

Technical Document

Niagara^{AX-3.x} Platform Guide

Updated: December 11, 2007



Niagara^{AX} Platform Guide

Copyright © 2007 Tridium, Inc.

All rights reserved.

3951 Westerre Pkwy, Suite 350

Richmond

Virginia

23233

U.S.A.

Copyright Notice

The software described herein is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior written consent from Tridium, Inc.

The confidential information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and is not to be released to, or reproduced for, anyone else; neither is it to be used for reproduction of this Control System or any of its components.

All rights to revise designs described herein are reserved. While every effort has been made to assure the accuracy of this document, Tridium shall not be held responsible for damages, including consequential damages, arising from the application of the information contained herein. Information and specifications published here are current as of the date of this publication and are subject to change without notice.

The release and technology contained herein may be protected by one or more U.S. patents, foreign patents, or pending applications.

Trademark Notices

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft and Windows are registered trademarks, and Windows NT, Windows 2000, Windows XP Professional, and Internet Explorer are trademarks of Microsoft Corporation. Java and other Java-based names are trademarks of Sun Microsystems Inc. and refer to Sun's family of Java-branded technologies. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, Niagara^{AX} and Vykon are registered trademarks, and Workbench, WorkPlace^{AX}, and ^{AX}Supervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners. The software described herein is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

CONTENTS

Preface	vii
Document Change Log	vii
Niagara platform.....	1-1
Platform overview	1-1
About a platform connection	1-2
Platform daemon on PC	1-2
Provisioning versus platform interface	1-2
Types of platform views	1-3
About platform differences	1-4
QNX-based	1-4
Dialup Configuration	1-4
Platform Administration	1-4
Win32-based	1-5
Dialup Configuration	1-5
Platform Administration	1-6
Models of platforms	1-6
Application Director	1-7
Installed applications (stations)	1-8
Station selections	1-8
Station output	1-9
Station output overview	1-9
Station log levels (spy:/logSetup)	1-10
Station LogHistory (LogHistoryService)	1-10
Station and output controls	1-11
Start checkboxes	1-11
Station control buttons	1-11
Output control buttons	1-12
Output settings	1-13
Ddns Configuration	1-13
DDNS core configuration items	1-13
Provider	1-13
Mode	1-14
Adapter	1-14
About TZO	1-14
Dialup Configuration	1-14
Platform dialup configuration sections	1-15
Modem section	1-15
Listener section	1-16
Captive Network section	1-17
About the dialup daemon and service	1-17
Dialup daemon	1-17
DialupService	1-18
About dialup operation	1-19
Dialup addresses	1-19
Calling from Workbench	1-21
Distribution File Installer	1-21
Dist file selection	1-22

Downgrading a JACE (Clean Dist)	1-23
<i>AX-3.3 dependency check change</i>	1-24
Restoring a backup dist	1-25
Upgrading a JACE	1-26
<i>Commissioning Wizard method (recommended to upgrade JACE)</i>	1-26
<i>Distribution File Installer method (if you must)</i>	1-27
Module updating from Distribution File Installer	1-27
Distribution file install process	1-28
File Transfer Client	1-28
Lexicon Installer	1-29
License Manager	1-30
License operations (AX-3.3 and later)	1-31
<i>Import</i>	1-31
<i>Export</i>	1-32
<i>License Import results</i>	1-33
License Install File notes (pre-AX-3.3)	1-33
<i>Other license operations (pre-AX-3.3)</i>	1-34
About the licensing server	1-35
Software Manager	1-36
About your software database	1-37
Default software listing and layout	1-38
<i>Software Manager table columns</i>	1-39
<i>Software Details</i>	1-40
Filtering displayed software	1-40
<i>Filter by status</i>	1-40
<i>Filter by name</i>	1-41
Software Import	1-41
<i>Import vs. copy into modules</i>	1-41
Software actions	1-41
<i>Upgrade All Out of Date</i>	1-42
<i>Install</i>	1-42
<i>Uninstall</i>	1-42
<i>Re-Install, Upgrade, Downgrade</i>	1-43
<i>Commit and Reset</i>	1-43
Platform Administration	1-43
Types of Platform Administration functions	1-44
View Details	1-44
Update Authentication	1-45
<i>Digest platform authentication</i>	1-45
<i>Basic platform authentication</i>	1-46
Change HTTP Port	1-47
Change Date/Time	1-48
<i>Use Local</i>	1-49
FTP/Telnet	1-49
Change Output Settings	1-49
<i>Log filter settings</i>	1-50
View Daemon Output	1-50
Set Module Filter	1-50
<i>Operations from a change in module content level</i>	1-51
Backup	1-52
Commissioning	1-53
Reboot	1-53
Station Copier	1-54
Station copy direction	1-54
Station Transfer Wizard	1-55
<i>Name step</i>	1-55
<i>Delete step</i>	1-55
<i>Content step</i>	1-56
<i>Disposition step</i>	1-56
<i>Station settings step</i>	1-56
<i>Details step</i>	1-57

Modules step	1-57
Stop station step	1-57
Review step	1-58
Transferring station	1-58
Deleting stations	1-59
TCP/IP Configuration	1-59
TCP/IP Host fields	1-60
TCP/IP DNS fields	1-61
TCP/IP Interface fields	1-61
User Manager	1-62
Users management	1-63
Review user	1-63
Add user	1-63
Delete user	1-63
Change user password	1-63
Drag and drop	1-64
Groups management	1-64
Review group	1-64
Add group	1-64
Delete group	1-65
Add member	1-65
Remove member	1-65
Remote File System	1-65
About Platform Services	1-66
PlatformServiceContainer parameters	1-67
PlatformServiceContainer status values	1-67
PlatformServiceContainer configuration parameters	1-68
Additional PlatformServiceContainer properties (AX-3.2 and later)	1-69
PlatformServiceContainer actions	1-70
SystemService (under PlatformServices)	1-71
Platform service types	1-72
Using platform services in a station	1-72
JACE power monitoring	1-72
JACE-NXS power monitoring	1-73
JACE-NX hardware monitoring	1-73
JACE-NXS hardware monitoring	1-73
PlatformServices binding and link caveats	1-73

Platform Component Guides 2-1

Component Reference Summary	2-1
Components in platDialup module	2-1
platDialup-DialupPlatformService	2-1
Components in platform module	2-1
platform-DaemonFileSpace	2-1
platform-DaemonSession	2-1
platform-LicensePlatformService	2-2
platform-Platform	2-2
platform-PlatformAlarmSupport	2-2
platform-PlatformServiceContainer	2-2
platform-SystemPlatformServiceQnx	2-2
Reboot action	2-2
platform-SystemPlatformServiceWin32	2-2
Reboot action	2-2
platform-TcplpPlatformService	2-2
Components in platLon module	2-3
platLon-LonPlatformServiceQnx	2-3
platLon-LonPlatformServiceWin32	2-3
Components in platPower module	2-3
platPower-JaceSlaBattery	2-3

platPower-Npm2NimhBattery	2-3
platPower-NpmDualBatteryPlatformService	2-3
platPower-NpmExternalSlaBattery	2-3
platPower-PowerMonitorPlatformServiceQnx	2-3
Components in platPowerNxs module	2-3
platPowerNxs-PowerMonitorPlatformServiceNxsWin32	2-3
Components in platSerialQnx module	2-4
platSerialQnx-SerialPortPlatformServiceQnx	2-4
platSerialQnx-SerialPortPlatformServiceQnx403	2-4
platSerialQnx-SerialPortPlatformServiceQnx404	2-4
Components in platSerialWin32 module	2-4
platSerialWin32-SerialPortPlatformServiceWin32	2-4
Components in platSysmonNx module	2-4
platSysmonNx-HardwareMonitorNxPlatformServiceWin32	2-4
Components in platSysmonNxs module	2-4
platSysmonNxs-HardwareMonitorNxPlatformServiceWin32	2-4

Platform Plugin Guides 3-1

Plugin Reference Summary	3-1
Plugins in platDaemon module	3-1
platDaemon-ApplicationDirector	3-1
platDaemon-DaemonLogSettingsView	3-1
platDaemon-DistInstaller	3-1
platDaemon-DistributionView	3-2
platDaemon-FileTransferClient	3-2
platDaemon-LexiconInstaller	3-2
platDaemon-LicenseManager	3-2
platDaemon-SoftwareManager	3-2
platDaemon-SoftwareView	3-2
platDaemon-PlatformAdministration	3-2
platDaemon-PlatformTextSummaryEditor	3-2
platDaemon-StationCopier	3-2
platDaemon-StationDirector	3-2
platDaemon-StationTextSummaryEditor	3-2
platDaemon-TcplpConfiguration	3-2
platDaemon-UserManager	3-3
Plugin in platDDNS	3-3
platDdns-DdnsConfigurationView	3-3
Plugins in platDialup	3-3
platDialup-ConfigureDialup	3-3
platDialup-DialupEditor	3-3
Plugins in platform module	3-3
platform-ActivityView	3-3
platform-LicensePlatformServicePlugin	3-3
platform-PlatformServiceContainerPlugin	3-3
platform-PlatformServiceProperties	3-3
platform-SystemPlatformServicePlugin	3-3
Reboot	3-3
platform-SystemPlatformServiceQnxPlugin	3-4
platform-TcplpPlatformServicePlugin	3-4

License Tools and Files. A-1

Workbench License Manager	A-2
Import File	A-2
Export File	A-3

Delete	A-4
Sync Online	A-4
Request License	A-5
About the local license database	A-6
Local license database rationale	A-6
Local license inbox	A-6
About license archive (.lar) files	A-7
Licensing server use of license archives	A-7
About NiagaraAX license files	A-7
Items common to all license files	A-8
<i>license</i>	A-8
<i>about</i>	A-8
<i>brand</i>	A-8
<i>signature</i>	A-9
JACE hardware features	A-9
<i>maxheap</i>	A-9
<i>mstp</i>	A-9
<i>ndio</i>	A-9
<i>serial</i>	A-9
Driver attributes	A-9
<i>name</i>	A-10
<i>expiration</i>	A-10
<i>device.limit</i>	A-10
<i>history.limit</i>	A-10
<i>point.limit</i>	A-10
<i>schedule.limit</i>	A-10
<i>parts</i>	A-10
Driver types	A-10
<i>aaphp</i>	A-10
<i>aapup</i>	A-10
<i>bacnet</i>	A-10
<i>bacnetws</i>	A-11
<i>dust</i>	A-11
<i>fileDriver</i>	A-11
<i>lonworks</i>	A-11
<i>modbusAsync</i>	A-11
<i>modbusSlave</i>	A-11
<i>modbusTcp</i>	A-11
<i>modbusTcpSlave</i>	A-11
<i>obixDriver</i>	A-11
<i>opc</i>	A-11
<i>niagaraDriver</i>	A-11
<i>rdbDb2</i>	A-11
<i>rdbOracle</i>	A-11
<i>rdbSqlServer</i>	A-12
<i>snmp</i>	A-12
Applications	A-12
<i>station</i>	A-12
<i>web</i>	A-12
<i>workbench</i>	A-12
<i>crypto</i>	A-12
<i>eas</i>	A-13
<i>email</i>	A-13
<i>genericAppliance</i>	A-13
<i>provisioning</i>	A-13
<i>tenantBilling</i>	A-13

Time Zones and NiagaraAX B-1

Time zones and terminology	B-1
UTC	B-1
DST	B-1
Selecting a time zone in NiagaraAX	B-2

About timezones.xml	B-2
DST start and end syntax variations	B-3
Using Workbench to edit and transfer timezones.xml	B-3
About the historical time zone database (AX-3.3 and later)	B-4
Issue example and fix	B-4
Parts of the historical time zone database	B-4
Updating a historical time zone database	B-4

CONTENTS

Preface

- [Document Change Log](#)

Document Change Log

Updates (changes/additions) to this *NiagaraAX Platform Guide* document are listed as follows.

- Updated: December 11, 2007
In the main “[Niagara platform](#)” section, numerous changes were made related to AX-3.3, including references to provisioning in “[Provisioning versus platform interface](#)”, name change for the Station Director to Application Director in sections “[Types of platform views](#)” and “[Application Director](#)”, and JACE-6 mention in the section “[Models of platforms](#)”. The “[Distribution File Installer](#)” section was reworked to de-emphasize using it for upgrading a JACE, but instead as means to restore a back-up dist, and also “clean” dists. See related subsections “[Downgrading a JACE \(Clean Dist\)](#)” and “[AX-3.3 dependency check change](#)”. The “[License Manager](#)” section was also reworked to describe its different operation in AX-3.3 vs. AX-3.2 or AX-3.1. A new note about static TCP/IP addressing for JACEs with two LAN ports was also added in the section “[TCP/IP Interface fields](#)”. A new section “[SystemService \(under PlatformServices\)](#)” was added in “[About Platform Services](#)” section. Corresponding to the new license management features, an appendix “[License Tools and Files](#)” was added, which includes sections on the “[Workbench License Manager](#)”, local license database and [license archive \(.jar\) files](#). The appendix also includes an “[About NiagaraAX license files](#)” section, to describe license feature entries. Another new appendix was also added: “[Time Zones and NiagaraAX](#)”, which includes sections “[Time zones and terminology](#)”, “[Selecting a time zone in NiagaraAX](#)”, “[About timezones.xml](#)”, and “[About the historical time zone database \(AX-3.3 and later\)](#)”.
- Updated: April 30, 2007
Added new subsection “[Additional PlatformServiceContainer properties \(AX-3.2 and later\)](#)” describing new resource-related properties for a QNX-based platform.
- Updated: January 15, 2007
Applied “new look” formatting to print (PDF) version of this document, reducing total page count yet adding “sidehead” white space to most pages (useful for note taking on printed pages). Few content changes from previous revision, apart from new “refer-to” note in “[Ddns Configuration](#)” section.
- Revised: November 14, 2006
Revised the descriptions and screen captures in the “[License Manager](#)” section to better describe on-line functions with the licensing server. Added new “[PlatformServices binding and link caveats](#)” section to describe known limitations. Made several other minor edits.
- Revised: July 7, 2006
Removed platform “part” components and the parent “RemoteDaemonPlatform” (Platform Snapshot) component from the “[Platform Component Guides](#)” section—these provisioning-related items were eliminated.
- Revised: June 22, 2006
Updated for changes in NiagaraAX-3.1 (typically noted as “AX-3.1”), and reversed the order of this change log to list most recent document versions at top. New sections include “[Ddns Configuration](#)”, “[Provisioning versus platform interface](#)”, and “[Models of platforms](#)”. Various other changes/additions were also made in the sections “[About platform differences](#)”, “[About Platform Services](#)” and “[Platform Component Guides](#)”, largely to describe new properties and components, often related to new platform types.
- Revised: October 31, 2005
Edited various sections in “[About Platform Services](#)”, including new section “[PlatformServiceContainer actions](#)” and various items related to [JACE-NX hardware monitoring](#).

- Revised: September 7, 2005
Added more details in “[Set Module Filter](#)” section “[Operations from a change in module content level](#)”.
- Revised: August 26, 2005
Added section “[Import vs. copy into modules](#)”, under [Software Manager](#) section “[About your software database](#)”, which explains options when copying newly-received software modules on a Workbench PC. Updated screenshots and property descriptions in all sections under the “[Dialup Configuration](#)” section, to reflect dialup changes since original document.
- Revised: August 12, 2005
Made changes to the “[Distribution File Installer](#)” section “[Restoring a backup dist](#)”, noting recent addition of dialog about restoring TCP/IP settings (see [Figure 1-33](#) on page 26). Also added brief note about this in the “[Backup](#)” section under “[Platform Administration](#)” section.
- Revised: August 2, 2005
Made a minor change to the “[Platform Administration](#)” section “[Change Date/Time](#)”, noting different popup dialog in rare case if Win32 host does not support a Java time zone (see [Figure 1-75](#) on page 49). Moved copyright and trademark information to front of document and used new (PDF) cover design.
- Revised: July 12, 2005
Made numerous edits/corrections and updated screenshots in the main “[Niagara platform](#)” section, following a content review. Removed Bajadoc references in “[Platform Component Guides](#)”.
- Publication: June 24, 2005
Changed numerous areas in the “[Niagara platform](#)” section, with most changes in the “[Distribution File Installer](#)” section (“[Dist file selection](#)”, “[Upgrading a JACE](#)”, “[Restoring a backup dist](#)”) and “[Software Manager](#)” section (“[About your software database](#)”, “[Software Import](#)”, “[Software actions](#)”). Also made changes in the “[Platform Administration](#)” section related to the new [Backup](#) command, and in various “[Application Director](#)” sections related to QNX-based JACEs (“[Start checkboxes](#)”, “[Station control buttons](#)”). Also made various changes in the “[About Platform Services](#)” section, mostly in the “[JACE power monitoring](#)” section. Added legal info block in this preface section.
- Draft: June 15, 2005
Globally, the name of Module Manager changed to [Software Manager](#). An upcoming document revision will describe new/changed functionality in this view, as well as in other views.
Added several missing entries in the “[Platform Component Guides](#)” section.
- Draft: May 25, 2005
(Initial change log) Same as previous revision.

CHAPTER 1

Niagara platform

Platform is the name for everything that is installed on a Niagara host that is not part of a Niagara station. The platform interface provides a way to address all the support tasks that allow you to setup and support and troubleshoot a Niagara host.

