the license server machine and the clients if you are running any type of firewall or security software.

4.0a Install Software

1. Download the proper package(s).
2. Log in as **root** on the machine where the license package will be installed.

   You need to log in as **root** to set up the license init scripts to automatically launch the daemons at boot time.
3. Execute the installer package.
   a. The package tries to invoke in GUI mode. This requires an X server and connection.

b. If you would like to run in non GUI mode, open a command shell and make sure that the environment variable DISPLAY is unset/blank and add the `-i console` option to the command line on invocation (`./package_name -i console`) .

c. Also note that for many platforms the installer is a self-extracting shell script and needs to either have the executable bit set or be invoked by `/bin/sh package_name.bin`

4. If there is a current installation of the license server the installer will try to detect it and ask if you would like to use the license file and config file from that installation.

5. Answer the questions and designate the location where you want to install the licensing package. If you provide a nonexistent directory, the installer tries to create it.

   The installer installs Altair License Manager, the usage reporting package, and a Java runtime environment.

6. The installer asks whether you already have a license file. If you do, enter its location in the install wizard. If you do not have one, the installer creates a blank `altair_lic.dat` file in the `<install_location>` and you can replace it later. The Hostid of the machine is displayed at the end of the installation if you should need it.

7. The installer asks if you want to have the init scripts for automatic startup installed, answer **yes** or **no**.

8. The installer attempts to configure the URT properly. If it cannot contact the Altair Server it will prompt you for your proxy information. Follow the prompts and enter the appropriate information. Leave any unneeded fields blank (e.g. username and password)

   If you do not have the data to configure the proxy, you can skip this step and manually modify the `<Install_directory>/alus.conf` file later.

   Please note that the proxy password is stored in the configuration file in plain text. It is the administrator's responsibility to limit access to this file.

   You have now completed the software installation.

4.0b Install a License File

You will need a license file supplied by Altair Engineering, Inc. to complete the licensing process. If you do not have a license, contact your account manager at Altair Engineering, Inc. or your local Altair support office for assistance. With your request, provide the Hostid value of your license server.

1. To obtain the host ID for the machine, run the following command:

   ```
   <Install_location>/bin/almutil -hostid
   ```

   This provides all the Altair supported valid Hostids for this machine. If you are using one of the Ethernet based Hostids, choose one that is on a permanent interface. Some interfaces are for VPN's or other software based adapters that may not always be present.

2. When you receive your license file, place it in the `<Install_location>` directory. By default it should be in a file named `altair_lic.dat` You can check the name by looking in the `altair-serv.cfg` file and finding the value for LICENSE_FILE.

3. Change the setting in the `altair-serv.cfg` file to point to your new license file, if necessary.

4.0c Starting, Stopping and Resetting the License Daemons

The installer provides startup scripts in the appropriate place for your version of the operating system. You should use the systemctl command to start, stop or restart the service, e.g. # systemctl restart altairlmxd. By default, the service will run as root. If you do not want this arrangement, you may make the following changes.

1. Edit the `altarlmxd.service` file, the exact location may vary between distros but a common location is:

   /etc/systemd/system/multi-user.target.wants/altairlmxd.service

   Configure the service to run as the user and/or group of your choosing by using the User and/or Group directives in the [Service] section of the unit file. E.g.:

   [Service]
   User=almuser
   Group=almgroup
   …
   Before the changes can take effect, you will need to refresh the systemd configuration e.g.:

   # systemctl daemon-reload

2. Go to `<Install_directory>` and change the ownership of all files and directories in the `<Install_directory>` to the user you are running the license system as. These changes must be performed recursively to get all the subdirectories as well.

Chapter 5: Installing Altair License Management System on Windows

Consult the table listed in Chapter 3 to download the package that is appropriate for the machine that will be your license server. You can download the package from the Altair web site (www.altair.com), or from the CD media sent by Altair.

In order to install the software with automatic startup during reboot, you will need to have admin rights for the account from which you are logged in. If you do not have admin access or do not want to install the ALS as a service, you may add the following option to the command line when running the installer: -DNO_ROOT=t.

It is also important to allow traffic on the port used by the Licensing systems. By default, this is port 6200. You may have to enable traffic on this port for both the license server machine and the clients if you are running any type of firewall or security software.

5.0a Install Software

1. Download the proper package(s).
2. Execute the installer package.

   To execute in non-GUI mode, open a command shell and execute the installer with the argument `-i console (altair_licensing_<version>.win64_x64.exe –i console`).
3. If there is a current installation of the license server the licensing the installer will try to detect it and ask if you would like to use the license file and config file from that installation.
4. Answer the questions and designate the location where you want to install the licensing package. If you provide a nonexistent directory, the installer will try to create it.
5. The installer asks whether you already have a license file. If you do, enter its location in the install wizard. If you do not have one, the installer creates a blank `altair_lic.dat` file in the `<Install_location>` and you can replace it later. The Hostid of the machine is displayed at the end of the installation if you should need it.
6. The installer asks if you want to have the service for automatic startup installed, answer **yes** or **no**.
7. The installer attempts to configure the URT properly. If it cannot contact the Altair Server, it will prompt you for your proxy information. Follow the prompts and enter the appropriate information. Leave any unneeded fields blank (e.g. username and password)

   If you do not have the data to configure the proxy, you can skip this step and manually modify the `<Install_directory>/alus.conf` file later.

   Please note that the proxy password is stored in the configuration file in plain text. It is the administrator's responsibility to limit access to this file.

   You have now completed the software installation.

5.0b Install a License File

You will need a license file supplied by Altair Engineering, Inc. to complete the licensing process. If you do not have a license, contact your account manager at Altair Engineering, Inc. or your local Altair support office for assistance. With your request, provide the Hostid value of your license server.

1. To obtain the host ID for the machine, run the following command in a command shell:

   ```
   C:\<Install_location>\bin\almutil –hostid
   ```

   This provides all the Altair supported valid Hostids for this machine. If you are using one of the Ethernet based Hostids, choose one that is on a permanent interface. Some interfaces are for VPN's or other software based adapters that may not always be present.

2. When you receive your license file, place it in the `<Install_location>` directory. By default, it should be in a file named `altair_lic.dat`. You can check the name by looking in the `altair-serv.cfg` file and finding the value for LICENSE_FILE.

3. Change the setting in the `altair-serv.cfg` file to point to your new license file, if necessary.

5.0c Starting and Stopping the License Daemons

The installer creates a windows service named *Altair License Service* and sets it for automatic startup. The installer will start the license daemon upon completion. If you want to stop the server, you can:

1. Right click on ***My Computer*** and select ***Manage***.
2. Click ***Services and Applications***.
3. Click ***Services***.
4. Right click on ***Altair License Server*** and select ***Stop***.

If you did not have the installer create the service during installation, you will have to use the Windows "sc" command to do so. Please refer to the *Licensing Release Notes* and the script named `install_service.bat` in the `install/scripts` directory.