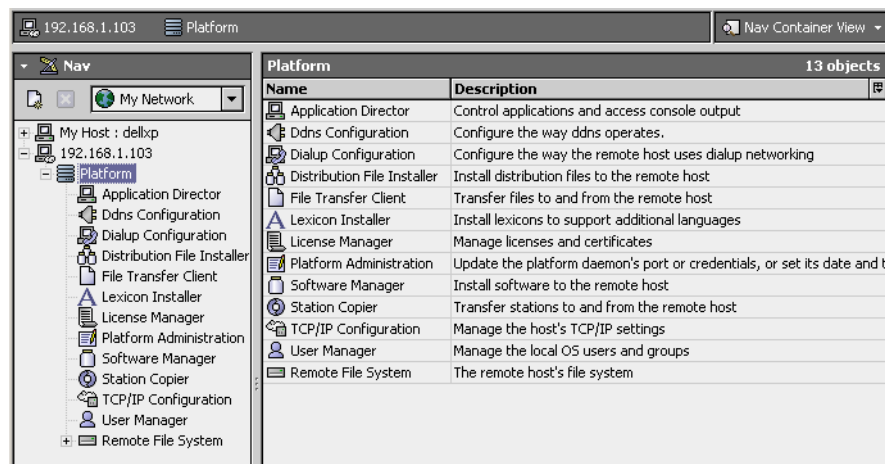
The following main sections provide more platform details:

- [Platform overview](#)
- [About platform differences](#)
- [Application Director](#) (formerly **Station Director**, see [Note](#): on page 7)
- [Ddns Configuration](#)
- [Dialup Configuration](#)
- [Distribution File Installer](#)
- [File Transfer Client](#)
- [Lexicon Installer](#)
- [License Manager](#)
- [Software Manager](#)
- [Platform Administration](#)
- [Station Copier](#)
- [TCP/IP Configuration](#)
- [User Manager](#)
- [Remote File System](#)
- [About Platform Services](#)

Platform overview

In Workbench, when you open a platform connection to a Niagara host (whether JACE or AxB Supervisor), that host's available platform functions are listed in the platform's Nav Container View, see [Figure 1-1](#).

Figure 1-1 Platform functions listed in platform's Nav Container View



Each function has its own Workbench view (plugin); you access it by simply double-clicking. The same platform views exist whether you are connected to a JACE or a AxB Supervisor, with exceptions noted below.

Note: For any *Win32-based* platform, a “*User Manager*” view is available. This view is not available if the platform is a *QNX-based* JACE. Also, a few views differ depending on platform type. See “*About platform differences*” on page 1-4 for details.

Also, if working at an AX-3.1 or later Supervisor PC (or engineering workstation), and you open a local platform connection, notice that some platform views are “missing,” namely: *Distribution File Installer*, *File Transfer Client*, and *Station Copier*. The views have no real application when working at your local PC host—instead, you simply use Windows Explorer.

The following sections provide additional background:

- [About a platform connection](#)
- [Provisioning versus platform interface](#)
- [Types of platform views](#)
- [About platform differences](#)

About a platform connection

A platform connection is different than a station connection. When connected to a [Niagara platform](#), Workbench communicates (as a client) to that host’s *platform daemon* (also known as “niagarad” for Niagara daemon), a *server process*. Unlike a station connection, which uses the Fox protocol, a client platform connection requires Workbench, meaning it is *unavailable* using a standard Web browser.¹

The platform daemon is a compact executable written in native code, meaning the daemon does not require the Niagara core runtime, or even a Java virtual machine (JVM). The platform daemon is pre-installed on every JACE controller (even as factory-shipped), and runs whenever the JACE has booted up.

Also, a Niagara host’s platform daemon monitors a different TCP/IP port for client connections than does any running station (if any). By default, this is port 3011. Finally, as a platform client, you sign on using “host level” credentials for authentication. This means a user account and password separate from any station user account, and should be considered the *highest level access to that host*.

Platform daemon on PC

When you install NiagaraAX on your PC, one of the last “Would you like to?” install options is:

Install and Start Platform Daemon

The default selection is to install. You need the platform daemon locally installed and running in order for any of the following:

- To host a Niagara station on *your local PC*, such as for an AxSupervisor. This lets you open a Workbench client platform connection to your local (“My Host”) platform. It also allow remotes *remote* client platform connections to your PC as well.
- To use Workbench to open a remote JACE platform (or station) using a *dialup connection*, that is, using your PC’s modem. (The platform daemon is not used to open a LAN-connected platform).
- For a PC to run as an AX SoftJACE, essentially a JACE running on a PC dedicated to this application. (Note that the AX SoftJACE Install wizard *automatically* installs and starts the platform daemon. Also, in this particular case, Workbench is *not* available to run locally at that PC.)

Once installed and started on a PC, you can see the platform daemon listed as a *Niagara service* from the Windows Control Panel, by selecting **Administrative Tools > Services**.

Note: Following NiagaraAX installation on your PC, you can alternately install and start the platform daemon at a later time, if needed. Under the Windows Start menu, do this by selecting **Start > All Programs > Niagara <3.n.n> > Install Platform Daemon**

In summary (except for dial-out support), your Workbench PC’s local platform daemon is not used in making client connections to other Niagara hosts, only to provide the ability to run a station locally.

Provisioning versus platform interface

The focus in this document is about the NiagaraAX platform *user interface*, meaning the different platform views and functions available when you (a Workbench *user*) open a direct platform connection to a Niagara host. However, you should know that an AxSupervisor station running AX-3.1 or later can perform “provisioning”, which can automate some platform tasks. This typically applies to its subordinate JACEs, which are represented as NiagaraStations (devices) under its NiagaraNetwork.

1. In AX-3.1 and AX-3.2, a “provisioning” module is available for an AxSupervisor, which provides automation of platform tasks performed by the AxSupervisor *station*. Starting in AX-3.3, provisioning functionality broadened with the new modules “batchJob” and “provisioningNiagara.” In these cases only, browser access of the AxSupervisor station can provide platform connectivity, albeit indirectly. See “[Provisioning versus platform interface](#)” on page 1-2.

For more details, please see either of the following:

- AX-3.3 or later Axsupervisor: “[Niagara Provisioning overview](#)” in the *NiagaraAX Provisioning Guide for Niagara Networks* document.
- AX-3.1 or AX-3.2 Axsupervisor: “[Provisioning overview](#)” in the *NiagaraAX Provisioning Guide*

Note: *A few of the provisioning views provided by an Axsupervisor are nearly identical to platform views described in this document, including the [Software Manager](#) and [Application Director](#) (Station Director), and work in the same fashion. However, if new to NiagaraAX, it is recommended that you become familiar with “direct” platform views described in this document, before using provisioning in an Axsupervisor.*

Types of platform views

A Workbench platform connection to a Niagara host (JACE or Axsupervisor) provides various functional views, as shown in [Figure 1-1](#).

Note: *In addition to the platform views listed below, a Commissioning Wizard is available as a right-click platform option. This wizard provides a “next step” method to perform a sequence of platform tasks used for commissioning a new JACE controller, or when upgrading an existing JACE. For more details, see “[About the Commissioning Wizard](#)” in the *JACE NiagaraAX Install & Startup Guide*.*

The following sections summarize the various platform functions and views, including typical usage:

- [Application Director](#) (before AX-3.3, **Station Director**)
To start, stop, restart, or kill a station on the Niagara platform. Output from the station displays in the view pane, useful for monitoring and troubleshooting. You also configure a station’s “Auto-Start” and “Restart on Failure” settings from this view.
- [Ddns Configuration](#) (AX-3.1 or later platforms only)
Allows for DNS IP addresses to be dynamically updated (DDNS), an option sometimes used for JACE hosts, typically in a dialup connection scenario—although infrequently.
- [Dialup Configuration](#)
To configure the platform’s dialup modem, including settings for a “captive network.” If a QNX-based JACE, separate “listening” settings apply for receiving dialup connections.
- [Distribution File Installer](#)
Used to upgrade NiagaraAX in a remote JACE. It compares the remote platform’s Niagara runtime environment (nre) to locally available distribution (.dist) files, and if desired, updates the remote platform. The nre is divided into two separate .dist files: a “config” and a “core,” where the config is customizable after installation.
In addition, you use this platform view to restore a backup to the target JACE.
- [File Transfer Client](#)
To copy files between your Workbench PC and the remote Niagara platform (in either direction).
- [Lexicon Installer](#)
To install Niagara lexicon files from your Workbench PC to the remote Niagara platform, in order to provide non-English language support.
- [License Manager](#)
To review, install, save, or delete licenses and certificates on the remote Niagara platform.
- [Software Manager](#)
To review, install, update, or uninstall “installable software” on the remote Niagara platform. This includes Niagara modules (.jars) as well as separate .dist files for NRE core, Java VM, and (QNX) OS. The Software Manager compares software installed on the connected platform against that available (locally) on your Workbench PC.
- [Platform Administration](#)
To perform configuration, status, and troubleshooting of the Niagara platform daemon. Included are commands to change time/date, backup all remote configuration, and reboot the host platform.
- [Station Copier](#)
To *install* (copy) a station from your Workbench PC and the remote Niagara platform, including different file-level options. Also to *backup* (copy) a station in a remote platform to your Workbench PC, or to *delete* a remote station. You can easily rename any copied stations.
- [TCP/IP Configuration](#)
To review and configure the TCP/IP settings for the network adapter(s) of the Niagara platform.
- [User Manager](#)
Applies to remote Win32 platforms (JACE-NX). To access host Windows OS user and group accounts, including ability to add or delete users/groups, change passwords and group members.
- [Remote File System](#)
To navigate among all files and folders under the platform’s Niagara root (system home) directory, including the ability to make local copies on your PC.

About platform differences

Depending on the platform *type* opened, some platform views differ. There are two general categories of platforms, by OS (operating system) used:

- [QNX-based](#)
- [Win32-based](#)

Among the two platform types, there are various host *models*, each with a “model” string descriptor. For a current list of host models, see “[Models of platforms](#)” on page 1-6.

QNX-based

Sometimes called “embedded” JACE controllers, these include JACE-2, -4, -5, and -6 series controllers, all shipped with the QNX operating system. These devices use onboard *flash memory* for file storage. An option for an onboard dial-up modem is available, or if needed, an external modem can be used.

All QNX-based JACE models have an integrated backup battery. The backup battery allows continuous operation during brief power outages. The [JACE power monitoring](#) feature allows monitoring AC power and backup battery level, and a configurable delay for orderly shutdown of the JACE upon AC power failures. Access power monitoring of a QNX-based JACE in the **PowerMonitorService** in a running *station*, see “[About Platform Services](#)” on page 1-66.

For any QNX-based platform, the following platform views differ from [Win32-based](#) platforms:

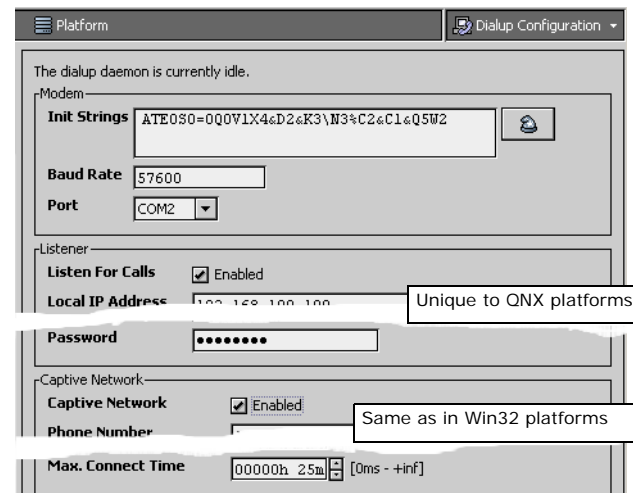
- [Dialup Configuration](#)
- [Platform Administration](#)

Also, in the [Application Director](#), you cannot **Start** a station after manually stopping it—you must reboot the JACE instead. See “[Station and output controls](#)” on page 1-11.

Dialup Configuration

Dialup Configuration for a QNX-based platform has 3 sections ([Figure 1-2](#)):

Figure 1-2 Dialup Configuration for QNX-based platform



- The [Modem section](#) includes properties to select JACE Comm port used, as well modem initialization strings and baud rate.
 - A [Listener section](#) allows configuration for receiving calls.
 - A [Captive Network section](#) is available (the same as for a [Win32-based](#) JACE-NX).
- See “[Dialup Configuration](#)” on page 1-14 for more details.

Platform Administration

Platform Administration for a QNX-based platform ([Figure 1-3](#)) differs as follows:

Figure 1-3 Platform Administration for QNX-based platform

Filesystem	Total	Free	Block Size
/ffs0	28,191 KB	9,537 KB	512 bytes

- An **FTP / Telnet** button is available. For details, see “FTP/Telnet” on page 1-49
- Selections possible in the **Update Authentication** function are simpler.
- Various data in summary information (repeated in View Details) differs greatly from Win32 hosts. See “Platform Administration” on page 1-43 for more details.

Win32-based

Win32 (Windows 32 bit) platforms include the JACE-NXS (and the discontinued JACE-NX), the AX SoftJACE, and any “AxSupervisor” PC host. As their operating system, Windows XP (if an AxSupervisor, possibly Windows 2000) is used, and file storage is typically a hard drive.

Note: The JACE-NXS, a Win32-based platform introduced with AX-3.1, is available in both a CompactFlash-memory based model and a hard-drive based model. In either case, “Windows XP Embedded” is the operating system. However, the flash-based model is typically installed with a special UPS, using communications between the JACE and the UPS. This provides continuous operation during brief power-lost scenarios, as well as station monitoring ability.

For any Win32-based platform, the following platform views differ from QNX-based platforms:

- **Dialup Configuration**
- **Platform Administration**

Note: When connected to any Win32 host, a **User Manager** platform view is also available. Intended use is for a Win32 JACE only. On an AxSupervisor PC, you typically configure Windows users and groups using normal Windows administrative tools.

Dialup Configuration

Dialup Configuration for a Win32-based platform has 2 sections (Figure 1-4):

Figure 1-4 Dialup Configuration for Win32-based platform

- The Modem section provides only a drop-down list to select a modem known to Windows.
- No “Listener section” to configure for receiving calls (you must do this instead using Windows Dialup Networking).
- A **Captive Network** section is available (the same as for any QNX-based JACE).

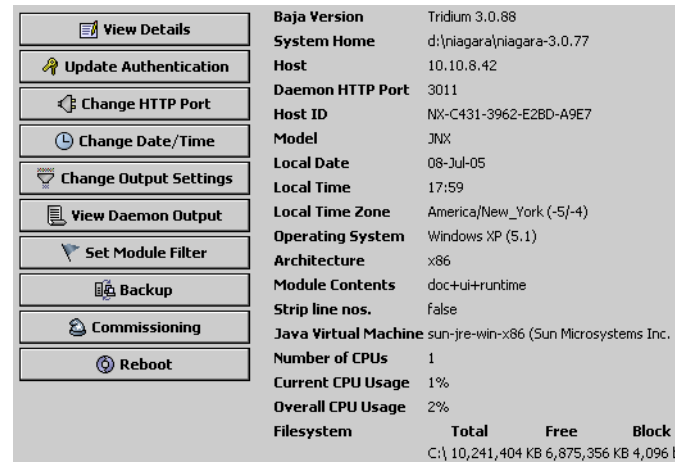
Note: Captive Network applies mostly to configuration of a JACE, vs. an AxSupervisor PC.

See “Dialup Configuration” on page 1-14 for more details.

Platform Administration

Platform Administration for a Win32-based platform (Figure 1-5) differs as follows:

Figure 1-5 Platform Administration for Win32-based platform



- No **FTP** / **Telnet** button is available (this configuration can be done directly using Windows).
- Choices available from the **Update Authentication** function are more involved.
- Various data in summary information (repeated in View Details) differs greatly from QNX hosts.

Note: If you have your local PC platform open, such as an AxSupervisor, buttons **Commissioning** and **Reboot** (and if AX-3.1 or later, **Set Module Filter**) are unavailable.

- Setting the Module Filter is intended only for initial configuration in a remote JACE. For more details, see “**Set Module Filter**” on page 1-50.
- The Commissioning Wizard is intended only for initial Niagara installation and startup in a remote JACE, or when upgrading a JACE. For more details, see “**Commissioning**” on page 1-53.
- Reboot is intended only for remote JACE platforms (see “**Reboot**” on page 1-53). To locally reboot an AxSupervisor, you should stop its local station, exit Workbench, then restart Windows.

See “**Platform Administration**” on page 1-43 for more details.

Models of platforms

Among the two groups of JACE controllers (**QNX-based** and **Win32-based**), there are different *models*, each of which has a “model” text descriptor. You see this model descriptor in the **Station Manager** view of a Niagara Network (**Host Model** column), and also in platform views such as **Platform Administration**, as well as the **PlatformServices** container of a station running on that host.

The following table lists various platform models (including JACEs), known at the time of this document, starting with the model text descriptor.

Table 1-1 Models of platforms

Model desc.	Actual Model	Notes
J401	JACE-401 series	Discontinued model, QNX-based.
J402	JACE-402P	Note: For new controller commissioning details for any QNX-based JACE model (except Security JACE), see the <i>JACE NiagaraAX Install and Startup Guide</i> .
J403	JACE-403 series	
J404	JACE-545 series, JACE-432 series	
J511	JACE-511 series	Discontinued models, QNX-based.
J512	JACE-512 series	
JNX	JACE-NX series	Discontinued model, Win32-based. See the the <i>JACE-NX NiagaraAX Install and Startup Guide</i> for commissioning details.
JNXS	JACE-NXS series	Win32-based. See the the <i>JACE-NXS NiagaraAX Install and Startup Guide</i> for commissioning details.
NPM2	JACE-2 series, incl. Security JACE (SEC-J-201)	QNX-based. JACE-2 series most common. For Security JACE commissioning details, see the <i>Security Appliance Guide</i> .

Model desc.	Actual Model	Notes
NPM6	JACE-6 series, incl. Security JACE (SEC-J-601)	QNX-based. JACE-6 series most common. For Security JACE commissioning details, see the <i>Security Appliance Guide</i> .
Jsoft	AX SoftJACE installed on user-supplied PC	Win32-based. Only an AX-3.1 or later SoftJACE appears this way, a 3.0 AX SoftJACE may appear as “Workstation”.
Workstation	User-supplied PC, either an AxSupervisor or engineering workstation.	Win32-based customer supplied PC that runs Workbench, minimally.