Chapter 6: Installing Altair License Management System on Mac OS X

Consult the table listed in *Chapter 3* to download the package that is appropriate for the machine that will be your license server. You can download the package from the Altair web site (www.altair.com), or from the CD media sent by Altair.

In order to install the software with automatic startup during reboot, you will need to be root (super-user) to run the installer. If you do not have root/admin access or do not want to install the startup scripts, you may add the following option to the command line at when running the installer: -DNO_ROOT=t Also note that with this version you must set DYLD_LIBRARY_PATH to include the installdir/bin directory in order for lmx-serv to find liblmxvendor.dylib if you don't use the Altair supplied start/stop scripts.

6.0a Install Software

1. Download the proper package(s).
2. Log in on the machine where the license package will be installed.

   You will need to be an administrator, or know a username and password that can be used to install the package.
3. Execute the installer package. This is a .zip file and needs to be unpacked. Double click on the zip file to unpack it, then double click on the unpacked package

   The default installation directory is in `/Applications/Altair/licensing<version>`
4. Answer the questions and designate the location where you want to install the licensing package. If you provide a nonexistent directory, the installer tries to create it.

   The installer installs Altair License Manager, the usage reporting package, and a Java runtime environment.
5. The installer asks whether you already have a license file. If you do, enter its location in the install wizard. If you do not have one, the installer creates a blank `altair_lic.dat` file in the `<Install_location>` and you can replace it later. The Hostid of the machine is displayed at the end of the installation if you should need it.
6. The installer asks if you want to have the init scripts for automatic startup installed, answer ***yes*** or ***no***.
7. The installer attempts to configure the URT properly. If it cannot contact the Altair Server, it will prompt you for your proxy information. Follow the prompts and enter the appropriate information. Leave any unneeded fields blank (e.g. username and password)

   If you do not have the data to configure the proxy, you can skip this step and manually modify the `<Install_directory>/alus.conf` file later.

   Please note that the proxy password is stored in the configuration file in plain text. It is the administrator's responsibility to limit access to this file.

   You have now completed the software installation.

6.0b Install a License File

1. You will need a license file supplied by Altair Engineering, Inc. to complete the licensing process. If you do not have a license, contact your account manager at Altair Engineering, Inc. or your local Altair support office for assistance. With your request, provide the Hostid value of your license server.
2. To obtain the host ID for the machine, use the almutilGUI tool or run the following command in a terminal window:

   ```
   <Install_location>/bin/almutil -hostid
   ```

   This provides all the Altair supported valid Hostids for this machine. If you are using one of the Ethernet based Hostids, choose one that is on a permanent interface. Some interfaces are for VPNs or other software based adapters that may not always be present.
3. When you receive your license file, place it in the `<Install_location>` directory. By default it should be in a file named `altair_lic.dat You` can check the name by looking in the `altair-serv.cfg` file and finding the value for LICENSE_FILE.
4. Change the setting in the `altair-serv.cfg` file to point to your new license file, if necessary.
5. Once you have put the file in the proper location, you will need to restart the license server to pick it up. See the next section and use the '`altairlmxd restart`' command to restart the license server.

6.0c Starting, Stopping or Resetting the License Daemons

The installer will install the appropriate plist for use with launchd. This is the appropriate way to launch daemons on OSX. The com.altair.lmx-serv.plist will be installed in /Library/LaunchDaemons and is owned by root. The server should start and stop on its own when the machine is rebooted. The commands below are useful form manual intervention. You will have to know a valid Administrator account and/or password to use them. Run the following commands in a Terminal session.

To check the status to see if it is running or not:

sudo launchctl list | grep lmx-serv

If the output line starts with a '-' then it is not running, if it starts with a number, it is running.

Starting the server:

sudo launchctl start com.altair.lmx-serv

Stopping the server:

sudo launchctl stop com.altair.lmx-serv

Setting the server to NOT start after the next reboot:

sudo launchctl unload –w /Library/LaunchDaemons/com.altair.lmx-serv.plist

Setting the server TO start after reboot:

sudo launchctl load –w /Library/LaunchDaemons/com.altair.lmx-serv.plist

Chapter 7: Operations and Maintenance

This chapter covers information you may find helpful in keeping the license system functioning normally. It contains information on setting up a HAL configuration of servers, administration tools, borrowing licenses, general troubleshooting, and maintenance.

7.1 Altair and LM-X Programs

Altair and X-Formation provide the following license administration tools and programs. These include almutil, almutilGUI (a GUI version of almutil) and the license server itself.

7.1a lmx-serv

lmx-serv is the actual license server. At a minimum, a valid `alus.conf` file must reside in the current working directory and you must have a valid license file. Everything that is needed, except the location of the `alus.conf` file, may be specified in a valid `.cfg` file. The location of the `alus.conf` file is taken from the environment variable ALUS_CONF_FILE or from the registry on Windows. The lmx-serv options are:

lmx-serv [options]

`-b or –background`	Run license server in background.
`-c or -config <configfile>`	Specify which config file to use.
`-l or –licpath <license path>`	Specify optional license file or a path to directory with license file(s).
`-lf or –logfile <logfile>`	Specify optional logfile.
`-port <port number>`	Specify optional port number.
`-h`	Show this help.

7.1b almutil and almutilGUI

almutil (and almutilGUI, the graphical version) provides Altair specific functionality. It is used to obtain one of the supported Hostids of the server and to manage license borrowing and return. While LM-X supports a variety of valid Hostids, only a subset is supported by Altair. Use the `–hostid` option of almutil to obtain a list of supported Hostids. Further details on borrowing are in *Section 7.5, Borrowing Licenses*.

Options for almutil are as follows:

almutil [option]

`-hostid`	Display Hostids for this system.
`-borrow -borrow <hours>`	Start the borrowing system, borrowed licenses will be available for <hours> number of hours.
`-return -return <feature>`	Return the feature specified by <feature>

Additional options for almutil include several reporting features as well as interacting with the Hosted HyperWorks Units (HHWU) system. In order to provide both familiar and more useful status data the following option was added to almutil:

```
-licstat[-host <host> -port <port> -feature <feature> -user <user> -inuse -collapse]
Display license statistics, by default list everything
          Information can be limited with optional arguments
          Multiple -feature <feature> and -user <user> can be specified
          -inuse will limit output to features with usage
          -inuse may not be used with -feature and -user options
          -collapse will collapse shared checkouts into one entry
```

For additional information you can run almutil --help

Altair now provides a graphical version of almutil. It provides all the functionality of almutil along with additional views of license usages, etc.

7.2 Setting Up a High Availability License (HAL) System and/or Multiple Servers

A HAL setup provides a greater level of availability over a single license server, providing options in case of machine or network failure. A HAL environment requires three license server machines capable of serving the licenses and having stable, low latency network connectivity between them. Using a HAL setup requires setting the clients to have the ability to see all three servers as well.

A HAL does not increase the number of available licenses or features, or provide any load balancing of the servers. It is simply meant to provide a failover mechanism in case of a hardware failure. To provide load balancing or multiple pools of licensing, you must setup up multiple independent servers.

Should you choose to have multiple servers providing a load balanced environment, you may setup multiple independent servers and set LMX_RANDOMIZE_PATH=1 on the clients. This will randomize the license servers set in ALTAIR_LICENSE_PATH.

7.2a Setting up a HAL Environment

You need three independent license server machines to set up a HAL environment. You are required to provide all three Hostids when you request your license file. Once you have your HAL license file, follow the directions in the earlier chapters, as appropriate, to install the license server on each machine as though it were a single server. Once that is done, you must perform the following additional steps.

1. Decide which machines will the primary, secondary, and tertiary server. Each of these has the following function:

HAL License Server Number	Role
Primary (1)	This HAL license server can allow clients to both checkout and borrow licenses exactly like a normal license server.
Secondary (2)	In the event that HAL license server #1 is down, this server can allow clients to checkout licenses, but will deny borrow requests.
Tertiary (3)	This HAL license server will deny any requests, but is required as a part of the configuration to ensure high availability.

2. Install the exact same HAL enabled license file on all three servers.
3. Edit the config file (`altair-serv.cfg`) and add the lines listed below. Make sure the lines and roles match on all three machines (it may be easier to edit one file and copy it to all three machines)

```
HAL_SERVER1 = port@primaryserver
HAL_SERVER2 = port@secondaryrserver
HAL_SERVER3 = port@tertiaryserver
```

4. Start the three servers.

7.2b Setting up Clients to use HAL and/or Multiple License Servers

In order for clients to take advantage of a HAL environment you must set ALTAIR_LICENSE_PATH to the appropriate value. The values should be separated by a semicolon (;) on Windows and a colon (:) on Unix/Linux. The proper format for a HAL license setup is shown below. You should list the servers in order, according to their roles of primary, secondary, and tertiary server.

On Windows:

ALTAIR_LICENSE_PATH=6200@primarysrv;6200@secondarysrv;6200@tertiarysrv

On Linux/Unix:

ALTAIR_LICENSE_PATH=6200@primarysrv:6200@secondarysrv:6200@tertiarysrv

This same syntax is used to specify multiple independent license servers as well. Simply list them in the order you would like them to be tried by the client.