Application Director

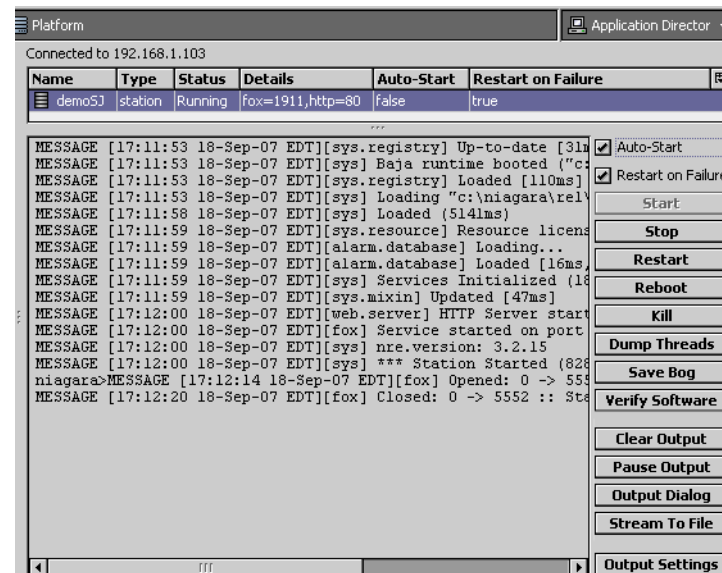
The Application Director (formerly: **Station Director**) is one of several [platform views](#). You commonly use it to *start* or *stop* a station in any Niagara host (whether a remote JACE or a local AxSupervisor PC), as well as see *station output* for troubleshooting purposes. From the Application Director you also define a station’s “restart” settings, plus have access to other station actions.

Note: *Starting in AX-3.3, this view’s name changed to the **Application Director**, as part of eventual (and additional) support for a different framework architecture, named “Sedona”. However, in the context of current NiagaraAX applications, that is “stations”, nothing changed in the functionality of this view. At this time, Sedona-related discussions about this view are not included in this document. Therefore, this entire section also applies to the “**Station Director**” if using a pre-AX-3.3 system.*

As shown in [Figure 1-6](#), the Application Director is split into three main areas:

- [Installed applications \(stations\)](#) (at top)
- [Station output](#) (main area)
- [Station and output controls](#) (right-side checkboxes and buttons)

Figure 1-6 Application Director view, looking at station



Note: *In the Application Director for any JACE, the “installed applications” area should show (at most) only one station, as shown in [Figure 1-6](#). However, the Application Director for a PC platform (AxSupervisor, Workbench workstation) may show multiple stations, as shown in [Figure 1-7](#).*

Figure 1-7 Application Director for AxSupervisor host showing multiple stations

Connected to localhost

Name	Type	Status	Details	Auto-Start	Restart on Failure	
AxDemo	station	Idle	fox=n/a,http=n/a	false	true	
AxSupT1	station	Idle	fox=n/a,http=n/a	false	true	
AxSupT2	station	Running	fox=1911,http=80	true	true	
BacTest89	station	Idle	fox=n/a,http=n/a	false	true	
demo	station	Idle	fox=n/a,http=n/a	false	true	
demoAppliance	station	Idle	fox=n/a,http=n/a	false	true	
demoTW	station	Idle	fox=n/a,http=n/a	false	true	

MESSAGE [10:05:26 19-Sep-07 EDT][sys.registry] Up-to-date
MESSAGE [10:05:26 19-Sep-07 EDT][sys] Baja runtime booted
MESSAGE [10:05:26 19-Sep-07 EDT][sys.registry] Loaded [156
MESSAGE [10:05:27 19-Sep-07 EDT][sys] Loading "d:\niagara\
MESSAGE [10:05:34 19-Sep-07 EDT][sys] Loaded (7375ms)
MESSAGE [10:05:37 19-Sep-07 EDT][alarm.database] Loading..
MESSAGE [10:05:37 19-Sep-07 EDT][alarm.database] Loaded [4
MESSAGE [10:05:37 19-Sep-07 EDT][sys] Services Initialized

☒ Auto-Start
☒ Restart on Failure
Start
Stop
Restart

Installed applications (stations)

The top area of the [Application Director](#) shows a table of installed applications (stations), as shown in [Figure 1-8](#). Apart from the data shown in the table, [station selections](#) are possible.

Figure 1-8 Application Director installed stations

Connected to 192.168.1.103

Name	Type	Status	Details	Auto-Start	Restart on Failure	
demoS3	station	Running	fox=1911,http=80	true	true	

Every 1.5 seconds, the platform daemon fetches data about the station(s) and updates this in the following columns:

- **Name**
The name of the station directory.
- **Type**
(AX-3.3 only) Currently “station” for NiagaraAX station, but as Sedona application support is introduced, other application type(s) may be possible.
- **Status**
One of the following, as applied to any station:
 - Idle — Station is not running, but can be started without a reboot.
 - Running — Station is running.
 - Starting — Platform daemon has started the station, but the station has not reported back its status back to the daemon.
 - Stopping — Daemon has ordered the station to stop, but its process has not yet terminated.
 - Halted — Station is not currently running, and cannot be restarted without a reboot.
 - Failed — Station terminated with a failure exit code.
- **Details**
For any station, shows two items:
 - fox= The TCP/IP port monitored for Fox connections to Workbench and other Niagara stations. Shows “n/a” if station is not running, or if it does not run the Fox service.
 - http= The HTTP port that the station’s WebService monitors for browser connections to the station. Shows “n/a” if station is not running, or if it does not have a running WebService.
- **Auto-Start**
Either true or false. If true, the station starts whenever the platform daemon starts. Configured with a right-side checkbox (see “[Start checkboxes](#)” on page 1-11).
- **Restart on Failure**
Either true or false. If true, the daemon automatically restarts the station after it terminates with a failure exit code. Configure with a right-side checkbox (see “[Start checkboxes](#)” on page 1-11).

Station selections

Click in the [installed stations](#) table for different results, as follows:

- **click**
Click a station to select it, highlighting it. When a station is selected, its standard output appears, and all right-side buttons apply to it. For details, see “[Station output](#)” on page 1-9 and “[Station and output controls](#)” on page 1-11.
- **right-click**

Right-click a station for its shortcut menu (a *subset* of the station and output actions buttons). For details on included menu commands, see “[Station and output controls](#)” on page 1-11.

- **double-click**

If running, double-click a station to open a Workbench (Fox) connection to that station.

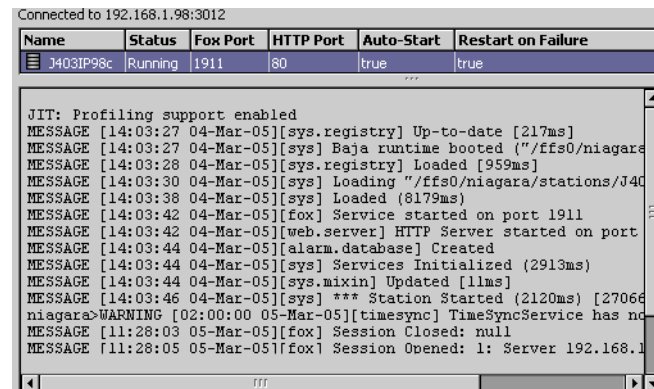
Press Ctrl and double-click to create a *new tab* for showing the station connection.

If not running, a station double-click does nothing.

Station output

The largest area in the [Application Director](#) view shows the “standard output / standard error” output text for the selected station, as shown in [Figure 1-9](#).

Figure 1-9 Station output area in Application Director



Depending on station selection, the station output text is one of the following:

- If a running station, output updates in real time. As more text is written by the station, it is appended to the bottom of the output area.
- If station is not running, output text is from the most recent execution of that station.
- If no station is selected, output text area is blank.

Note: Use the Windows copy shortcut (Ctrl + C) to copy output text to the clipboard.

As needed, use scroll bars to view all text, and use the right-side output control buttons. For more details, see “[Output control buttons](#)” on page 1-12.

The following sections provide more details related to a station’s standard output:

- [Station output overview](#)
- [Station log levels \(spy:/logSetup\)](#)
- [Station LogHistory \(LogHistoryService\)](#)

Station output overview

Station output messages can include errors and warnings that let you why something is not working, as well as simple informational messages about events as they occur. If needed, you can also change the “level” of station output—see “[Station log levels \(spy:/logSetup\)](#)” on page 1-10.

The general format of a station output message is:

```
TYPE [timestamp] [station_process] message_text
```

For example:

```
MESSAGE [13:53:08 08-Mar-05][fox] Service started on port 1911
```

Message types seen in station output include the following, by leading text descriptor

- **MESSAGE**
Typically comprise most default station output messages. Usually, each message lets you know some process milestone was started or reached, such as a service or the station itself.
- **WARNING**
Informs you of a potential problem, such as inability to open a specific port. Typically, warnings do not keep a station from starting.
- **ERROR**
Informs you of a problem that might keep the station from starting. Or, if it can start, an error that prevents some function of the station from operating correctly.
- **TRACE**

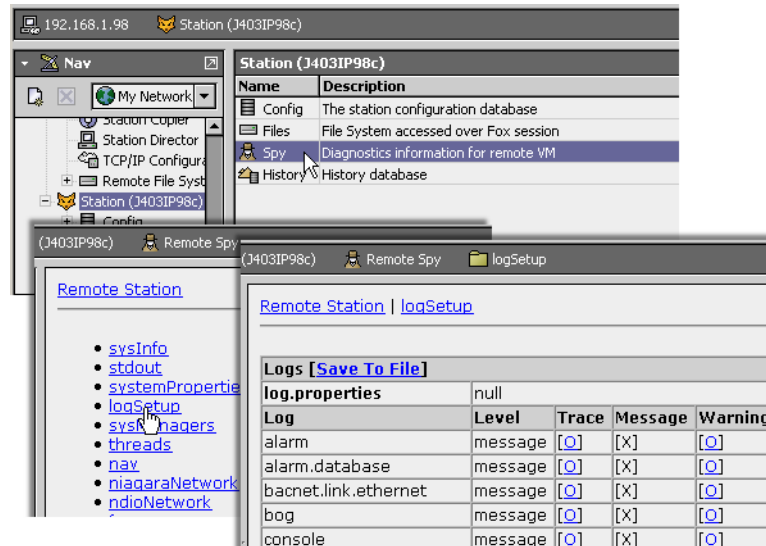
A verbose debug-level message that may be generated upon every process transaction. Useful only in advanced debugging mode. You see these for station processes only if you have set the log level at “Trace”.

In addition to the “typed” output messages described above, occasionally you may see a string of “java exception” text in the a station’s output. This indicates an unforeseen station execution problem. If an exception reoccurs, copy the exception text and report the problem to Systems Engineering.

Station log levels (spy:/logSetup)

A running station is actually a combination of many ongoing processes. Using the station’s spy “logSetup” page (Figure 1-10), you can change the “log level” of the station processes of interest in order to “tune” station output.

Figure 1-10 Station spy logSetup (from Station Summary)



Note: To get to a running station’s logSetup page in Workbench, double-click the station in the Nav tree for its **Station Summary** view. From there, double-click **Spy**, then click **logSetup**.

By default, all station processes have a “Message” log level (level selection denoted by [X]). To change the level of any listed process, click in the desired level column.

Level columns are ordered left-to-right in decreasing order of message volume, as follows:

- **Trace**
Returns all message activity (*verbose*). This includes all transactional messages, which may result in too many messages to be useful. Be careful using Trace!
- **Message**
(Default) Returns informational “MESSAGE”s, plus all “ERROR” and “WARNING” types.
- **Warning**
Returns only “ERROR” and “WARNING” type messages (no informational “MESSAGE”s).
- **Error**
Returns only “ERROR” type messages (no “WARNING” or informational “MESSAGE”s).
- **None**
No messages are returned to the station’s output.



Caution Increasing station output by assigning trace levels consumes extra station resources and exacts a performance penalty! After troubleshooting, return log levels to default values.

Station LogHistory (LogHistoryService)

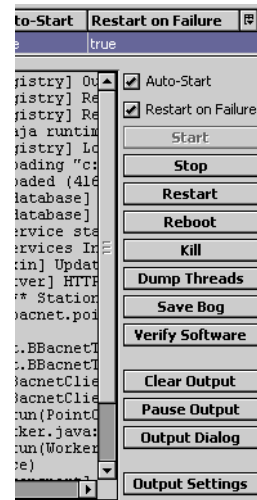
If the station is configured with the LogHistoryService (under its Services container), it maintains a buffered history (“LogHistory”) of *some* of the messages seen in the station’s standard output. In the LogHistoryService’s configuration, you specify its log level, meaning the minimum message type (from station output) to log. By default, the log level (property “Minimum Severity”) is **Error**. You may wish to change this to **Warning**.

This is mentioned because when looking at a station's output, you are usually troubleshooting. As part of troubleshooting, you should always check the station's histories for LogHistory. It should contain recently recorded station errors and (if configured) warnings. This information may help when evaluating "live" output from the station.

Station and output controls

Unlike in most Workbench views, where changes are entered first and then applied with a "Save" button, in the [Application Director](#) when you click checkboxes and buttons ([Figure 1-11](#)), changes are applied *immediately* to the selected station.

Figure 1-11 Application Director checkboxes and buttons



From top-to-bottom, these controls are grouped as follows:

- [Start checkboxes](#) (Auto-Start, Restart on Failure)
- [Station control buttons](#) (Start, Stop, Restart, Reboot, Kill, Dump Threads, Save Bog, Verify Software)
- [Output control buttons](#) (Clear Output, Pause Output, Output Dialog)
- [Output settings](#) button

Start checkboxes

For the currently selected station in the [Application Director](#), you can enable (check) or disable (clear) two start settings using checkboxes ([Figure 1-11](#)). Typically, for any JACE station you *enable both* checkboxes. In certain troubleshooting scenarios, you may clear **Restart on Failure** in order keep the station from constantly restarting (or host rebooting) after successive failures.

Note: Changes are reflected in the corresponding column of the Application Director's [installed stations](#) area.

The two start settings for the station are as follows:

- **Auto-start**
Specifies whether the station starts following platform daemon startup. This means following a host reboot, perhaps as a result of a power cycle, but possibly from a [Reboot](#) command.
Note: For any [QNX-based JACE](#) (JACE-2, -4, -5 series), a reboot also occurs following any installed software, meaning any module(s) or dist file(s), as well any TCP/IP-related changes.
- **Restart on Failure**
Specifies whether the platform daemon restarts the station if the station's process exits with a non-zero return code (e.g., engine watchdog had killed the station because of a "deadlock" condition).
Note: [QNX-based JACEs](#) cannot have a station restart without a reboot. Therefore, if this setting is enabled on such a JACE, if the station fails (terminates with error), the JACE reboots.

If the JACE continues to have 3 "automatic reboots" like this within an hour, the station remains in a "Failed" state, regardless of the setting above.

Station control buttons

For the selected station in the [Application Director](#), station control buttons ([Figure 1-11](#)) apply as follows:

Note: Be careful about using station controls, and understand the difference between them before using them.

- **Start**
Enabled only if the station has an “Idle” or “Failed” status in the [installed stations](#) area. When pressed, that host’s platform daemon immediately starts that station. Text in the [station output](#) is cleared, and output messages begin with the new startup of that station.
*Note: If you manually stop a station on a QNX-based JACE (using **Stop** button), it has a status of “Halted.” In this case, the Start button will not be available. You must Reboot the platform to restart the station. This differs from a Win32-based host, where a stopped station shows “Idle.”*
- **Stop**
Enabled only if the station has a “Running” status in the [installed stations](#) area. When pressed, a pop-up confirmation appears. If you confirm, the host’s platform daemon shuts the station down *gracefully* (saving its configuration to its `config.bog` file, and potentially saving history data). See the preceding note about stopping the station on a QNX-based host.
- **Restart**
(Available [Win32-based](#) platforms only) Enabled only if the station has a “Running” status in the [installed stations](#) area. When pressed, a popup confirmation dialog appears. If you confirm, the host’s platform daemon shuts the station down gracefully, then restarts it.
- **Reboot**
Always enabled. When pressed, a popup confirmation dialog appears. If you confirm, the selected host is rebooted. This is considered a drastic action. For details, see “[Reboot](#)” on page 1-53.
- **Kill**
Enabled only if the station has a status of “Starting”, “Stopping”, or “Running” in the [installed stations](#) area. When pressed, a popup confirmation dialog appears. If you confirm, the host’s platform daemon terminates the station process immediately.
*Note: Always use **Stop** instead of **Kill**, unless unavailable (stuck for a long time as either “Starting” or “Stopping”). Unlike a station stop, a station kill does not cause the station to save its config.bog, nor does it update the station output area.*
- **Dump Threads**
Enabled only if the station has a “Running” status in the [installed stations](#) area. When pressed, the host’s platform daemon has the station send a VM thread dump to its [station output](#).
- **Save Bog**
Enabled only if the station has a “Running” status in the [installed stations](#) area. When pressed, the host’s platform daemon has the station locally save its configuration to `config.bog`.
- **Verify Software**
Enabled regardless of station status. When pressed, Workbench parses the station’s `config.bog` file and the host’s `platform.bog` file, looking for module references. Workbench then checks to see if those modules (and any other software upon which they depend) are installed. Any missing software is listed in a popup dialog, and if available in your Workbench installation, the dialog offers to install the missing software into the remote host.