7.3 LM-X Options File

The options file allows the license administrator to control various operating parameters of LM-X. A complete listing of the capabilities and of the sample is listed for reference:

```
# LM-X License server sample configuration file
#
# Copyright (C) X-Formation. All rights reserved.
#
# http://www.x-formation.com
#
#***********************************************
# lmx-serv.cfg notes:
#
# 1) Any line beginning with a '#' is a comment
# 2) User and hostnames are case-insensitive, but you're advised to
#   use those present in the log file to avoid typos.
# 3) License clients can be specified by user name (USER),
#   or computer name (HOST), or IP address (IPADDR)
#***********************************************
#***********************************************
# 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
#***********************************************
# Limit which networks the license server allows for client
# connections.
# When this setting is specified, the license server will only
# accept clients that connect from a network that uses the
# specified IP addresses. You can specify only one address
# for each IP version (one for IPV4 and one for IPV6), separated
# by space.
```

```
# This setting is useful when the license server is connected to
# more than one network (has more than one IP address) and you
# want to limit allowed connections based on which network the
# client is on.
# When this setting is unspecified, the license server accepts
# clients from all available networks.
#
# Syntax:
# TCP_BIND_ADDRESS = <IP_address_1 IP_address_2>
#*********************************************
# Example:
# TCP_BIND_ADDRESS = 192.168.21.321 8000:8000:8000:8000:abcd:1234:12df:fd54
#*********************************************
# Specify High Availability Licensing (HAL) servers.
# HAL enables redundant servers, so if one server
# goes down, two others will still work.
#
# HAL consists of 3 specified servers, at least 2 of which
# must be up and running at all times.
#
# Each HAL_SERVER line indicates a license server
# that has HAL enabled by its license(s). Each HAL server
# has a specific role, and should be specified in terms of how
# many resources each server has:
#
# HAL_SERVER1 is your master server, which
# allows both CHECKOUT and BORROW.
# HAL_SERVER1 should be your most powerful server.
#
# HAL_SERVER2 is your first slave server,
# which allows CHECKOUT but denies BORROW
# in the event that your master server goes down.
# HAL_SERVER2 should be your second most powerful server.
#
# HAL_SERVER3 is part of your configuration to ensure
# that everything works as expected, and does not
# allow any CHECKOUT or BORROW requests.
# HAL_SERVER3 should be your least powerful server.
```

```
#
# Syntax:
#
# HAL_SERVER<server_number> = [port]@hostname
# or
# HAL_SERVER<server_number> = [port]@IP_address
#
# Port is optional.
#
# Important: The HAL_SERVER list must be identical
# on all your servers for HAL to function properly.
#
# See the LM-X End Users Guide for further information
# about setting up HAL servers.
#************************************************
# Examples:
# HAL_SERVER1 = 6200@server1
# HAL_SERVER2 = 6200@server2
# HAL_SERVER3 = 6200@server3
#************************************************
# Set the log file path:
# It is preferred to write out the full path.
#
#************************************************
# Examples:
# LOG_FILE = c:\program files\lmx-server.log
# LOG_FILE = /home/user1/lmx-serv.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.
#
#************************************************
```

```
# Examples:
LOG_FORMAT = NORMAL
#************************************************
# Exclude messages from the log.
# The following messages can be excluded:
# CHECKOUT, CHECKIN, STATUS, BORROW, BORROW_RETURN,
# REMOVE_USER, REMOTE_RESTART, REMOTE_SHUTDOWN or
# AUTOMATIC_DISCOVERY
#
# Syntax:
# LOG_EXCLUDE = <message1, message2, etc.>
#************************************************
# Example:
# LOG_EXCLUDE = CHECKOUT, CHECKIN, STATUS
#************************************************
# Set the minimum elapsed time for user removal.
# This will set a minimum time that must elapse from the connection
# before a user can be removed using lmxendutil.
# This time is entered in seconds, and must be equal to or greater than
# the number of seconds specified by your application vendor.
# Default minimum time is 120 seconds.
# If the time is set to -1, user removals will not be allowed.
#************************************************
# Example:
# MIN_USER_REMOVE_TIME = 120
#************************************************
# Set a license file path:
#
# On Windows: If no file is set, the license server
# will look for <vendor>.lic in the same directory as the license server.
# On Unix: If no file is set, the license server will look for
# /usr/x-formation/<vendor>.lic
#
# In both cases, the filenames must be lowercase.
#
# You can specify one or multiple paths as needed.
#
#************************************************
```

```
# Examples:
# LICENSE_FILE = d:\server\network.lic
# LICENSE_FILE = c:\extra_file.lic
# LICENSE_FILE = /home/user1/floating_license.lic
# LICENSE_FILE = /home/user1/floating_license2.lic
#************************************************
# Specify a pay-per-use usage database, which can be used for billing
# purposes.
#
# The format of this database and an example of data printout is
# described in the LM-X end user documentation.
#
#************************************************
# Examples:
# USAGE_DATABASE = d:\server\usage.db
# USAGE_DATABASE = /home/user1/usage.db
#************************************************
# Specify pay-per-use detail level.
# STANDARD includes basic usage information.
# DETAILED includes user information in addition
# to the basic usage information.
# USAGE_LEVEL = STANDARD
#************************************************
# Specify the remote administration password which is used when remotely
# stopping and restarting the license server and removing users from it.
#
# The password is case-sensitive.
#************************************************
# REMOTE_ACCESS_PASSWORD = MyPassword123
#************************************************
# Specify client data returned in a STATUS request
# FULL includes full client data: usernames, hostnames
#     IP address will be visible to any caller, e.g. almutil
# NONE - Do not return *any* client data
#
# Default is FULL
#
# *NOTE* FULL client data is always returned if the request
```

```
# comes from localhost (i.e. the server itself).
#************************************************
# STATUS_CLIENT_DATA = FULL
#************************************************
# Enable fast queuing when license queuing is enabled.
#
# Fast queuing allows requests that can be fulfilled immediately to
# be fulfilled. For example, if a client is waiting for two
# licenses, and only one license is immediately available,
# another client that needs only one license can bypass the
# queue and take the single license without waiting.
# Default behavior of license queuing is to put the
# client at the end of the queue regardless whether
# the license request could be satisfied.
#
# Syntax:
# FAST_QUEUE = <feature1, feature2, etc.>
# or
# FAST_QUEUE = ALL
#
#************************************************
# Example:
# FAST_QUEUE = f2, d5, app2
#************************************************
# Allow/deny specific clients from using the license server.
# The allow/deny rules work as follows:
#   - Rules are attempted to be matched in the order they are written
#   - If no rule matches the specific client, then that client is allowed.
#   - For ALLOW_IPADDR_* and DENY_HOST_* rules, you can specify addresses using IPv4
and IPv6.
#    If you are using both protocols, ensure that you have set rules for both of them.
#
# Syntax:
# ALLOW_IPADDR_ALL = <one or more IP addresses>
# ALLOW_IPADDR_<feature name> = <one or more IP addresses>
# (For IPv4 must be either specific A.B.C.D or with wildcards A.*.B.*)
# (For IPv6 must be either specific A:B:C:D:E:F:G:H or with wildcards A:*:C:*:E:*:G:*)
# DENY_IPADDR_ALL = <one or more IP addresses>
```

```
# DENY_IPADDR_<feature name> = <one or more IP addresses>
# (For IPv4 must be either specific A.B.C.D or with wildcards A.*.B.*)
# (For IPv6 must be either specific A:B:C:D:E:F:G:H or with wildcards A:*:C:*:E:*:G:*)
# ALLOW_HOST_ALL = <one or more hostnames or "localhost" for current machine>
# ALLOW_HOST_<feature name> = <one or more hostnames or "localhost" for current
machine>
# DENY_HOST_ALL = <one or more hostnames or "localhost" for current machine>
# DENY_HOST_<feature name> = <one or more hostnames or "localhost" for current machine>
# ALLOW_USER_ALL = <one or more users>
# ALLOW_USER_<feature name> = <one or more users>
# DENY_USER_ALL = <one or more users>
# DENY_USER_<feature name> = <one or more users>
#
#*********************************************
# Example 1:
# ALLOW_IPADDR_ALL = 192.168.1.* 192.168.2.*
# ALLOW_USER_ALL = Administrator root
# DENY_IPADDR_ALL = *.*.*.*
# This will allow only clients on 2 subnets, user Administrator and
# root from any host and deny everyone else. This applies
# to all features.
#
# Example 2:
# DENY_HOST_f2 = localhost untrusted crackerjack
# ALLOW_IPADDR_f2 = 192.168.*.*
# DENY_IPADDR_f2 = *.*.*.*
# This will deny clients on localhost, deny the machines with
# hostname 'untrusted' and 'crackerjack', allow clients on the internal
# network, and deny everyone else. This applies to the feature f2.
#
# Example 3:
# ALLOW_IPADDR_ALL = 2001:0db8:85a3:0000:0000:8a2e:0370:* 2001:0db8:85a3::8a2e:a460:*
1:5567::12c5:*
# DENY_IPADDR_ALL = *:*:*:*:*:*:*:*
# DENY_IPADDR_ALL = *.*.*.*
# This will allow only clients on 3 IPv6 subnets.
# This applies to all features and users.
#*********************************************
# Allow/deny specific clients from borrowing licenses.
```

```
#
# Syntax:
# ALLOW_BORROW_IPADDR_ALL = <one or more hosts>
# ALLOW_BORROW_IPADDR_<feature name> = <one or more hosts>
# (Must be either specific A.B.C.D or with wildcards A.*.B.*)
# DENY_BORROW_IPADDR_ALL = <one or more hosts>
# DENY_BORROW_IPADDR_<feature name> = <one or more hosts>
# (Must be either specific A.B.C.D or with wildcards A.*.B.*)
# ALLOW_BORROW_HOST_ALL = <one or more hosts>
# ALLOW_BORROW_HOST_<feature name> = <one or more hosts>
# DENY_BORROW_HOST_ALL = <one or more hosts>
# DENY_BORROW_HOST_<feature name> = <one or more hosts>
# ALLOW_BORROW_USER_ALL = <one or more users>
# ALLOW_BORROW_USER_<feature name> = <one or more users>
# DENY_BORROW_USER_ALL = <one or more users>
# DENY_BORROW_USER_<feature name> = <one or more users>
#*********************************************
# Example 1:
# ALLOW_BORROW_USER_ALL = daisy harry tom
# DENY_BORROW_HOST_ALL = server1 machine5
# DENY_BORROW_IPADDR_ALL = 192.168.3.* 192.168.4.*
# This will allow the specific users, and deny host and
# IP addresses on the list from borrowing any feature.
# Everyone else will be allowed.
# Example 2:
# ALLOW_BORROW_USER_f2 = lazyjack rabbit joeuser
# DENY_BORROW_IPADDR_f2 = *.*.*.*
# This will allow the specific users and deny everyone
# else from borrowing f2.
#*********************************************
# Limit the number of licenses that can be used by individual users
# or groups to implement fair/desired distribution of licenses.
#
# Syntax:
# LIMIT_USER_<feature name>_<limit count> = <one or more users>
# LIMIT_HOST_<feature name>_<limit count> = <one or more hosts>
# LIMIT_IPADDR_<feature name>_<limit count> = <one or more hosts>
# (Host must be specified completely A.B.C.D or with wildcards A.*.B.*)
```

```
#
# Limiting of users is done by a first match rule, so if a user
# belongs to more than one group specified in restrictions, the first
# restriction will apply to that user.
#************************************************
# Example:
# LIMIT_USER_f2_5 = harry joe sam
# LIMIT_IPADDR_f3_3 = 192.168.2.* 192.168.4.*
#************************************************
# Reserve a number of licenses that can be used by individual users
# or groups to implement fair/desired distribution of licenses.
#
# Syntax:
# RESERVE_USER_<feature name>_<reserve count> = <one or more users>
# RESERVE_HOST_<feature name>_<reserve count> = <one or more hosts>
# RESERVE_IPADDR_<feature name>_<reserve count> = <one or more hosts>
# (Host must be specified completely A.B.C.D or with wildcards A.*.B.*)
#
# Reservation of users is done by a first match rule, so if a user
# belongs to more than one group specified in the rules, the first
# rule will apply to that user.
#************************************************
# Example:
# RESERVE_USER_f2_5 = harry joe sam
# RESERVE_IPADDR_f3_3 = 192.168.2.* 192.168.4.*
#************************************************
# Limit the number of licenses that can be borrowed to prevent
# all licenses from being borrowed at the same time.
#
# Syntax:
# BORROW_LIMIT_COUNT_ALL = <limit count>
# BORROW_LIMIT_COUNT_<feature name> = <limit count>
#************************************************
# Example:
# BORROW_LIMIT_COUNT_f2 = 1
# BORROW_LIMIT_COUNT_ABCDEF = 5
#************************************************
# Limit the number of hours licenses can be borrowed
```

```
# to prevent licenses from being borrowed for too long.
#
# Syntax:
# BORROW_LIMIT_HOURS_ALL = <limit hours>
# BORROW_LIMIT_HOURS_<feature name> = <limit hours>
#***********************************************
# Example:
# BORROW_LIMIT_HOURS_f2 = 1
# BORROW_LIMIT_HOURS_ABCDEF = 5
#***********************************************
# Specify how often to rotate the log file.
# Valid values are "day," "week," or "month."
# The log file rotation occurs at midnight for any of these settings.
# Setting this to any value other than those given above disables log file rotation.
#
# Syntax:
# LOGFILE_ROTATE_INTERVAL = <rotation_interval>
#***********************************************
# Example:
# LOGFILE_ROTATE_INTERVAL = day
```

7.4 Stopping the License Server

On Unix/Linux

1. Log into the license server machine as root.
2. Use systemctl to stop the service. i.e.: systemctl stop altairlmxd

On Windows

1. Right click on ***My Computer*** and select ***Manage***.
2. Click ***Services and Applications***.
3. Click ***Services***.
4. Right click on ***Altair License Server*** and select ***Stop***.

7.5 Borrowing Licenses

Licenses can be borrowed by client machines that are intermittently connected to the network. When the licenses are borrowed, the client machine can be disconnected from the network, and will be able to run the applications for as long as the borrowed license is valid. When the licenses are borrowed from the license server, the licenses remain checked out and unavailable to other clients until the borrowing period expires or the licenses are returned.

7.5a Procedure for Borrowing Licenses

1. Use the command line utility `almutil` or the GUI tool on supported platforms.
2. Open a command or shell window and run the command:

   ```
   almutil –borrow <length in hours>
   ```
3. Run all the applications for which the licenses need to be borrowed. For example, if licenses for HyperMesh and MotionView need to be borrowed, open the applications HyperMesh and MotionView.

   Note: In the above example, even though 21 licenses will be borrowed from the license server, HyperGraph cannot be run on the client with a borrowed license, as the application was not explicitly borrowed.
4. Once you have started all the applications you want to borrow, you must stop the `almutil` command. This is typically done by pressing `<ctrl-c>` in the shell window in which it is running. This prevents additional applications from borrowing licenses from the server.

7.5b Checking the Borrowing Status

To display information about borrowed features, issue the following command from `<installed location>/security/bin/` on the client machine from which the licenses are borrowed.

```
almutil –licstat –borrow
```

The borrowed licenses are displayed at the end of the output.

You may also use the GUI tool on supported platforms.

7.5c Returning Borrowed Licenses Early

To return a borrowed license before the borrowing period expires, reconnect the borrowing system back to the network, and, from the same machine that initialed the borrowing, issue the command below. Make sure that ALTAIR_LICENSE_PATH is set to contain the machine from which the licenses were borrowed.

```
almutil –return
```

7.6 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.

Variable	Description
ALTAIR_LICENSE_PATH	Sets the path to the license file or address of the license server. 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)**
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.**

7.7 Maintenance

For each major release of HyperWorks software, a new license file must be obtained from Altair Engineering. Major releases follow the format #.0 where # represents the version number. For example: 12.0 is a major release.

Periodically, Altair releases a *service pack*. These releases are upgrades of the software and, in general, do not require a new license file.

Replacing an existing license file with a new one

Note: When replacing an existing server license with a new one, the license server process must be shut down.

1. Go to the license server machine.
2. Replace the existing license file in use with the new one or change the setting in the `.cfg` file.
3. Restart the licensing system.

This will not affect running programs. Each client will reconnect to the server once it is restarted. There is an approximate 15-minute window for this take place.

Chapter 8: Usage Logging System

The license server now logs all usage data in a transactional logging system. By default, the system produces only anonymous usage logs used for reporting information back to Altair according to the standard Altair license agreement. This chapter details how the system works and the information available in the logs.

The type of output created by the logging system is controlled by the value of ALUS_LOG_LEVEL in the `alus.conf` file. Values for ALUS_LOG_LEVEL are shown in the table in *Section 8.2, Transactional Log File Contents*.

Note that the transactional log file described in this section differs from the server's normal log file. The server's operational or debug log is output to `stdout` and/or the location designated in the `-lf` argument, or in the config file. It is important to make sure that appropriate file system space is available for all log files. The server's debug log always appends to the old one, it does not overwrite the original. You may move the debug log while the server is running; it will create a new one on the next write operation. The transactional logs should not be moved while the server is running.

To conserve space on the license server, you should routinely cleanup the files in the `data/sent` directory and the `data/detailed` directory.

8.1 Transactional Log Files

Log files are created in the processing directory as set by ALUS_PROCESSING_DIRECTORY in the `alus.conf` file.

A log file name reflects the type of log (detailed or anon) and contains the date and time it was created/started in the format of `alus_log_YYYYMMDDHHMMSS_type.dat` where `type` is either `anon` or `detailed`.

The license server rotates the log files every evening at midnight. When the server opens the new log file, it records all currently open transactions. This allows easier parsing of the data, as anytime there is a check-in record, there should be a corresponding check-out record in the same log file. The timestamps of the open transactions will still be their original timestamps.

When the server rotates the log files, one or both of them are digitally signed, and then moved to the appropriate location. The anonymous logs are moved to ALUS_PENDING_DIRECTORY and the detailed logs are moved to ALUS_DETAILED_DIRECTORY.

8.1a Logging System Settings (in alus.conf file)

Name	Values	Comments
ALUS_LOG_LEVEL	0 1 2	No logging Anonymous logs only Anonymous and detailed logs
ALUS_PENDING_DIRECTORY	Valid directory name	Location of anonymous log files pending transmission to Altair. Log files will remain here until the URT runs.
ALUS_PROCESSING_DIRECTORY	Valid directory name	Location of the current open and in use logs by the server
ALUS_DETAILED_DIRECTORY	Valid directory name	Location of detailed log files that have been closed and rotated by the server

8.2 Transactional Log File Contents

The following documents version 1.0.3 of the transaction log files.