Output control buttons

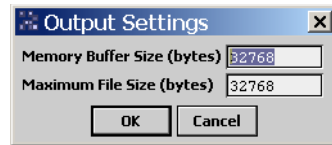
For the selected station in the [Application Director](#), output control buttons ([Figure 1-11](#)) apply as follows:

- **Clear Output**
Enabled regardless of station status. When pressed, all text in the [station output](#) area is cleared.
Note: Data in the station output area is fetched from a memory buffer in the platform daemon. Clearing the output does not actually clear the daemon’s buffer. Therefore, if you change the selection away from, and then back to the station, it re-fetches all buffered data.
- **Pause Output**
Enabled only if the station has a “Running” status in the [installed stations](#) area. When pressed, the button toggles to “**Load Output**”, and the next press back to “**Pause Output**” (and so on).
 - During a paused output, text remains frozen in the [station output](#) area. This is useful when the station is rapidly writing output.
 - When you press **Load Output**, text in the [station output](#) area is reloaded with the station’s buffered output, and output remains updating in real time.
- **Output Dialog**
Enabled regardless of station status. When pressed, this produces a separate “non-modal” output window displaying the same output text as in the Application Director’s [station output](#) area. Included are buttons to **Dump Threads**, **Pause Output**, **Clear Output**, and **Close** the window.
Note: You may find this “compact” version of a station’s standard output easier to work with than in the main area of the [Application Director](#). Also, if needed you can open multiple output dialogs for comparison purposes.

Output settings

For the currently selected station in the [Application Director](#), the **Output Settings** button produces a dialog ([Figure 1-12](#)) in which you specify how the platform daemon buffers the output from that station.

Figure 1-12 Output Settings dialog



The two available settings are in bytes, and are:

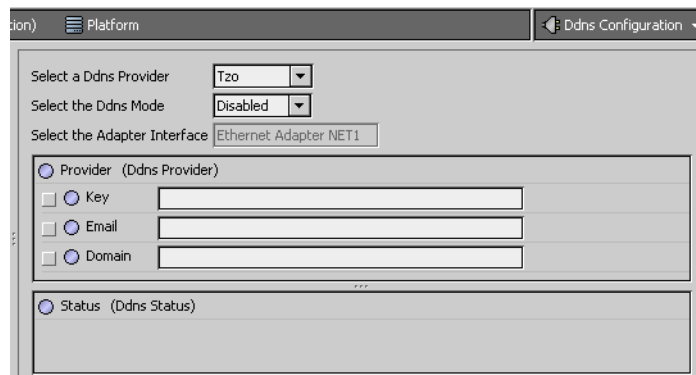
- **Memory Buffer Size**
Size of the memory buffer for the station output. If the station creates more output than the size of the memory buffer, the oldest output is lost.
- **Maximum File Size**
When the station stops, its output buffer is written to a `console.txt` file. This setting defines the maximum size of that file.

Note: Changes to either output setting may result in the output buffer's contents to be cleared.

Ddns Configuration

Ddns Configuration is one of several [platform views](#), available on any platform running AX-3.1 or later. DDNS (Dynamic Domain Name System) allows for DNS IP addresses to be dynamically updated. Typically, these are DHCP (Domain Host Configuration Protocol) addresses (Internet or Intranet) or dialup addresses. [Figure 1-13](#) shows the default **Ddns Configuration** view.

Figure 1-13 Ddns Configuration platform view



Note: Refer to the Engineering Notes document *“NiagaraAX-3.1 DDNS”* for more details and examples.

The following sections provide a few basic DDNS configuration details:

- [DDNS core configuration items](#)
- [About TZO](#)

DDNS core configuration items

The platform [Ddns Configuration](#) view has the following configuration sections:

- [Provider](#)
- [Mode](#)
- [Captive Network section](#)

Provider

The top property in the [Ddns Configuration](#) view is to select from a list of supported DDNS providers. Currently, the only supported provider is TZO (see *“About TZO”* on page 1-14), although future builds may support other providers.

Figure 1-14 DDNS Provider selection and Provider-supplied configuration properties

The screenshot shows a configuration window for DDNS. At the top, there are three dropdown menus: 'Select a Ddns Provider' (set to 'Tzo'), 'Select the Ddns Mode' (set to 'Tzo'), and 'Select the Adapter Interface' (set to 'VIA Rhine III Fast Ethernet Adapter'). Below these, there are three radio buttons under the heading 'Provider (Ddns Provider)': 'Key', 'Email', and 'Domain'. Each radio button is followed by a text input field. At the bottom, there is a radio button labeled 'Status (Ddns Status)'.

Note: A DDNS account with a provider is typically a fee-based subscription, in which you must first register and pay before your account is active.

Once you select the DDNS Provider, you require information from that provider about your DDNS account, in order to populate the following properties in the Provider section (Figure 1-14).

- **Key**
Furnished by provider (TZO) after account creation.
- **Email**
Email address associated with the provider (TZO) account.
- **Domain**
Domain name associated with the DDNS account.

Mode

This property in the [Ddns Configuration](#) view selects the operational mode of DDNS.

DDNS mode choices include:

- **Disabled**
DDNS functionality is completely disabled.
- **Internet**
Uses the IP address assigned to the adapter specified in the [Adapter](#) property.
- **Intranet**
Uses the IP address as seen when connected to the DDNS provider (note that not all providers will support this).
- **Dialup**
Use the IP address assigned once a dialup connection has been established.

Adapter

This property in the [Ddns Configuration](#) view applies if DDNS [Mode](#) is Internet, and selects the adapter to interrogate for an IP address.

About TZO

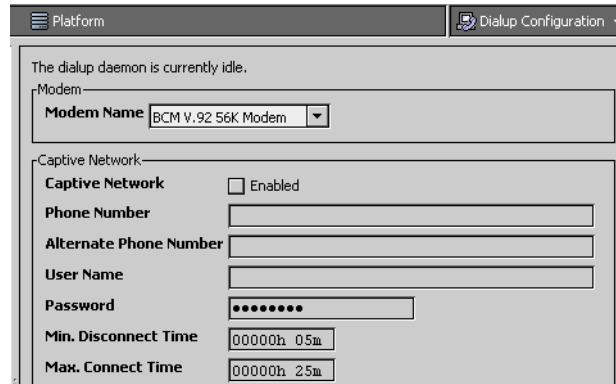
You can create a DDNS account with TZO (Tzolkin Corporation) at <http://www.tzo.com>. Testing of the NiagaraAX [Ddns Configuration](#) has focused primarily with TZO accounts.

Dialup Configuration

Dialup Configuration is one of several [platform views](#). It provides platform setup of the host's modem and optional settings for "Captive Network" dialup support. For any [QNX-based](#) platform, an additional "Listener" section allows configuration of the modem for receiving calls.

Note: Dialup configuration varies by platform type, see ["About platform differences"](#) on page 1-4.

Figure 1-15 Dialup Configuration platform view



The following main sections provide dialup configuration details:

- [Platform dialup configuration sections](#)
- [About the dialup daemon and service](#)
- [About dialup operation](#)

Platform dialup configuration sections

The platform [Dialup Configuration](#) view has the following configuration sections:

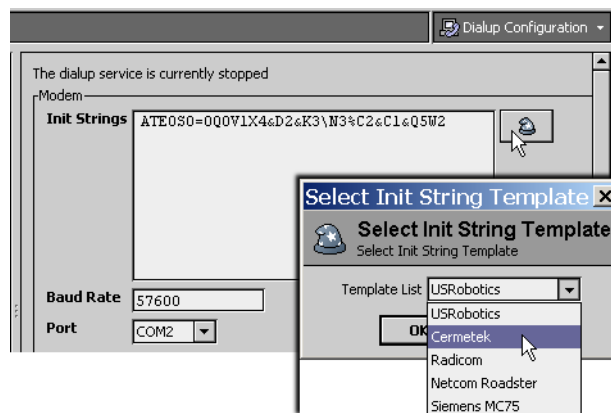
- [Modem section](#)
- [Listener section](#)
- [Captive Network section](#)

Note: If the platform's [dialup daemon](#) is started when you make configuration changes, a [dialup daemon restart](#) occurs when you **Save**. See [“About the dialup daemon and service”](#) on page 1-17.

Modem section

This section of the platform [Dialup Configuration](#) view is to select the modem/port used by the platform for dialup access. For [QNX-based](#) JACEs, other parameters are available too ([Figure 1-16](#)).

Figure 1-16 Modem section properties, platform Dialup Configuration (QNX-based JACE only)



QNX-based Modem properties include:

- **Init Strings (AX-3.1 or later) or Init String**
The modem initialization string(s) sent to the modem upon station restart, dialup service start, and any dial out activity. This configures the modem's options for things like error correction, data compression, flow control, and other parameters. Typically, Hayes-type commands are used. Starting in AX-3.1, support is provided for *multiple* init strings, useful in GPRS scenarios. In addition, a “wizard” Select Init String Template dialog was added (see [Figure 1-16](#)) which simplifies selecting and entering init strings for core-supported modems, including US Robotics, Cermetek, Radicom, Netcom Roadster, and Siemens MC75. Currently, init strings for modems connected to an AX-3.1 or later QNX-based JACE include:
 - Cermetek (typical optional onboard modem, see [“Init String notes”](#) on page 1-16):
ATE0S0=0Q0V1X4&D2&K3\N3%C2&C1&Q5W2
 - US Robotics 33.6 and above modem:

- AT&F1E0&A1X4
- **Radicom:**
ATE0S0=0Q0V1X4&D2&K3\N3%C2&C1&Q5W2
- **Netcom Roadster:**
AT&FE0\N3&K3&R1
- **Siemens MC75 (GPRS type):**
AT&D2
AT^SMONG
AT^SGAUTH=0
AT+CREG=1
AT+CGDCONT=1,ip,wwantrial.acfes.org
- **Baud Rate**
Modem's baud rate used for all activity. Default is 57600 (compatible with onboard 56K modem).
- **Port**
COM port name associated with modem. For most QNX-based JACEs, this will be either COM2 for an onboard dialup modem, or COM1 for an external modem.

Note: *If an AX-3.1 or later JACE-2/6 series controller has a GPRS (wireless) module installed and being used, the Baud Rate and Port properties (above) are automatically disabled. This allows dialup to use the GPRS module properly.*

Init String notes Due to varying command sets used among modems, determining the needed init string for any particular modem is often time consuming. For comparison purposes, here is a breakdown of modem commands used within the “known functional” init string for the Cermetek modem:

- AT — attention sequence
- E0 — echo off
- S0=0 — disable auto-answer
- Q0 — allow result codes to DTE
- V1 — return long form result codes (verbose)
- X4 — report basic call progress codes: OK, CONNECT, RING, NO CARRIER, etc.
- &D2 — interpret DTR On to Off transition per &Q0, &Q5, &Q6
- &K3 — enable RTS/CTS DTE/DCE flow control
- \N3 — elects auto reliable mode
- %C2 — enable v.42bis data compression
- &C1 — profile 1 after warm reset
- &Q5 — modem negotiates an error corrected link
- W2 — report DCE speed in EC mode

Listener section

This section of the [Dialup Configuration](#) view is for configuration of receiving calls (QNX-based platforms only).

Note: *In order to receive a call on Win32-based platforms, you must configure standard Windows Dialup Networking.*

As shown in [Figure 1-17](#), you must enable the “Listen For Calls” checkbox to access properties.

Figure 1-17 Listener properties in Dialup Configuration for QNX-based JACE

Listener properties include:

- **Local IP Address**
The local IP address to use once the connection has been established.
Note: *This is just a recommendation, there is no guarantee as to what the final address will be.*
- **Remote IP Address**
The remote IP address to use once the connection has been established.
Note: *This is just a recommendation, there is no guarantee as to what the final address will be.*
- **User Name**

- The dialup user name used when authenticating a dialup connection.
- **Password**
The dialup password used when authenticating a dialup connection.

Note: *User name and password is used to establish the dialup connection itself, before the additional (and independently maintained) authentication needed to either open the station or a host platform session. On the remote (calling) side, the corresponding dialup user name and password is part of the “dialup address” associated with this host platform. See “About dialup operation” on page 1-19 for more details.*

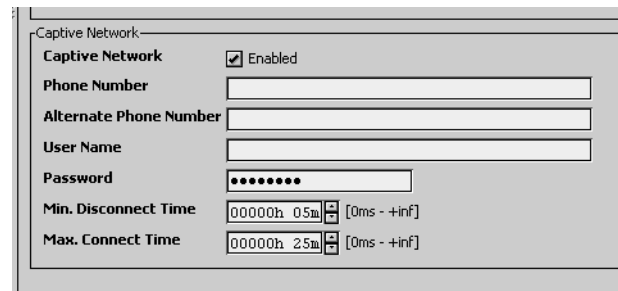
Captive Network section

This section of the [Dialup Configuration](#) view is for any modem-equipped JACE. A captive network is where the JACE can try and remain connected to a dialup network.

When used in this fashion, the station is generally not aware that a dialup connection is being used, but instead simply tries to connect to a dialup address. If the dialup connection breaks, the JACE tries to re-establish the connection.

As shown in [Figure 1-18](#), you must enable the “Captive Network” checkbox to access properties.

Figure 1-18 Captive Network properties in Dialup Configuration



Captive Network properties include:

- **Phone Number**
The phone number to dial.
- **Alternate Phone Number**
An additional phone number to dial if the first one fails.
- **User Name**
The dialup user name used when authenticating.
- **Password**
The dialup password used when authenticating.
- **Min. Disconnect Time**
The minimum amount of time, after a forced hangup, before a new captive network connection can be initiated (does not apply to Workbench).
- **Max. Connect Time**
The maximum amount of time a captive network-initiated connection can remain established before a hangup is forced (does not apply to Workbench).

Note: *If configuring a JACE for a captive network connection, to ensure that there is time allowed for incoming calls, it is recommended that you set the “Min. Disconnect Time” to be greater than the “Max. Connect Time.” Furthermore, in the station running on the JACE, for each NiagaraStation accessed via dialup, its dialup address also has similar connection time properties. In this case, you should set each “Min. Disconnect Time” (both captive network, and that in each NiagaraNetwork dialup address) to be greater than the sum of all “Max. Connect Times” (the captive network one plus all NiagaraNetwork dialup address ones).*

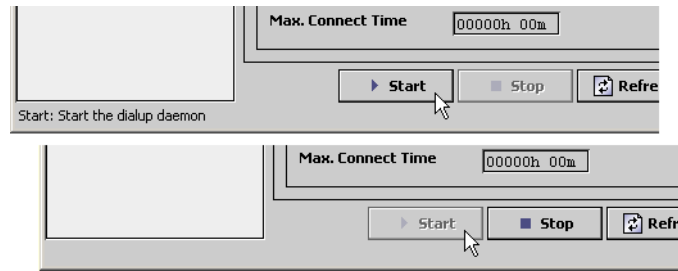
About the dialup daemon and service

Dialup operation is enabled and disabled by starting or stopping the [dialup daemon](#) (process) in the host platform. To provide station access to the dialup daemon, a running station has a corresponding platform service under its Services container: the [DialupService](#).

Dialup daemon

Any modem-equipped Niagara platform requires you to start the dialup daemon to allow modem operation. Click the **Start** button at the bottom of the platform [Dialup Configuration](#) view to do this, as shown in [Figure 1-19](#).

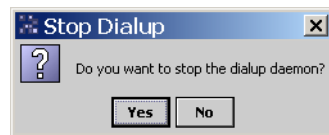
Figure 1-19 Start dialup daemon from bottom of platform Dialup Configuration view



When you start the dialup daemon, platform dialup configuration becomes effective, and modem operations are enabled. Once successfully started, the Stop button becomes enabled.

Note: If you click **Stop** to stop the dialup daemon, a confirmation dialog appears (Figure 1-20).

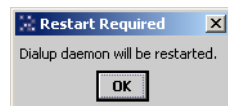
Figure 1-20 Stop dialup daemon dialog



If you click **Yes**, modem operations are disabled (any active dialup session is dropped).

Dialup daemon restart If the [dialup daemon](#) is already started, and you make any platform dialup changes, the daemon is restarted upon **Save**. A notification dialog appears, as shown in Figure 1-21.

Figure 1-21 Restart dialup daemon notification



Upon a dialup daemon restart, any active dialup session is dropped.

DialupService

Among a station's [PlatformServices](#) is a DialupService, the station's interface to the platform's [dialup daemon](#). Three slots of interest on the DialupService include:

- **Daemon Session**
Shows the current connection state to the dialup daemon (not to be confused with a dialup session with a remote host). The possible values are:
 - Init
 - Auth
 - Active
 - Closed

Note: Values are “just-in-time” types and do not change unless some dialup code executes. In other words, just starting the [dialup daemon](#) does not necessarily change this value.
- **Block Listen**
A StatusBoolean slot you can use to disable or enable listen functionality (if enabled in platform) by linking to station logic. See [Block out functionality](#).
- **Block Connect**
A StatusBoolean slot you can use to disable or enable a dialup connection by linking to station logic. See [Block out functionality](#).

Block out functionality If dialing functionality needs to be disabled programmatically (from the station), you can link the **Block Listen** and/or **Block Connect** slot of the [DialupService](#) to the StatusBoolean out of a controlling object or point.

An example would be to link a [BooleanSchedule](#) to one or both slots to disable dialup during certain periods of time.

Note: Block out functionality only works when a station is running. If no station is running, the defaults of `false` are assumed.

About dialup operation

Dialup in NiagaraAX can occur from Workbench (user-initiated call) or from a station. The following sections provide more details:

- [Dialup addresses](#)
- [Calling from Workbench](#)

Dialup addresses

When dialing out from Workbench or configuring a station to dial out, the dialup “Host” ord uses a *named* dialup address for making the connection. Dialup addresses are local to the platform that initiates the connection. In other words, dialup addresses used by your Workbench are unique to your instance of Workbench, dialup addresses used by a JACE station are unique to that station, and so on.

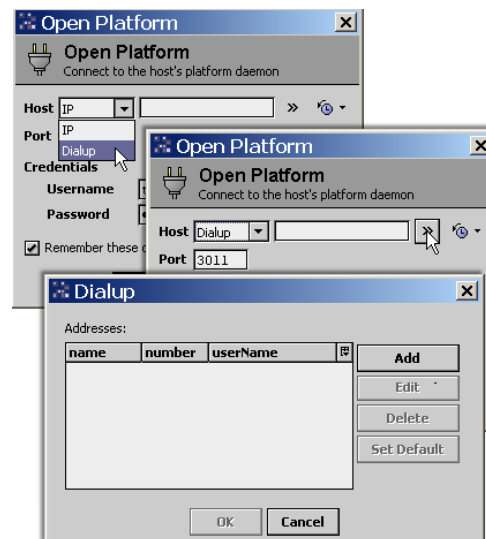
You add dialup addresses for Workbench or a JACE starting in different dialogs, as follows:

- [Workbench dialup address](#)
- [Station dialup address](#)

In either case, you use a [Dialup dialog](#) to add, edit, or delete dialup addresses.