There are three general types of transactions:

Check-out/Check-in/Borrow-out/Borrow-in	Records an actual feature transaction. These are logged anytime a client application attempts to acquire features from the license server.
General Messages	This is used for clients to send messages to be logged.
UserProfiles	This is used to log specific user profiles loaded by client applications such as HyperMesh and HyperCrash.

The log files contain detailed information for each transaction. The two different types of log files (anonymous vs. detailed) differ only in the way private information is represented. They both contain the same records and each record has the same format. For anonymous logs, private information such as username, hostname, and share string have a one-way hash applied to make them unusable to retrieve such information. The hashes are used to process records for matching check-in and check-outs, and leveling. The detailed log file contains readable versions of these values and is not transmitted to Altair. These are to be used for detailed license analysis by the customer.

The log files contain a header section, a data section, and a footer section. The log file contents are described in the tables below. All fields are comma separated.

8.2a Header and footer records

Field Name	Description
Format Vers	Format of this log file in X.Y.Z format (1.0.2)
Version1	Internal Version number 1 of server software
Version2	Internal Version number 2 of server software
Build data	Build date of server software
Build time	Build time of server software
Features	List of features in license file in the format featurename:count,featurename:count… where count is the number of available features
License Id(s)	Altair License ID of all licenses in this license file

Field Name	Description
Lic Platform	Platform that the server is running on
Server Hostname	Hostname of server
Server OS	Operating system running on the license server
Server Vers	Version of the operating system running on the license server
Server Arch	Hardware architecture of the license server
Server Startup	Timestamp of license server startup in the YYYY-MM-DD HH:MM:SS format
Log Initialized	Timestamp that this log file was opened in YYYY-MM-DD HH:MM:SS format
Log Level	Setting of ALUS_LOG_LEVEL in the `alus.conf`
<data>	Beginning of transactional log records
</data>	End of transactional records
Logroll Time	Timestamp of when this log file was closed
Logroll Status	Status of log file closure. 0 = Normal Log Roll 1 = Server Shutdown/restarted 2 = Crash Recovery mode
Signature	The last line of the file is a digital signature.

8.2b Check-out/Check-in/Borrow-out/Borrow-in/Reservation Transaction Details by field Name (appear in this order)

Field name	Description
Timestamp	Timestamp of record in YYYY-MM-DD HH:MM:SS
Record Type	Two letter code representing the type of transaction: CO = Check out CI = Check in BO = Borrow Out BI = Borrow In (return) RO = Reservation RC = Reservation Claimed RE = Reservation early return RN = Reservation normal return
UUID	38 character unique license identifier assigned by Altair
JobHandle	Unique identifier of this session between the license server and the client
Status	Status of transaction. Refer to the error code table in *Appendix B, ALM Error Messages*, for all possible values. Typical values are: LMX_SUCCESS = Success LMX_SOFTLIMIT = usage over soft limit LMX_FEATURE_NOT_FOUND = feature not in license file LMX_NOT_ENOUGH_LICENSES = Not enough features left of requested feature
FeatureName	Name of feature
Feature Checksum	Internal feature checksum
Requested number of features	# of requested by client for this transaction
Actual number of features	The actual number of licenses checked out/in after applicable decay, etc.
Prior features in use	The total number of licenses in use for this feature PRIOR to this transaction

Field name	Description
Version Number	The version number of the feature requested by the client. In the case of a Borrow In, the version number will be that of the license file feature, not necessarily the version the client requested. The matching Borrow Out should contain the clients requested version number.
Share String	This string value is used in determining criteria for leveling and stacking. Value is a one-way hash in the anonymous logs and is unchanged in the detailed logs. See the note after this table for more information.
User Name	User name of user on client machine. Value is a one-way hash in the anonymous logs and is unchanged in the detailed logs.
Host Name	Host name of client machine. Value is a one-way hash in the anonymous logs and is unchanged in the detailed logs.
Project String	Value of ALM_PROJECT set on client. Value is blank in the anonymous logs and is unchanged in the detailed logs.
Borrow ID	Unique identifier for tracking borrowed licenses (only if it is a borrow record)
Borrow Expire	Expire time if this transaction is a borrow out (only if it is a borrow record)
Product String	Name of Product requesting this checkout
Reservation Token	Reservation Token String (ONLY IF A RESERVATION)
Reservation Expire	Reservation Expire Time (Full length of Reservation) (ONLY IF A RESERVATION)
Extended Version Tag	Extended Version information passed from Client
Decay Information	Applicable Decay information. Only if application was subject to decay (CO/CI) otherwise empty field.
Digital Signature	Digital Signature/Checksum of this line

Note: The share string is the key to determining if a checkout was stacked or leveled. If a share string is the same value as any other checkout of the same feature during the same time, those checkouts are leveled against each other.

8.2c General Message and User Profile Transaction Details by Field Name (appear in this order)

Field name	Description
Timestamp	Timestamp of record in YYYY-MM-DD HH:MM:SS
Record Type	Two letter code representing the type of transaction: MG = General Message UP = User Profile
JobHandle	Unique identifier of this session between the license server and the client
Message Contents	String containing message contents.
User Name	Username of user on client machine. Value is a one-way hash in the anonymous logs and is unchanged in the detailed logs
Host Name	Hostname of client machine. Value is a one-way hash in the anonymous logs and is unchanged in the detailed logs.
Digital Signature	Digital Signature/checksum of this line

Appendix A: License File Format

This appendix describes the basic format of the license file. The difference between a server based license and a node locked license is that in a server based license, each feature has a COUNT field. For more details, refer to the LM-X End User Guide. The example shows one typical FEATURE block. There are many of these in a license file.

```
FEATURE GlobalZoneAM
{
VENDOR = ALTAIR
COUNT = 10000000
KEYTYPE = EXCLUSIVE
VERSION = 11.0
END = 2010-12-31
BORROW = 168
SOFTLIMIT = 5000000
SHARE = CUSTOM|VIRTUAL
COMMENT =          "HOSTID=005056BC49E1"
SN =      "22c1c559-a926-4959-b00c-a909039f035001"
KEY =     OLKE7SHR7FF86PJT23FV52L6F6IELJ3NLR7ONR9ANEO4AQ8N2T2NDVL1RLTSBO8LF2A77O \
        55O00RPH5K5FNENKUKC84T1NPTJKTK3H2QNF5AOQJ7P84CMPQJ1OME6TGGAU2PBTFH2BH3 \
          F6F2DUHNBPEUUP9EU403M8ETF5NTQL6NO5OQLNJK0SL202OD8FO0KLOJUUJIA2Q1N2427Q \
          TQS749P2MT11I76LHEFA2OI89V8N8G0U5JAJ6T63U7GNSOETTGJ1G9E2MJ28KIFG2URNA7 \
          45I3RO0BO73MUB30K8SLMD8CHFO0B8FQD1JSBVVN1C2LUI94OFGIDUMB7IAQQUMP6RI2OC \
          SLLSNAM9B8662JD00O3USSSFN5QJTADG67S3SI32RB2FGM8KP8NSSG93RIHI6VCS6VHG9L \
          VB7LHM93Q3GMM9FTBC0C1FO122B05PENHLUDMCFRNLHV9761VHPDUVB9O1BUNCVDFCHCPF \
          J07PTOF9S98PK8GDAR4MF2DNIUQVA4IEBKIUUIRPU81MBI1AHOB8P73U71MHD4C9MIC81F \
          IHLTI5BVPDDPA828QQ6OS9OKAEQO1Q7C78MO3J9GBM6EAS4QPA178HTN6LPH5AMNKEHRI8 \
          0HUJLDGLO26PDSQJHVDNVUUDBHVE3BCI0
}
```

Appendix B: ALM Error Messages

The table below lists the error codes and strings from the license server and their meanings.

#	Error String	Description
0	LMX_SUCCESS	Operation successful
1	LMX_UNKNOWN_ERROR	Unknown error occurred
2	LMX_INVALID_PARAMETER	Invalid input parameter
3	LMX_NO_NETWORK	Unable to initialize network subsystem
4	LMX_BAD_LICFILE	License file is using unknown/invalid syntax
5	LMX_NO_MEMORY	No more available memory
6	LMX_FILE_READ_ERROR	Unable to read file
7	LMX_BAD_DATE	Invalid date
8	LMX_BAD_KEY	Invalid license key
9	LMX_FEATURE_NOT_FOUND	Feature not found
10	LMX_BAD_HOSTID	Hostid does not match license
11	LMX_TOO_EARLY_DATE	Software activation date is not yet reached
12	LMX_TOO_LATE_DATE	Software expired
13	LMX_BAD_VERSION	Software version does not match license
14	LMX_NETWORK_ERROR	Unexpected network-related error occurred
15	LMX_NO_NETWORK_HOST	Unable to use license server
16	LMX_NETWORK_DENY	Rejected from license server
17	LMX_NOT_ENOUGH_LICENSES	Request for more licenses than available on license server
18	LMX_BAD_SYSTEMCLOCK	System clock has been set back

#	Error String	Description
19	LMX_TS_DENY	Feature not allowed to run on remote desktop (terminal server) clients
20	LMX_VIRTUAL_DENY	Software not allowed to run on a virtual machine
21	LMX_BORROW_TOO_LONG	The specified borrow period is too long
22	LMX_FILE_SAVE_ERROR	Unable to save file
23	LMX_ALREADY_BORROWED	Feature already borrowed
24	LMX_BORROW_RETURN_ERROR	Unable to return borrowed feature
25	LMX_SERVER_BORROW_ERROR	License server returned borrow error
26	LMX_BORROW_NOT_ENABLED	Borrow functionality not enabled on client side
27	LMX_NOT_BORROWED	The feature that was attempted to be returned was not borrowed
28	LMX_DONGLE_ERROR	Dongle is not attached or does not function correctly
29	LMX_SOFTLIMIT	Request exceeds the number of softlimit licenses available
30	LMX_BAD_PLATFORM	Platform not permitted by license
32	LMX_TOKEN_LOOP	Infinite token loop detected
33	LMX_BLACKLIST	Feature is blacklisted
34	LMX_VENDOR_DENY	Feature checkout rejected by vendor-defined rules
35	LMX_NOT_NETWORK_FEATURE	Unable to use network license as a local license
36	LMX_BAD_TIMEZONE	Checkout is not permitted in the client time zone
37	LMX_SERVER_NOT_IN_USE	License server is not currently in use
38	LMX_LICSERVICE_ERROR	Problem with License Distribution Service
39	LMX_USER_NOT_ROOT	User does not have superuser privileges
40	LMX_NOT_IMPLEMENTED	Functionality not implemented

#	Error String	Description
41	LMX_BORROW_LIMIT_EXCEEDED	License server borrow limited exceeded
42	LMX_SERVER_FUNC_ERROR	License server function error
43	LMX_HEARTBEAT_LOST_LICENSE	License is lost due to heartbeat failure
44	LMX_SINGLE_LOCK	Unable to obtain single lock
45	LMX_AUTH_ERROR	Cannot authenticate user on license server
46	LMX_NETWORK_SEND_ERROR	Error sending network message
47	LMX_NETWORK_RECIEVE_ERROR	Error receiving network message
48	LMX_QUEUE	Feature has been queued
49	LMX_BAD_SECURITY_CONFIG	LM-X security configuration file mismatch
50	LMX_FEATURE_HAL_MISMATCH	Feature has different HAL settings than others
51	LMX_NOT_LOCAL_FEATURE	Unable to use network license as local
52	LMX_FEATURE_NOT_REPLACEABLE	Unable to replace missing feature
53	LMX_HOSTID_NOT_AVAILABLE	HostID is not available on the current machine
54	LMX_FEATURE_ALREADY_RESERVED	Feature is already reserved
55	LMX_FEATURE_ALREADY_CHECKED_OUT	Feature is already checked out
56	LMX_RESERVATION_NOT_FOUND	Reservation not found
57	LMX_API_NOT_REENTRANT	An LM-X API function was called from a callback
58	LMX_LICENSE_UPLOAD_ERROR	Error with license file upload

Appendix C: Example of alus.conf 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=/usr/local/altair/licensing13.0/data/pending
ALUS_PROCESSING_DIRECTORY=/usr/local/altair/licensing13.0/data/processing
ALUS_SCRATCH_DIRECTORY=/usr/local/altair/licensing13.0/data/scratch
ALUS_DETAILED_DIRECTORY=/usr/local/altair/licensing13.0/data/detailed
ALUS_LOGGER_DIRECTORY=/usr/local/altair/licensing13.0/data/logs
ALUS_MANUAL_DIRECTORY=/usr/local/altair/licensing13.0/data/manual
ALUS_SENT_DIRECTORY=/usr/local/altair/licensing13.0/data/sent
ALUS_INVALID_DIRECTORY=/usr/local/altair/licensing13.0/data/invalid
ALUS_UNDELIVERABLE_DIRECTORY=/usr/local/altair/licensing13.0/data/undeliverable
ALUS_UNDELIVERABLE_DB_FILE_PATH=/usr/local/altair/licensing13.0/data/undeliverable.prop
erties
#*****************************************************************
#
#*****************************************************************
## Location of Decay Directory
#
ALUS_DECAY_DIRECTORY=/usr/local/altair/licensing13.0/decay
#*****************************************************************
```