Workbench dialup address In Workbench, you create dialup addresses by starting from the **Open Platform** or **Open Station** dialog, selecting Host type “**Dialup**,” and then using the >> button on the right side of the host field, as shown in [Figure 1-22](#).

Figure 1-22 Dialup dialog from Open Platform or Open Station dialog in Workbench

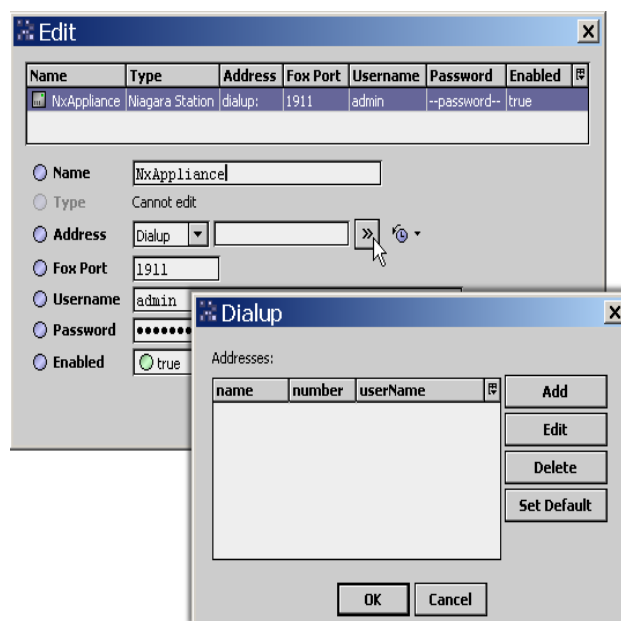


This produces the [Dialup dialog](#), for managing dialup addresses.

Station dialup address In a station, you create a dialup address by *manually adding* a target NiagaraStation (in the station’s NiagaraNetwork), using the **New** button.

In the station’s **New** (or **Edit**) dialog, you select Address type of “**Dialup**,” and use the >> button on the right side of the address field, as shown in [Figure 1-23](#).

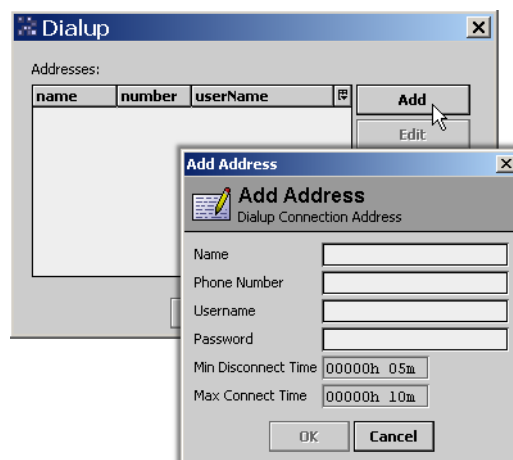
Figure 1-23 Dialup dialog from NiagaraStation New dialog in station



This produces the [Dialup dialog](#), for managing dialup addresses.

Dialup dialog In the Dialup dialog you manage all dialup addresses for use by the platform (Workbench PC or JACE), as shown in [Figure 1-24](#).

Figure 1-24 Add Address dialog for new dialup address



Buttons in the **Dialup** dialog allow you to:

- **Add** — Add a new dialup address.
- **Edit** — Edit an existing dialup address.
- **Delete** — Remove an existing dialup address.
- **Set Default** — Change the current default address.

When adding or editing a dialup address, the following fields are available:

- **Name**
Name for the dialup address, must be unique within that platform.
- **Phone Number**
Name for the dialup address, must be unique within that platform.
- **User Name**
Name for the dialup address, must be unique within that platform.
- **Password**
Name for the dialup address, must be unique within that platform.
- **Min. Disconnect Time**

Applies only to dialup address used by a station (not to Workbench). This specifies the minimum amount of time, following a disconnect from a dialup connection, before another dialup connection can be initiated.

- **Max. Connect Time**

Applies only to dialup address used by a station (not to Workbench). This specifies the maximum amount of time a dialup connection will remain established before a hangup is forced.

Calling from Workbench

You can open a platform or station connection to a JACE from Workbench using a dialup connection, just like making a LAN (IP) connection.

Note: *On your Workbench PC, the platform daemon must be installed and running. Also, the dialup daemon must be started. See “About the dialup daemon and service” on page 1-17 for details.*

To dialup a JACE from Workbench

To dialup a JACE from Workbench, do the following:

- Step 1 Select **File > Open**, then either **Open Platform** or **Open Station**.
 - Step 2 In the Open Platform or Open Station dialog, click the **Host** drop-down and select **Dialup**.
 - Step 3 Click the **>>** button on the right.

A popup **Dialup** dialog appears, showing all saved dialup addresses. See “Workbench dialup address” on page 1-19 for details.
 - Step 4 Click a dialup address to select it, then click **OK**.

The dialup address name is now in the “Host” field of the Open Platform or Open Station dialog.
- Note:** *You can also simply type in a known dialup address, without going to the popup dialog. Consequently, you can type in a phone number in the Host field instead of a dialup address name. This results in the default address being used, but uses the provided phone number instead of the default phone number.*
- Step 5 Type in the appropriate user name and password for the platform or station.
 - Step 6 Click **OK** to initiate the dial out to the JACE.
- If not currently connected, and the modem is not already in use, a dialup connection will be established. If the connection to the same host already exists, that connection will be reused. If a different connection already exists, this attempt will be rejected.

Distribution File Installer

The Distribution File Installer is one of several [platform views](#). You use this view to *restore* any locally available “backup” .dist file of a remote JACE. You can initiate a backup using the [Backup](#) command from the platform’s [Platform Administration](#) view, or directly from the **BackupService** in the station running on that host.

In addition to restoring a backup .dist, this view can be used for either of the following:

- To install a “clean” dist file, used to safely downgrade a JACE to an older release level, or to restore it to a “known good” state. Note that this *wipes* the file system (including all station files), leaving the JACE in a “near factory” state. For details, see “[Downgrading a JACE \(Clean Dist\)](#)” on page 1-23.
- Upgrade a previously installed JACE with a newer build of the Niagara runtime environment (NRE). In the process, updates can also occur to the remote host’s Java VM, and (if a QNX-based JACE) a newer QNX OS. All of this software is packaged in distribution files (“dist” for short).

Note: *Systems Engineering recommends that you use the platform **Commissioning Wizard** to upgrade a JACE, instead of the **Distribution File Installer**. The wizard is a right-click option on the platform when opened in Workbench. For details, see “[Upgrading a JACE](#)” on page 1-26.*

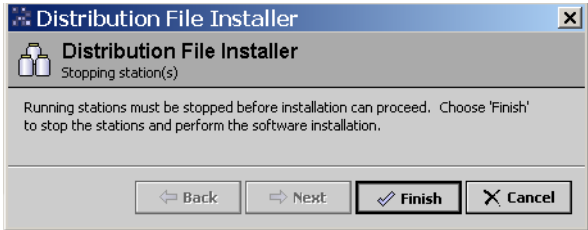
A dist file is a zip file that contains other files and a “manifest” that describes the contents of the distribution. For technical details, see the “Distribution” section in the online *Niagara Developer Guide*.

Whenever a station is already running on the remote host, *any* dist file install requires that station to be stopped. After selecting a dist file, the Installer provides a confirmation dialog for this, as shown in [Figure 1-25](#). When you finalize the install (click **Finish**), the Installer automatically stops the station, then continues with the [distribution file install process](#).



Caution Before finalizing any dist installation, make sure that controlled equipment that might be adversely affected by the JACE's station stopping and then restarting (from this installation) is put in a manually controlled state.

Figure 1-25 Stop station dialog in Distribution File Installer



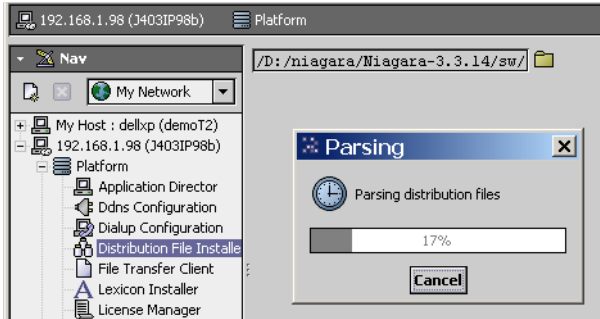
The following sections explain more details about dist file selection and the install process:

- [Dist file selection](#)
- [Module updating from Distribution File Installer](#)
- [Distribution file install process](#)

Dist file selection

When you select the Distribution File Installer, it “parses” through the dist files on your local PC, using the last source folder selected (Figure 1-26).

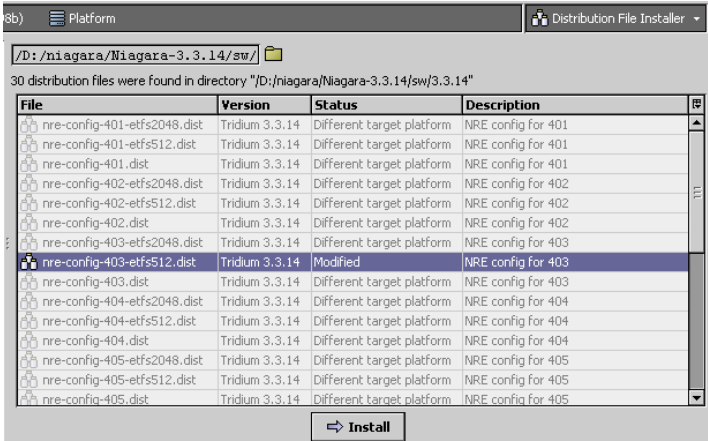
Figure 1-26 Parsing dialog in Distribution File Installer



By default, the first time you use the Installer, the most *recent* Niagara build folder under the default Niagara software location (<NiagaraRel>\sw\dist\3.n.nn) is parsed. If NiagaraAX was installed on your PC with the “installation tool” option, dist files were copied there. However, you can also click the folder control to produce a “Change Directory” dialog, and point the Installer to that location. You may need to do that, for example, when [Restoring a backup dist](#).

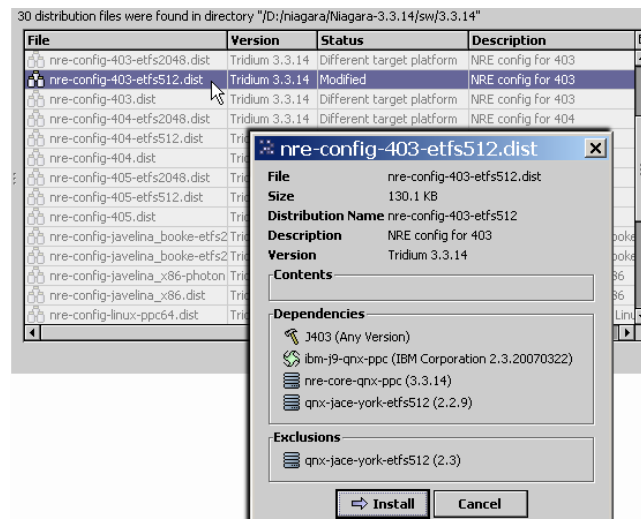
When parsing completes, a table of the found dist files appears, with the appropriate dist files available for selection in the table, as shown in Figure 1-27 (using default software location).

Figure 1-27 Available dist file in Distribution File Installer



Note: For details on a dist file, double-click it for a popup dialog, including a list of its dependencies (Figure 1-28).

Figure 1-28 Example details dialog for a dist file, showing dependencies.



Dist file selection differs depending on usage of the Distribution File Installer.

- [Restoring a backup dist](#)
- [Downgrading a JACE \(Clean Dist\)](#)
- [Upgrading a JACE](#)

Downgrading a JACE (Clean Dist)

Although the NiagaraAX Framework has historically provided smooth *upgrades* between software releases, *downgrading* from one software release to an earlier release can present compatibility problems. This applies particularly to [QNX-based](#) JACEs, as binaries for the (QNX) OS are included in dist files. Starting in AX-3.3, dependency checks may prevent a JACE downgrade in some cases (see [“AX-3.3 dependency check change”](#) on page 1-24).

There are times, however, when it is necessary to install an older release onto a JACE, or to restore a JACE to a known good state. During the AX-3.3 development period, “clean” dist files became available for this. To downgrade a JACE, or otherwise “wipe it clean to start over,” you can install a clean distribution file.



Caution

Installing a clean dist will wipe the file system and install the appropriate version of Niagara platform daemon, resetting the unit to a “near factory state.” Only the following settings are preserved:

- TCP/IP settings
- license files
- brand.properties

All other data is removed from the file system, including station bog files, Px files, modules, etc. If the JACE came with an appliance installed, installing a clean dist will remove that application.

In addition, a clean dist restores the factory-default platform credentials and port (3011).

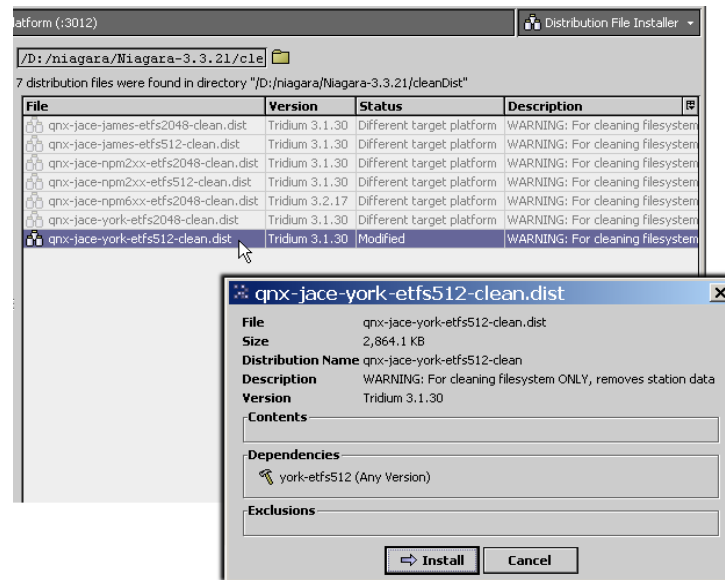
*Therefore before installing a clean dist file, make sure to backup station files plus any other modules on the JACE you wish to keep. Remember, that a “Station Backup” creates a dist file that (when restored) includes the same level of software installed at backup time, so instead of (or in addition to) this type of backup, you may wish to use the **Station Copier**. For related details, see [“Station Copier”](#) on page 1-54.*

*Note that after installing a clean dist, you must recommission the JACE. It is recommended you use the **Commissioning Wizard** to do this.*

A clean dist file has the suffix “-clean” in its name. Starting with Workbench build 3.3.21, a clean dist file for every QNX-based JACE hardware platform is located under a <NiagaraRel>\cleanDist folder—apart from other dist files under your software database. If using Workbench AX-3.2 or AX-3.1, you may be able to obtain clean dist files from Systems Engineering. If you do this, it is recommended that you create a similar “cleanDist” folder to store such files.

Clean dist files appear listed in the Installer with a “WARNING” in the Description column, as shown for the one highlighted in [Figure 1-29](#).

Figure 1-29 Clean dist file shown selected in the Distribution File Installer



Note: Currently, clean dists support re-installation of AX-3.1, AX-3.2, or AX-3.3, but not AX-3.0 (which uses a different file system). In the case of a JACE-6 platform, only re-installation of AX-3.2 or AX-3.3 is supported, as this platform has a dependency on AX-3.2 or later.

As for any dist file, only the appropriate one for the currently opened platform will be selectable. The following step summary describes the downgrade process for a QNX-based JACE from AX-3.3 to an earlier release.

Example JACE downgrade from AX-3.3 to an earlier release

To downgrade a QNX-based JACE from any AX-3.3 build to an earlier release:

- Step 1 Before installing a clean dist file, make sure you have backed up the station files plus any other data and modules on the JACE you wish to keep.
- Step 2 Start AX-3.3 Workbench and open a platform connection to the JACE.
- Step 3 Open the [Distribution File Installer](#) and change to the !/cleanDist directory (or if using a previous Workbench version, wherever the previously-obtained clean dist files were copied).
- Step 4 Select the appropriate clean dist file for the platform and install.

The file system clean will take a few minutes, then the JACE will automatically reboot. Wait for the reboot to complete.

Note: After reboot from a clean dist install, the JACE is using default platform credentials and port (3011).

- Step 5 To re-install the software versions to the JACE:
 1. Use a version of Workbench that runs the same software versions that you want on the JACE, and use the platform **Commissioning Wizard** to install the desired software build. For details, refer to the section [“About the Commissioning Wizard”](#) in the *JACE NiagaraAX Install & Startup Guide*.
 2. If you have a backup dist file for the JACE that was made when it had the desired older software versions, use the Distribution File Installer to install it. See the previous section, [“Restoring a backup dist”](#) on page 1-25.

AX-3.3 dependency check change

Because using software components from different NiagaraAX releases together on the same JACE can cause serious problems, starting in AX-3.3 dependency checks have become stricter. In particular, *downgrading* from one software release to an earlier release can present compatibility problems.

The AX-3.3 Distribution File Installer no longer allows the software provided in the following distribution files to be downgraded, except within the same minor software version (first two numbers in dotted version are the same):

- qnx-*.dist
- nre-core-qnx-*.dist
- nre-config-*.dist (except for nre-config-win-x86.dist)

Examples The following are examples of dist file dependency checks starting in AX-3.3:

- version 2.2.7 of the qnx-jace-james.dist file can be installed to a JACE that is currently running qnx-jace-james.dist version 2.2.11, but version 2.1.13 of that file cannot.
- version 3.3.15 of the nre-core-qnx-ppc.dist file can be installed to a JACE that is currently running nre-core-qnx-ppc.dist version 3.3.21, but version 3.1.29 of that file cannot.

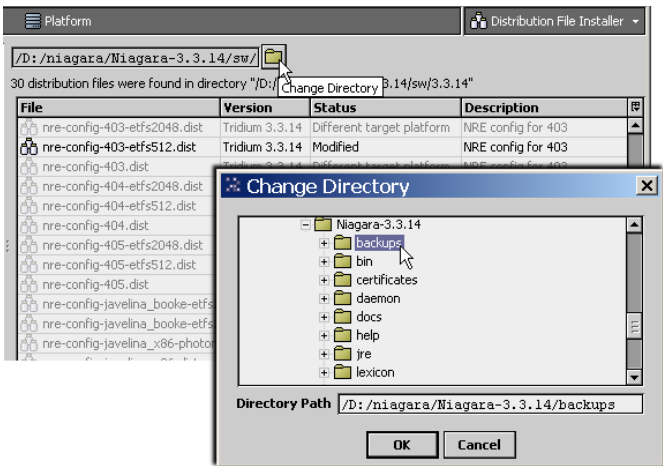
Note that during the AX-3.3 development era, “clean” dist files became available for the purposes of downgrading a JACE to an earlier release. See “Downgrading a JACE (Clean Dist)” on page 1-23.

Restoring a backup dist

A backup dist includes not only the entire station folder, but all other Niagara configuration that may be customized for that platform. This allows for a complete replication from the one backup file. You can do a backup from the Platform Administration view—for details, see “Backup” on page 1-52.

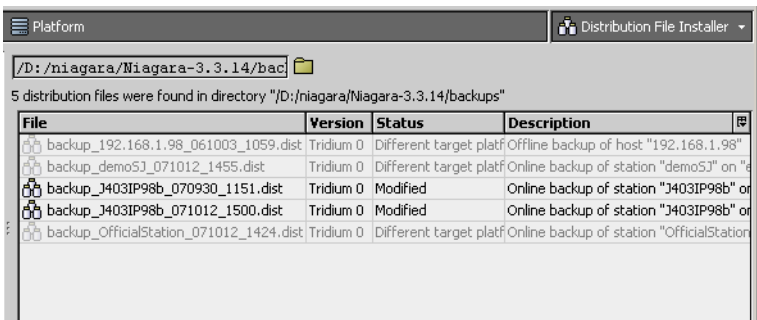
Note: More typically, backups are done from Workbench station connections (must be a station running, that has the BackupService). In the Nav tree, simply right-click the opened station, and select Backup Station. To do restore a backup, use the Change Directory control in the Distribution File Installer to point to the backup dist file location. For example, point to the backups subdirectory under your Niagara build folder (Figure 1-30).

Figure 1-30 Change Directory dialog from Distribution File Installer



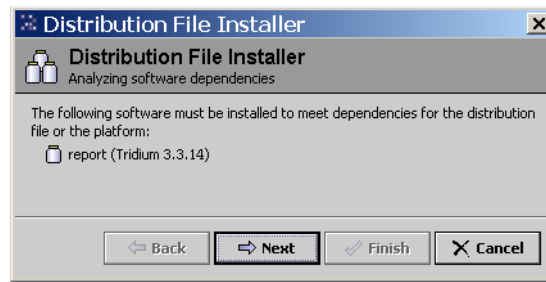
The Distribution File Installer parses through the available dist files, and makes selectable only those files available that are compatible with the opened JACE platform. When done parsing, available backup dists appear listed, as shown in Figure 1-31.

Figure 1-31 Backup dist files listed in Distribution File Installer



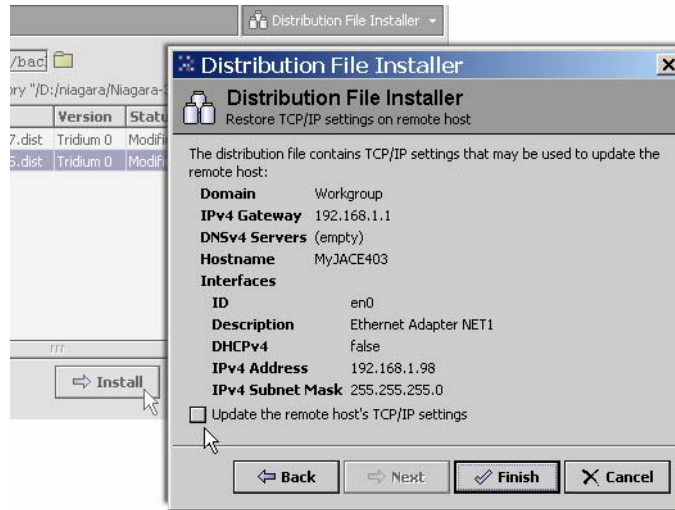
You can double-click any dist for more details in a popup dialog. To restore the backup, click **Install** to continue. If the host is already running a station, a dialog appears telling you that it must be stopped, as previously shown in Figure 1-25 on page 22. If the dist file contains software modules different from (or in addition to) those already installed in the remote host, another dialog appears to inform you, as shown in Figure 1-32. Click **Next** to continue.

Figure 1-32 Software dependencies dialog in Distribution File Installer



As shown in [Figure 1-33](#), whenever restoring a backup, another dialog always asks if you wish to restore the TCP/IP settings stored in the dist file (as displayed) into the remote host.

Figure 1-33 Restore TCP/IP settings in remote host from dist file



TCP/IP settings contained in the dist file are listed, and by default, the checkbox “Update the remote host’s TCP/IP settings” is *cleared* (AX-3.1 or later, or in build 3.0.101 and later).

- If you *leave* this cleared, after the dist file installs and host reboots, it *retains its current* TCP/IP settings. This allows you to use the same dist file on differently addressed hosts, if needed.
- If you *check* (select) this checkbox, after the dist file installs (and host reboots), it will use the TCP/IP settings stored in the dist file—meaning the same ones shown in this dialog.

When you click **Finish**, the [distribution file install process](#) begins.

Upgrading a JACE

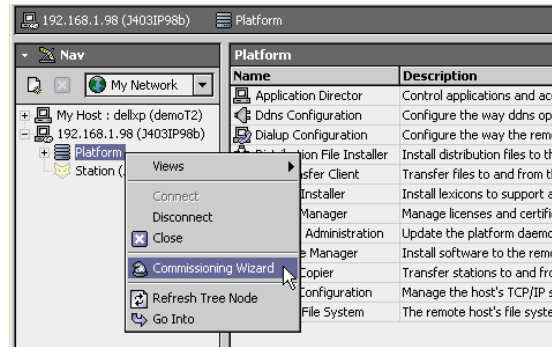
There are two methods possible to upgrade a previously-commissioned JACE:

- [Commissioning Wizard method \(recommended to upgrade JACE\)](#)
- [Distribution File Installer method \(if you must\)](#)

Commissioning Wizard method (recommended to upgrade JACE)

To avoid confusion about .dist files when upgrading a previously-commissioned JACE, it is recommended that you use the **Commissioning Wizard** instead of the Distribution File Installer. With a platform connection to any JACE, access this wizard by simply right-clicking on that platform and selecting it from the menu ([Figure 1-35](#)).

Figure 1-34 Commissioning Wizard is right-click option of opened platform



For a JACE upgrade, in the wizard's opening "selection of steps" you typically *de-select* most items that were previously run at the JACE's initial commissioning time—for example to set the module content filter level, date and time, install lexicons, and so on.

To upgrade a JACE, you *do select*:

- "Install/upgrade core software from a distribution file"
And, in the case where the upgrade also requires an updated license installed:
- "Request or install software licenses"

When you proceed in this manner, the wizard automatically finds and installs all core distributions needed for the JACE in the *first* portion of the upgrade. Next, after the JACE reboots and the wizard continues, the remotely installed Niagara modules are compared against those in your software database. See "[Module updating from Distribution File Installer](#)" on page 1-27.

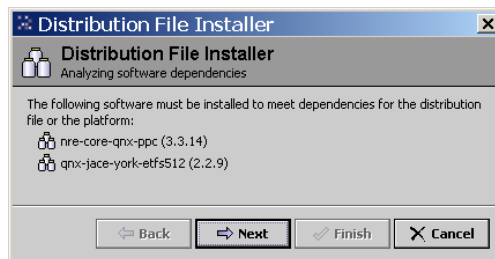
For further details, refer to the section "[About the Commissioning Wizard](#)" in the *JACE NiagaraAX Install & Startup Guide*.

Distribution File Installer method (if you must)

A JACE upgrade typically requires at least two separate dist files for the Niagara runtime environment (NRE): "nre-config" and "nre-core." The specific config file needed varies by JACE model, particularly among models of "embedded" (QNX-based) JACEs.

Although you can use the **Distribution File Installer** to upgrade a JACE, pointing to the default source folder for dist files (most recent Niagara build folder under the "sw" subfolder), this has proven confusing for some, and has lead to problems in some cases. Even though the Installer parses through the dist files and limits selection among ones applicable to the open platform, you can select only *one* dist file to install. To upgrade a JACE where *both* are not "up to date," the nre-config file (see [Figure 1-27](#) on page 22) is the *correct one to pick*. When you click the **Install** button, the Installer automatically informs you that dependent dist(s) are also needed in another popup dialog ([Figure 1-35](#)).

Figure 1-35 Distribution File Installer dependencies notification

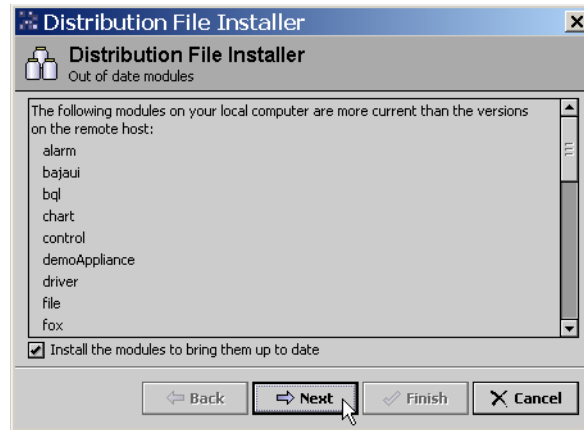


Clicking **Next** effectively selects both dist files for installation, plus any other binaries needed. The Installer then compares the remotely installed Niagara modules against those in your software database. See "[Module updating from Distribution File Installer](#)" on page 1-27.

Module updating from Distribution File Installer

By default, when upgrading the core distribution in an opened JACE platform, its versions of installed *modules* are checked too. If modules are found that are out-of-date (compared to the modules installed on your NiagaraAX workstation), it lists them in a dialog ([Figure 1-36](#)), with the option to install them *pre-selected*.

Figure 1-36 Update out of date modules in Distribution File Installer



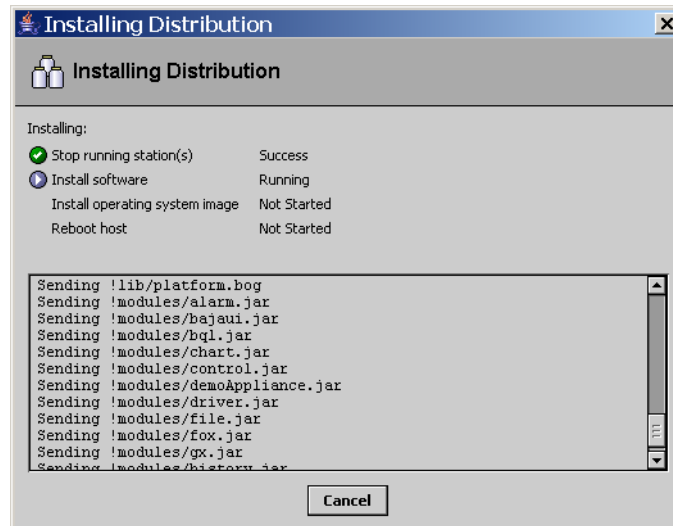
Typically, you accept this and select **Next** to continue the [distribution file install process](#). If modules are out-of-date and you *clear* the option to install them, you can install them *after* the distribution file installation completes, using the platform's [Software Manager](#) view.

Note: *If all modules installed are already up-to-date, you do not see the update modules step and dialog as shown in [Figure 1-36](#).*

Distribution file install process

After proceeding with **Finish**, the dist installation process appears in a dialog that tracks its progress as it continues, as shown in [Figure 1-37](#).

Figure 1-37 Distribution File Installer progress dialog



After the distribution file (and modules, if selected) are installed on the JACE platform, the JACE is rebooted, and the progress dialog indicates complete. You must click the **Close** button to continue. You can then reopen a platform connection, perhaps to view output in the [Application Director](#).

File Transfer Client

The File Transfer Client is one of several [platform views](#). It allows you to *copy* files and/or folders between your Workbench PC and the remote Niagara platform, as needed. You can also use it to *delete* files and folders.

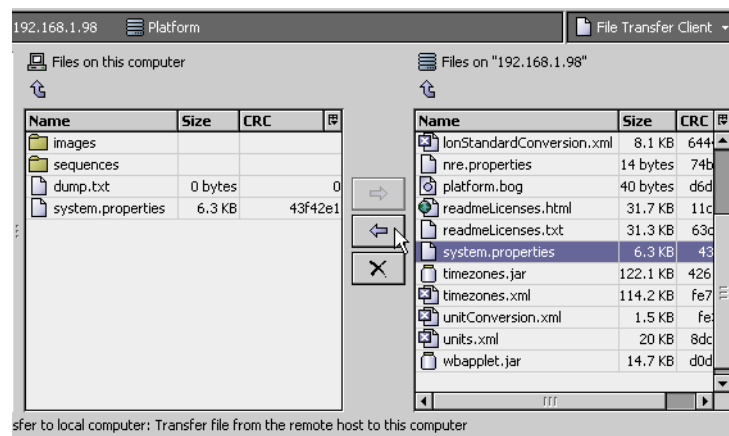
This may be useful if you wish to copy graphics images to a JACE, as one example. Or, you can use it to copy text files *from* the `!lib` folder of a remote JACE to your local PC, to allow editing. Then you can use the File Transfer Client to copy the edited version back to the the JACE's `!lib` folder. Examples for this type of usage include `lib files system.properties` and `timezones.xml`.

The File Transfer Client provides a two-pane view, as shown in [Figure 1-38](#).



Caution *Be careful when using the File Transfer Client, especially when copying files to a target JACE platform, or whenever using the delete (X) control. Note that in either direction, when transferring a file and an identically-named file already exists, a popup confirmation dialog appears before the copy. A popup confirmation dialog also appears before any delete. However, after confirmation there is no “Undo.”*

Figure 1-38 File Transfer Client

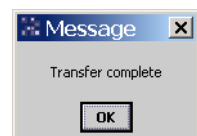


In the File Transfer Client view, the *left* pane provides access to *local* (Workbench PC) files, and the *right* pane provides access to files on the *remote* platform.

Use is straightforward, you simply click navigation controls at the top of each pane to go to the appropriate location for source and target. Then you click one or more items on one side (as source) to select for copying to the other side (target). Then, you click the appropriate transfer arrow.

A dialog appears when all files are transferred, as shown in [Figure 1-39](#).

Figure 1-39 Transfer complete dialog



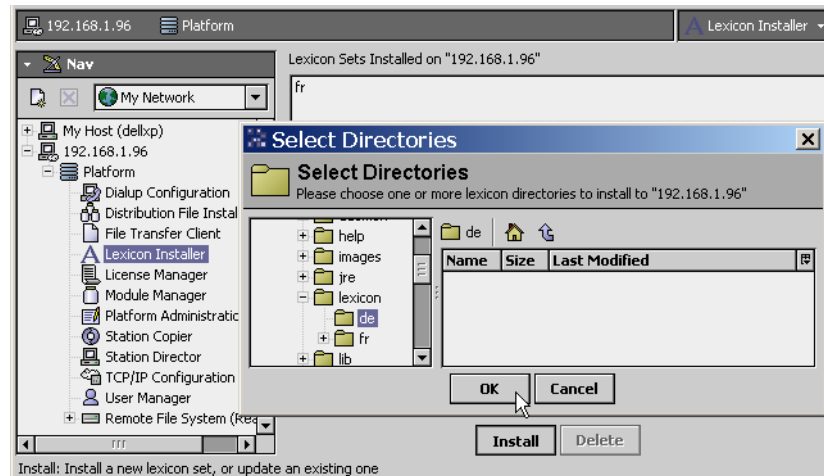
Lexicon Installer

The Lexicon Installer is one of several [platform views](#). This view lets you install lexicons (for non-English language support) from your Workbench PC to a remote JACE platform, as needed. Or, you may have modified the English (en) lexicon in order to change the wording used in default labels.

Beforehand, you typically use the **Lexicon Editor** tool in Workbench to review and edit entries (or *keys*) in the individual lexicon files with localized values needed for language support. See the *User Guide* sections [“About lexicons”](#) and [“Lexicon editor”](#) for more details.

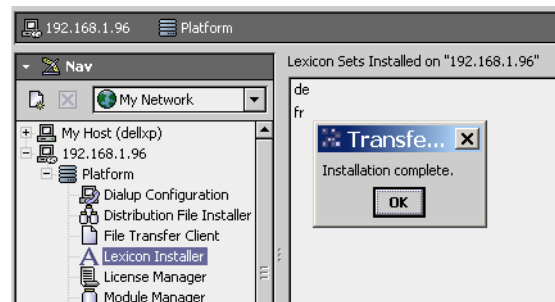
When you select Lexicon Installer, any existing lexicons (already installed in that platform) are listed in the view pane. When you click Install, a “Select Directories” dialog appears for you to select lexicons (on your Workbench PC) to install in the remote platform, as shown in [Figure 1-40](#).