```
#
#*************************************************************************
## Location of 3rd party license files for Catia Readers
#
#TTF_LICENSE_FILE_PATH=/usr/local/altair/licensing13.0/ttfLicense.txt
#
#*************************************************************************
#
#*************************************************************************
## Setting for Altair Logging level of transactional data
#
## 0 = No logging
## 1 = Anonymized only
## 2 = Anonymized and Detailed
#
ALUS_LOG_LEVEL=2
#*************************************************************************
#
#*************************************************************************
## Location of script to run reporting tool
#
ALUS_URT_LOCATION=/usr/local/altair/licensing13.0/bin/urt.sh
#*************************************************************************
#
#*************************************************************************
## Number of attempts to transmit data each week
#
ALUS_MAX_ATTEMPTS=2
#*************************************************************************
#
#*************************************************************************
## Address of the Altair Usage Collection Server for HTTP
## **Only one pair of the addresses below should be uncommented.**
#
ALUS_ALAS_URL=http://urtserv01.altair.com/NewURT
ALUS_WSDL_URL=http://urtserv01.altair.com/NewURT/services/AppServer
#
## Address of the Altair Usage Collection Server for HTTPS
```

```
#
#ALUS_ALAS_URL=https://urtserv01.altair.com/NewURT
#ALUS_WSDL_URL=https://urtserv01.altair.com/NewURT/services/AppServer
#
#*************************************************************************
#
#*************************************************************************
## Configuration of proxy should be either HTTPS(SSL) or HTTP.
## Please uncomment the settings below to match the address chosen above
## choose either ALUS_HTTP_ or ALUS_HTTPS_ to match http or https from above
## settings of ALUS_ALAS_URL and ALUS_WSDL_URL
#
## HTTP Proxy setup (choose this one or the HTTPS below)
## user and password should not be set if they are not required
## by your proxy server.
#
#ALUS_HTTP_PROXYHOST=
#ALUS_HTTP_PROXYPORT=
#ALUS_HTTP_PROXY_USER=username
#ALUS_HTTP_PROXY_PASSWORD=password
#
## HTTPS(SSL) Proxy setup (choose this one or the HTTP one above)
## user and password should not be set if they are not required by your proxy server
#
#ALUS_HTTPS_PROXYHOST=
#ALUS_HTTPS_PROXYPORT=
#ALUS_HTTPS_PROXY_USER=username
#ALUS_HTTPS_PROXY_PASSWORD=password
#*************************************************************************
#
#
#*************************************************************************************
## Optional email address for usage report confirmation
## uncomment and set to a single email address to recieve confirmation
## from the Altair server after report delivery
## PLEASE NOTE THAT THIS ADDRESS IS SENT TO THE ALTAIR SERVER EACH
##TIME THE URT SENDS DATA.  IT IS NOT RETAINED FOR FUTURE USAGE.
#
```

```
#REPORT_CONFIRMATION_EMAIL=user@domain.com
#*************************************************************************
```

Appendix D: Uninstall Process

Uninstalling from Windows

1. Stop the License Manager.
2. Open the **Windows Control Panel** and select the ***add/remove*** or ***Programs*** control panel icon.
3. Locate and highlight the Altair Licensing <Version> Package and click the ***remove*** button.
4. Click ***Finish*** to complete the process.

Uninstalling from Unix/Linux

1. Stop the license manager.
2. cd to `<install_location>` in a terminal and run `./run_uninstall`.

Appendix E: Sending Reports Manually

Realizing that not every customer wants a completely automated system, Altair provides two methods to manually send license usage reports. Method 1 takes the automated sending out of the loop, and allows the customer to start the URT by hand. It still utilizes the totally automated system for collecting, zipping, communicating, and moving the logs. Method 2 is completely manual, requiring some work on the administrator's part to gather logs, etc. and deliver them manually via email or web upload to Altair.

The current, open log file is in the processing directory. All closed, finished logs are stored in pending, and successfully sent zip files are kept in sent. The directory bin contains the file that is used to start the URT (`urt.bat` or `urt.sh`). All log files are plain text and digitally signed to prevent tampering.

The current URT utilizes a Java program (`lib/altairURT.jar`) to send the data from the host to Altair. The configuration file for the program is `alus.conf`. This is where things such as proxy settings and HTTP vs. HTTPS are configured.

In the end, the license server simply executes the file `bin/urt.bat` (or `urt.sh`) once a week. This is a simple script that launches the URT. It is the easiest way of knowing that information is being sent to the correct location during the correct time period.

Method 1: Manual Usage of Existing URT Software

All that is required to implement this method is to copy/backup the `urt.sh` or `urt.bat` file to another name, such as `urt_original.bat` or `urt_original.sh` and change the original file to be empty. The file resides in `bin/urt.bat`.

For Linux/Unix:

```
cp urt.sh urt_original.sh
```

For Windows:

copy `urt.bat urt_original.bat`

Then, edit the `urt.bat` or `urt.sh` and remove all lines, leaving the file completely empty. When the license server 'starts' the URT, nothing happens and the UMT continues normally.

When you are ready to send your data, go to the bin directory and use `urt_original script`. Simply type `./urt_original.sh` or `urt_original.bat`. This processes all the logs and sends the data. During this process, detailed output provides information on what is happening. If you would like to examine all the logs before they are sent, they are located in the `pending` directory.

Method 2: Completely Manual Processing and Email/upload of Data

This method is a completely manual way to collect the log files and sending them to Altair for processing. The administrator should inspect the files (if desired), then use the provided utilities to process the files and prepare them to be emailed to Altair.

First, follow the steps in Method 1 to move the `urt.bat` or `urt.sh` file aside and have an empty script in its place.

Assuming you copied the original file to `urt_original.bat` you will need to edit that file. Simply invoke the old script with the `-M` option. This option prepares the files to be sent and places the final file in the directory pointed to in the `alus.conf` file (ALUS_MANUAL_DIRECTORY, by default `…/data/manual`). Running `./urt_original.bat -M` creates a file(s) in the manual directory with a name like `YYYYMMDD######_#.zip`. For customers that choose manual reporting an upload link will be sent to you monthly via a reminder email. Please follow this link to upload your data. Alternatively, you may send directly via email (as an attachment) to usagereporting@altair.com.

Please contact your Altair support organization if you have any questions.

Appendix F: LM-X Reference for FLEXlm/FLEXnet Users

This section provides some basic cross reference information for customers familiar with FLEXlm.

F.1 License File

FLEXlm and LM-X license files are very similar; they are both plain text files containing feature blocks. Unlike FLEXlm, you should not need to edit the license file that you receive from Altair.

F.2 License Paths

License paths describe where licenses are found. Both systems use environment variables to provide applications with the appropriate values. Both require different separators on Windows and Unix/Linux; they are the same for both systems: a list separated by ";" (semicolon) on Windows and ":" (colon) on Unix/Linux.

FLEXlm Values	LM-X Values
ALTAIR_LM_LICENSE_FILE =port@host	ALTAIR_LICENSE_PATH=port@host
LM_LICENSE_FILE=port@host	LMX_LICENSE_PATH=port@host

F.3 Useful Commands

FLEXlm	LM-X
lmutil lmstat –c port@host –a	almutil –licstat –host hostname –port port#
lmutil lmhostid	almutil -hostid
Lmutil lmreread	`almutil –restartserver –host <hostname> –port <port#> -password <pwd>` (Note: admin password must be set in the `.cfg` file for this to function)
lmutil lmremove	`almutil -removeuser -clientusername <username> -clienthostname <client's host> -host <server's host> -port <port> –password <password>`