Figure 1-40 Lexicon Installer, selecting lexicon



When you click **OK**, the selected lexicon directory is installed in the remote JACE platform. An “Installation Complete” dialog appears when all files are transferred, as shown in Figure 1-41.

Figure 1-41 Lexicon Installer, lexicon installed

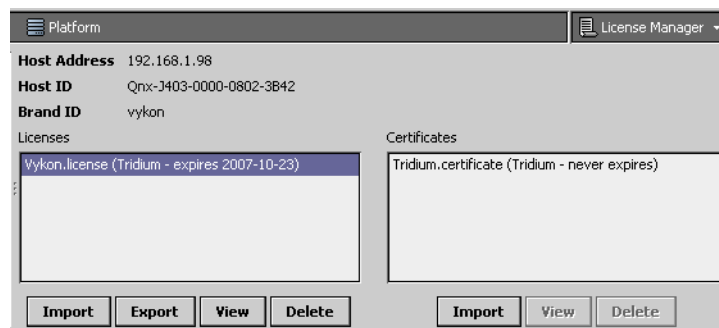


License Manager

The License Manager is one of several [platform views](#). This view lets you install (import) licenses and certificates to a remote JACE platform, sourced either from your Workbench PC or the Niagara licensing server. You can also view contents of licenses and certificates, and if desired, delete them from a JACE.

Note: *Starting in AX-3.3, Workbench management of licenses was enhanced to include a structured “local license database” and utilization of a “license archive file” format. In addition, there is a new Workbench “tool,” the Workbench License Manager, which does not require a platform (or station) connection to use. These new features are explained in an appendix to this document, “[License Tools and Files](#)”, along with details about the contents (features) of license files.*

Figure 1-42 License Manager (AX-3.3 version shown) lists existing licenses and certificates



A license and a certificate are each a digitally-signed text file, with differences briefly as follows:

- A *license* file is unique to a *specific Niagara host*, and enables a set of vendor *features*. All NiagaraAX hosts require a branded “Tridium” license. If third-party modules are installed, one or more addi-

tional licenses may be needed. For details about license file contents, see the section “About NiagaraAX license files” on page A-7.

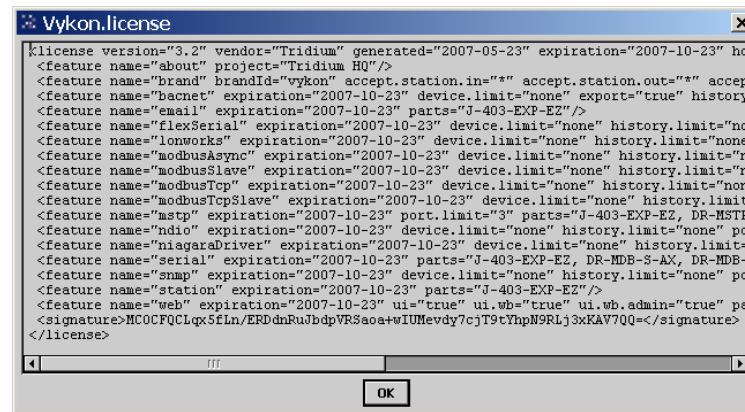
- A *certificate* file varies by *vendor*, and matches that vendor to a public key used for encryption. It is used for verifying the authenticity of license files. All NiagaraAX hosts require a “Tridium” certificate. If third-party modules are installed, one or more additional certificates may be needed.

The License Manager lists any existing licenses and certificates (already installed in that platform), as shown in Figure 1-42.

Note: Prior to AX-3.3 changes, the License Manager had different buttons at the bottom, and slightly different behavior. See “License Install File notes (pre-AX-3.3)” on page 1-33 for more details.

Click a license or certificate to select it, or *double-click* to view in a dialog, as shown in Figure 1-43.

Figure 1-43 Viewing a license in License Manager



Buttons below each side let you install (import) a *new* license or certificate file. Typically, license files are imported from either the online licensing server or from your local license database.

If selected, you can also view or delete an *existing* file (**View** button is the same as simply double-clicking item, see Figure 1-43), or save (export) a license file as a “license archive” (.lar) file.



Caution Do not delete an existing license or certificate without specific reason, as you will likely render the JACE inoperable until a proper license or certificate is reinstalled!

For further [License Manager](#) details, see the following sections:

- [License operations \(AX-3.3 and later\)](#)
- [License Install File notes \(pre-AX-3.3\)](#)
- [About the licensing server](#) (applies to all NiagaraAX revisions)

License operations (AX-3.3 and later)

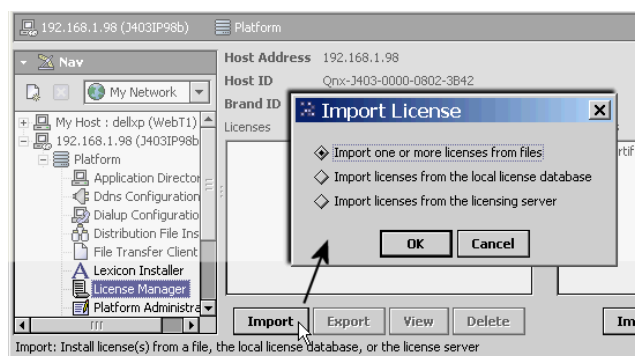
Starting in AX-3.3, below the *license-side* of the [License Manager](#) (Figure 1-42), these two buttons (commands) are displayed in addition to **View** and **Delete**:

- **Import** — Always available, this provides various options for installing a license file from local files, from the [licensing server](#), or from your “local license database.”
- **Export** — Available if you have a license selected, to save locally as a “license archive file.”

Import

If you choose **Import** from the License Manager (in AX-3.3 or later), the Import License dialog asks you to select where the source license is, as shown in Figure 1-44.

Figure 1-44 Import dialog from License Manager (AX-3.3 or later)



Select *one* of the following options (depending on scenarios, some may be unavailable, as noted):

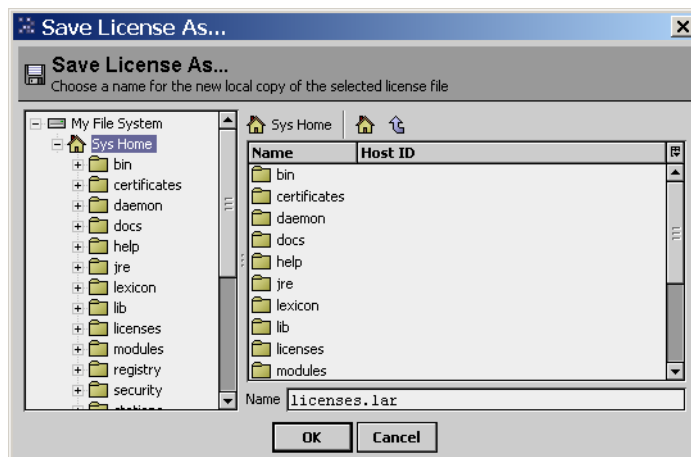
Note: See “[License Import results](#)” on page 1-33 for details on results after making a selection below.

- **Import one or more licenses from files**
Always an available option, this provides a **Select File** dialog in which you can navigate to either a source license archive (.lar) file or an unzipped license file. When you select a license or license archive file, an attempt is made to install the license in the host platform.
- **Import licenses from the local license database**
This option will be unavailable (dim) if this host's license file is *not* in your local license database, or if the license in your local license database already *matches* the currently installed license. With this option selected, the license is immediately installed in the remote host platform. See “[About the local license database](#)” on page A-6 for related details.
- **Import licenses from the licensing server**
Typically, this option is available if your Workbench PC has Internet connectivity. When you select this option, Workbench silently searches the licensing server and installs the license.

Export

With a license selected in the [License Manager](#) in AX-3.3 or later, the **Export** button provides a **Save License As...** dialog to save that license file locally on your Workbench PC, as a *license archive* (.lar) file. See [Figure 1-45](#). For related details, see “[About license archive \(.lar\) files](#)” on page A-7.

Figure 1-45 Save License As dialog in AX-3.3 and later (save as .lar file)

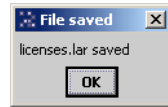


Note: **Export** is similar to the **Save** button in the pre-AX-3.3 License Manager, but with the advantage of the license archive format, preserving the uniquely named “host ID” subfolder path. You can use the License Manager’s [Import](#) command to install any exported license archive, or the equivalent [Import File](#) command in the Workbench License Manager view of Workbench.

By default, a license archive file is saved in the root of your Niagara release directory. If needed, you can use the dialog’s navigation controls to specify another target folder or drive. Before saving, you can also *rename* the license archive file, to make it more identifiable. For example, instead of: `licenses.lar`, you could rename it `MyJaceNxs.lar`.

After exporting a license, a notification dialog appears in Workbench, as shown in [Figure 1-46](#).

Figure 1-46 Exported license archive notification dialog



License Import results

Depending on the [Import](#) option chosen in the **License Manager** ([Figure 1-44](#)) and the success of the import attempt, after you click **OK**, one of several dialogs may appear to signal completion, as follows:

- **Licensing Complete**
The license was successfully added, as shown in [Figure 1-47](#).

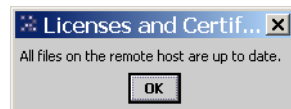
Figure 1-47 Licensing Complete dialog



Note: If a station is running on the host platform, this dialog informs you that the host must be rebooted (if a QNX-based platform) or station restarted (if Win32-based platform) to become effective, and provides a **Yes** button to do this now. Or, you can select **No** and do this manually later.

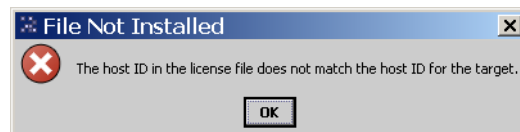
- **Licenses and Certificates Already Current**
The license currently installed on the host already matches the source license (whether specifying any of the license import options). A dialog appears as shown in [Figure 1-48](#).

Figure 1-48 License and Certificates Already Current



- **File Not Installed**
No appropriate license (by host ID) was found in either the license file or the license archive specified when importing by file, noted with a dialog similar to [Figure 1-49](#).

Figure 1-49 File Not Installed



- **(License Request Form, in browser)**
If importing from the license server, and an existing license was not found for this host platform, a separate window (of your default browser) opens with a license request form, showing the host ID for this host. See [Figure 1-56](#) on page 36.

License Install File notes (pre-AX-3.3)

Prior to release AX-3.3, the License Manager has an “**Install File**” button you can use to install an unzipped license file (with .license file extension). See [Figure 1-50](#).

Figure 1-50 License Manager (pre-AX-3.3) has Install File and Save buttons instead of Import and Export

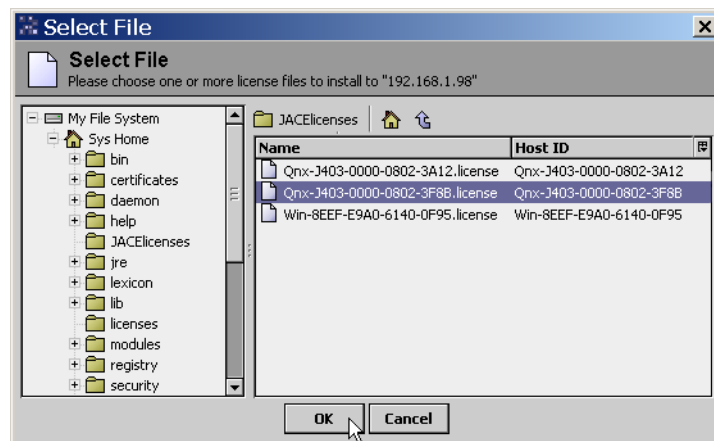


Prior to the AX-3.3 (and later) structured “local license database,” license files on your Workbench PC may be stored in various folder and subfolder locations, often under your Niagara release directory (system home, or “!”). Among the license files, only one is actually required for your Workbench operation—named *brand.license* (for example, *Vykon.license*), in your *!\licenses* folder.

Other licenses are typically for remote JACE controllers. To store these, it is recommended that you create a separate subfolder, for example “JACElicenses”. You may be emailed a JACE license file, or you can use the [Save](#) command in the pre-AX-3.3 License Manager (with a platform connection open to a JACE) to save a local copy. See “[Other license operations \(pre-AX-3.3\)](#)” on page 1-34 for details.

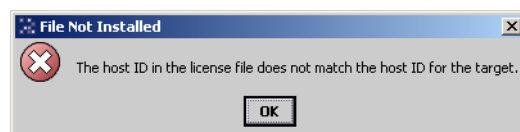
[Figure 1-51](#) shows an example “Select File” dialog produced from an **Install File**, showing three license files, with one being selected.

Figure 1-51 Select File dialog in License Manager



Regardless of how many licenses you have, Workbench automatically *prevents* installation of an incorrect license in a host. For example, if the license selected for installation in [Figure 1-51](#) was for a different JACE (instead of the JACE platform currently open), it does not install. Instead, you see a “File Not Installed” dialog explaining a host ID mismatch ([Figure 1-52](#)).

Figure 1-52 Error dialog explaining license file was for another host



Other license operations (pre-AX-3.3)

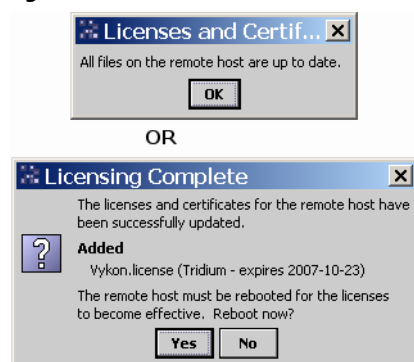
Below the *license-side* of the pre-AX-3.3 License Manager ([Figure 1-50](#)), these two buttons (commands) are displayed in addition to **Install File**, **View**, and **Delete**:

- **Get Online** — Available only if Workbench has detected your PC has Internet connectivity; this retrieves and installs license files from the [licensing server](#).
- **Save** — Available if you have a license selected, to save a local copy on your Workbench PC.

Get Online If you choose **Get Online** from the pre-AX-3.3 License Manager, Workbench silently searches the licensing server for a valid license for this platform host. When found, a dialog explains what operation was performed ([Figure 1-53](#)), such as:

- All files on the remote host are up to date.
- Added (with details)

Figure 1-53 Possible Get Online results from License Server

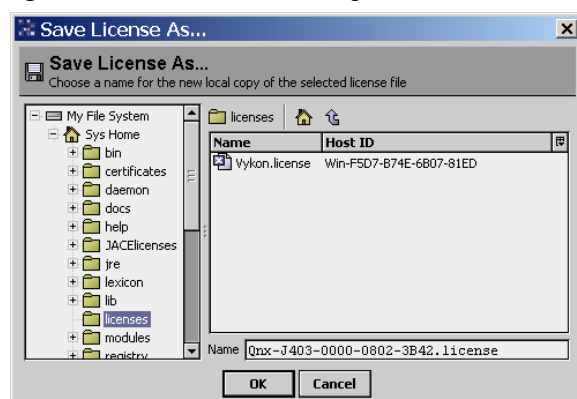


For additional details, see “About the licensing server” on page 1-35.

Save With a license selected in the License Manager, the **Save** button provides a dialog to save that license file locally on your Workbench PC, as shown in Figure 1-54.

Note: You might save JACE licenses for emergency restore purposes, or you might be requested for a particular license, for update purposes. In the latter case, once you received the updated (and digitally-signed) license back, and you could use the **Install File** command to reinstall.

Figure 1-54 Save License As dialog



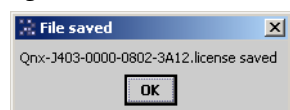
By default, the license file is saved in the licenses folder under your Niagara release directory. If needed, you can use the dialog’s navigation controls to specify another target folder or drive.

Before saving, you can also *rename* the license file, to make it more identifiable. For example, instead of: Qnx-J403-0000-0802-3A12.license, you could rename it J403-Bldg1.license.

Note: Name of a license file can be freely changed, however any edit of the license file contents breaks its digital signing, and will render the license inoperable!

After saving a license file, a notification dialog appears in Workbench, as shown in Figure 1-55.

Figure 1-55 Saved license notification dialog



About the licensing server

For license files validated against the Tridium certificate, installation can be automated from Workbench. All such purchased licenses (including JACEs, AxSupervisor, or Workstation-only) are stored and available to Workbench through the online *licensing server*.

Note: The licensing server is the final license authority—the most current version of any NiagaraAX host platform's license is always stored there. In addition to accessing licenses via the licensing server when using the License Manager, operations in other AX-3.3 views access the licensing server too. Examples include the [Workbench License Manager](#) tool of Workbench, or the Network License Summary view of the “Licenses” slot of the NiagaraNetwork's ProvisioningExt.

Providing that your PC currently has Internet connectivity while running a platform connection to any Niagara host, the [License Manager](#) allows you to automatically retrieve and install any needed licenses.

- Starting in AX-3.3, do this with the **Import** button, then selecting the license server option. See [“Import”](#) on page 1-31 for details. As a side benefit, your “local license database” is also updated.
- Prior to AX-3.3 (AX-3.2, AX-3.1) do this with the **Get Online** button in the License Manager. See [“Get Online”](#) on page 1-34 for details.

Note: If sourcing from the license server while platform-connected to a host that has not yet been assigned a license by the server (or has a “pending” license), a license request form opens in your computer's default browser, as shown in [Figure 1-56](#).

Figure 1-56 Submit License (browser) dialog for unlicensed platform.

This lets you submit a license request to the licensing server that includes the platform's Niagara Host ID. In this dialog, be sure to enter your name, email address, and other purchase related information for the license ([Figure 1-56](#)).

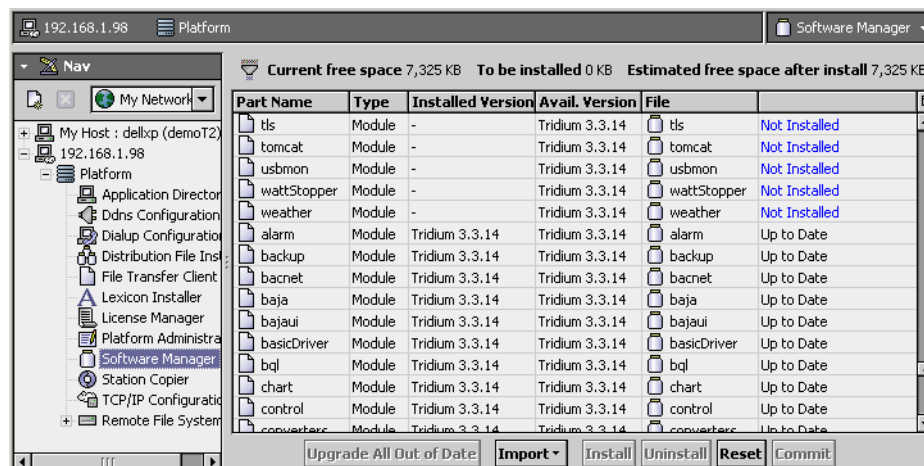
If you already have been sent a “License Key”, note that a pending license already exists on the licensing server. In this case, you can enter it along with the part number to activate that license, and make it immediately available.

Upon approval, the license file for this JACE will be emailed back to the entered address. This license may be either in unzipped form (have a “.license” extension), or be a “license archive” (have a “.lar” extension)—see [“About license archive \(.lar\) files”](#) on page A-7. At that point forward, it will also be available for automatic retrieval using the corresponding “licensing server” operations from various views, such as the [License Manager](#), [Workbench License Manager](#) view, and so forth.

Software Manager

As shown in [Figure 1-57](#), the **Software Manager** is one of several [platform views](#). This view lets you install, upgrade, uninstall, or simply review all software installed in a remote JACE platform, including modules and dist files. The view compares the platform's installables against your “locally available” software, meaning the most current Niagara modules and dist files in the [software database](#) on your Workbench PC.

Figure 1-57 Software Manager compares remotely installed software to locally available



Initially, the Software Manager rebuilds the software list, reflecting your available software. The standard “software database” under your Niagara build directory (<NiagaraRel>\sw) is always used as your Niagara software database.

The following sections explain further:

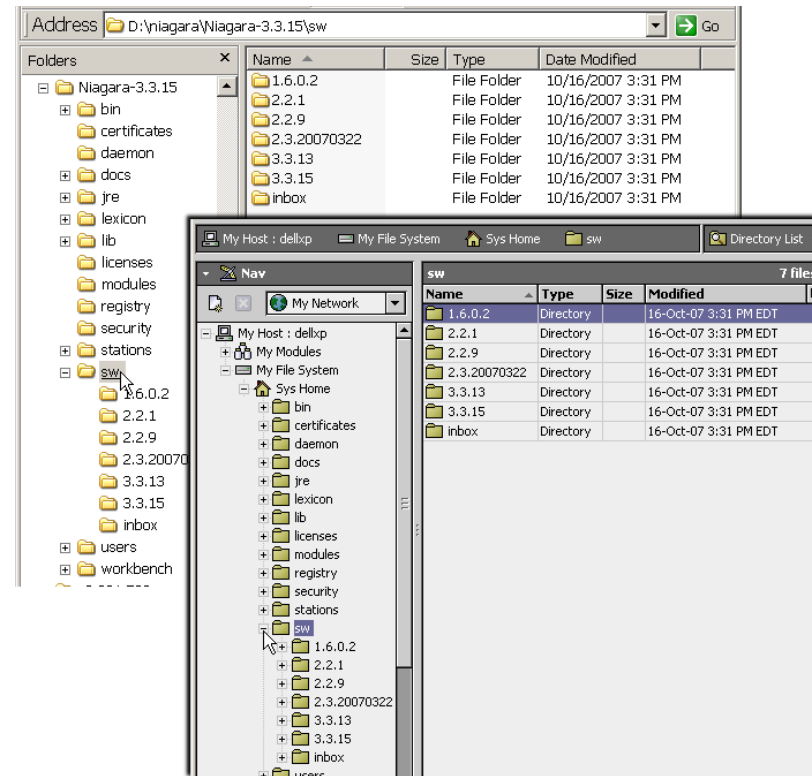
- [About your software database](#)
- [Default software listing and layout](#)
- [Filtering displayed software](#)
- [Software actions](#)

About your software database

The software database for your Niagara Workbench is located under the Niagara build folders’s “sw” subdirectory. If Workbench was installed using the “use as an installation tool” option, this directory will contain a several subdirectories, each named using “version numbers”.

You can see your sw subdirectory structure using either Windows Explorer or the local “My File System” in Workbench, as shown in [Figure 1-58](#).

Figure 1-58 Software database is everything under sw



Using the [Figure 1-58](#) example, this sw software database has the following versioned subdirectories:

- 1.6.0.2 — Reflects version of the Sun JVM for Win32 hosts. Contains 1 “sun-jre” dist file.
- 2.2.1 — Reflects version “photon-x86” dist for future platform support. Contains 1 dist file.
- 2.2.9 — Reflects version of QNX operating system. Contains 12 different qnx dist files.
- 2.3.20070322 — Reflects version of the IBM J9 JVM used on QNX-based hosts. Contains an “ibm-j9” dist file.
- 3.3.13 — Reflects a previous *Niagara build number*. Contains several Niagara nre “clean” dist files, installed by the “installation tool” Workbench installation option.
- 3.3.15 — Reflects the current *Niagara build number*. Contains numerous Niagara nre “config” and “core” dist files, installed by the “installation tool” Workbench installation option. Also, after the [Software Manager](#) is first used, contents of the build’s modules directory (module .jars) are copied here too.
- inbox — Provides a means for you to copy any installable file here, and have the [Software Manager](#) automatically create a proper “versioned” subdirectory for it. Or, if the correct subdirectory already exists, the Software Manager will copy the inbox file(s) there.

Note: Numbers of subdirectories and version number names in your sw subdirectories will be different, this is only a simple example. Do not manually create or rename subdirectories in this area for proper operation—instead, let the Software Manager administer this database.

As an equivalent to the inbox feature, you can use the **Import** button at the bottom of the [Software Manager](#) to add to your Workbench software database. For details, see “[Software Import](#)” on page 1-41.

When you add different-versioned installable files, the number of different subdirectories under your sw directory will continue to increase. However, the Software Manager monitors and displays only the most recent installable file for each software item.

Note: Even though older software files (modules, dists) are not selectable in the Software Manager, they can be useful in the software database when restoring a backup dist for a JACE using a previous software release. You use the [Distribution File Installer](#) to restore a backup.

Default software listing and layout

By default, the [Software Manager](#) lists all *out-of-date* software at the *top* of the table, then *uninstalled* software, and lastly *up-to-date* software (sorted alphabetically inside each group); see [Figure 1-59](#).

- **Out of Date** software is older than what you have in your PC software database.

- **Not Installed** software does not exist on the platform, but is in your PC software database.
- **Up to Date** software is the same (or possibly newer) than that in your PC software database.

Figure 1-59 Software Manager default listing out-of-date, then uninstalled modules

Part Name	Type	Installed Version	Avail. Version	File	Status
web	Module	Tridium 3.3.14	Tridium 3.3.15	web	Out of Date
wiresheet	Module	Tridium 3.3.14	Tridium 3.3.15	wiresheet	Out of Date
workbench	Module	Tridium 3.3.14	Tridium 3.3.15	workbench	Out of Date
aaphp	Module	-	Tridium 3.3.15	aaphp	Not Installed
aapup	Module	-	Tridium 3.3.15	aapup	Not Installed
alarmRdb	Module	-	Tridium 3.3.15	alarmRdb	Not Installed
andoverAC256	Module	-	Tridium 3.3.15	andoverAC25	Not Installed
andoverInfinity	Module	-	Tridium 3.3.15	andoverInfini	Not Installed
bacnetws	Module	-	Tridium 3.3.15	bacnetws	Not Installed
batchJob	Module	-	Tridium 3.3.15	batchJob	Not Installed
crypto	Module	-	Tridium 3.3.15	crypto	Not Installed
csmgrbase	Module	-	Tridium 3.3.15	csmgrbase	Not Installed
demoAppliance	Module	-	Tridium 3.3.15	demoApplanc	Not Installed
devDriver	Module	-	Tridium 3.3.15	devDriver	Not Installed

As needed, you can scroll down the table or click on headers of [table columns](#) to resort alphabetically.

Software Manager table columns

The [Software Manager](#) lists software using six columns, from left-to-right labeled as follows:

- **Part Name** — The name used to refer to the software (from its manifest), or that is identified by the platform daemon for some other type of part.
- **Type** — Module, or (for a some entries) another type (Arch, Model, NRE, OS, Other, VM).
Note: Only modules, NRE, OS, and VM can be managed, other types appear simply for dependency information. Further, only modules can be uninstalled.
- **Installed Version** — Version of the software installed in the remote host, or blank if not installed. For related details, see [“About module versions”](#) in the *User Guide*.
- **Avail. Version** — Version of locally available software, or blank if the software is on the remote host only.
- **File** — File name of locally available software file, or blank if the software is on the remote host only.
- **<unlabeled>** — *Status* of the software in the remote JACE platform. For each module, status is one of the following:
 - **Not Installed** — Software is not in remote platform, but is available locally. Blue text is used for Not Installed status.
 - **Up to Date** — Software is installed in the remote platform, and is equal to (or higher) than locally available software version.
 - **Out of Date** — Software is installed in remote platform, and is *older* than your local version. Red text is used for Out of Date status.
 - **Not Available Locally** — Software installed in remote platform is not in your software database.
 - **Cannot Install** — Local software is unreadable or has a bad manifest; you cannot install it.
 - **Bad Target** — Remotely installed software is unreadable or has a bad manifest, and is therefore unusable by a station. Software in this state should probably be fixed, since it could cause the station to not work correctly.
 - **Downgrade to <version>** — Remotely installed software is intended to be replaced with a module having a lower version.
 - **Install <version>** — Software is intended to be installed; it does not currently exist on the remote platform.
 - **Re-Install <version>** — Remotely installed module is intended to be replaced with a module having the same version.
 - **Uninstall <version>** — Remotely installed module is intended to be uninstalled.
 - **Upgrade to <version>** — Remotely installed module is intended to be replaced with a module having a higher version.

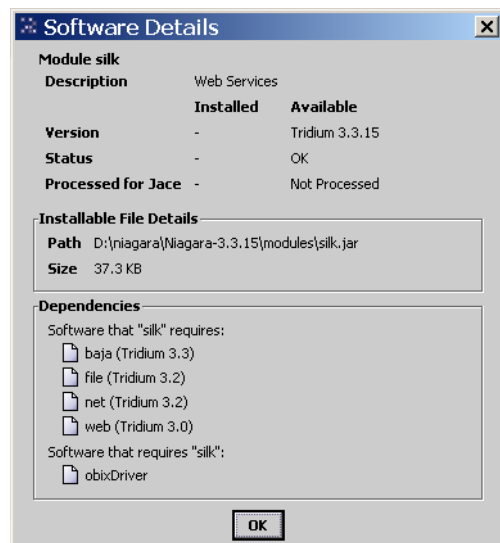
Note: “Intended” status values like “Install <version>” reflect un-committed actions made during your Software Manager session. Blue text is used to list these statuses.

You can also view [software details](#) about any item in the table. In addition, you can filter (reduce) the number of software items listed, based on text included in part name or the softwares' status values. See [“Filtering displayed software”](#) on page 1-40 for more details.

Software Details

From the [Software Manager](#), double-click any listed item to see a dialog with details ([Figure 1-60](#)).

Figure 1-60 Software Details dialog from Software Manager



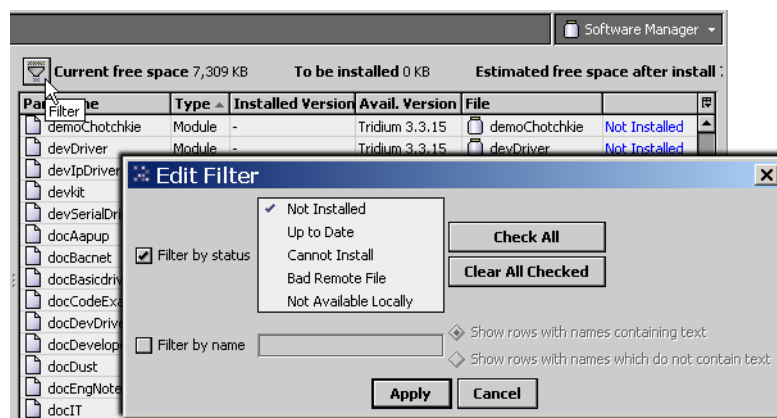
Details include a brief module description, comparisons between installed and available module, module file and size, and whatever module *dependencies* exist, by part names. Dependencies are listed for both cases: what part(s) are required *by* this module, plus parts dependent *on* this module.

Note: *From a practical viewpoint, dependency details are for information only. When installing modules from the Software Manager, all dependent modules are automatically included when you select a module to install.*

Filtering displayed software

By default, the [Software Manager](#) lists all remotely installed and locally available software items, which can produce a very large table. A filter control provides an Edit Filter dialog ([Figure 1-61](#)), in which you select items for listing, thereby filtering undesired items.

Figure 1-61 Filter control and dialog to limit displayed modules



You can use either [Filter by status](#) or [Filter by name](#), or a combination of the two.

Filter by status

Software items with “Out of Date” status always appear in the [Software Manager](#). So do any with uncommitted (intended) status values, such as “Install,” “Uninstall,” and so on.

When you enable filter by status, you can *check* other statuses *to include* (or clear to omit) the listing of associated items in the table, as follows:

- Not Installed — Software on your PC, but not in the remote platform.
- Up to Date — Software on your PC *and* in the remote platform, where the software is not older.
- Cannot Install — Local software is unreadable or has bad manifest, you cannot install it.
- Bad File — Remote software is unreadable or has bad manifest.

Note: With status filtering enabled, you can also simply “check all” and “clear all checked.”

- If all status items are cleared, only “Out of Date” and uncommitted status modules appear.
- If all status items are checked, the display is similar to disabled status filtering, except “non-module” items are not listed.

Filter by name

Name filtering lets you include *or* exclude items based on character string portion of Part Name. When enabled (checked), you can *type* in a string of characters, and then check one of the following:

- Show rows with names containing text — Only items with part name containing this string.
- Show rows with names which do not contain text — Only items with part name that does not contain this string.

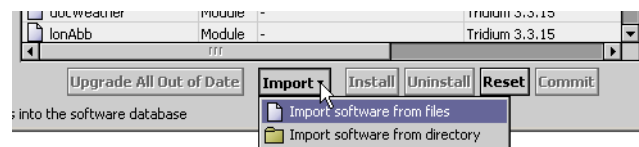
This feature can be useful to filter many modules with common name characters, for example “lon” or “doc” part-named modules.

Software Import

As shown in [Figure 1-62](#), an **Import** button at the bottom of the [Software Manager](#) provides two menu choices for you to add new installable software files (module .jars, .dists) in your [software database](#).

Note: Also see the next section, “[Import vs. copy into modules](#)”.

Figure 1-62 Import choices to bring in file(s) or entire folders



The two import options are:

- **Import software from files**
This produces the standard **File Chooser** dialog, in which you navigate to the proper location and select one or more software files for import.
- **Import software from directory**
This produces the standard **Directory Chooser** dialog, in which you navigate to the proper location and select a directory, for inclusion of any contained software files. For example, you might do this for an older installed build of Niagara, selecting its “sw” folder, or a portion thereof.

Upon import, the software list is again rebuilt by the Software Manager (popup dialogs appear briefly). Afterwards, any software items that are newer-versioned, or that did not previously exist, will now be represented in the software table.

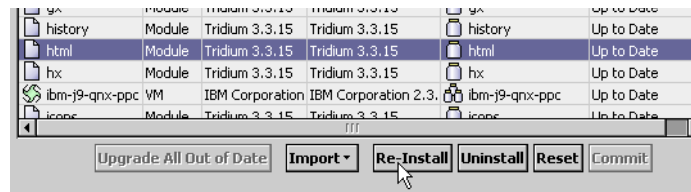
Import vs. copy into modules

When receiving updated or new *module* jar files (say sent from Systems Engineering or downloaded from Niagara Central), you have two basic options when copying them to your Workbench PC, as follows:

1. Copy directly into your `<niagaraRel>\modules` directory. This makes the module(s) available in your Workbench environment, and also available to install in other remote platforms (when the installer runs, the module(s) are also copied into your [software database](#), available for installation). This is the typical choice.
2. Copy into your `<niagaraRel>\sw\inbox` directory (or, use the equivalent “[Software Import](#)” feature in the [Software Manager](#)). In this case, the module(s) are *not* used in your Workbench environment, but *are* available in your software database for installation in remote platforms. This would be the choice where you want to keep using a newer (or older) version of the received module(s) in your Workbench environment. A scenario that fits here, is if you received *older* versions of modules, perhaps needed to restore an older backup dist file in a certain remote platform.

Software actions

As needed, from the [Software Manager](#) you can take actions on software items, such as install, uninstall, upgrade, and so forth. You flag intended actions on software items using *action buttons* near the bottom of the manager’s view pane, as shown in [Figure 1-63](#). Action buttons typically become enabled when you have one or more items selected.

Figure 1-63 Software Manager action buttons

Included in action buttons are **Reset** and **Commit**. When you reset, all flagged software changes (since the last commit) are cleared. Commit is how you actually *launch* the flagged changes, and station running on that platform is stopped. Then, after the software operation completes, the host is rebooted (if a **QNX-based** JACE), or if a **Win32-based** JACE, its station is restarted.

Note: Before committing software actions, make sure that controlled equipment that might be adversely affected by the JACE's station stopping and then host rebooting (from software changes) is put in a manually controlled state.

The following action buttons are explained in further detail:

- [Upgrade All Out of Date](#)
- [Install](#)
- [Uninstall](#)
- [Re-Install, Upgrade, Downgrade](#)
- [Commit and Reset](#)

Upgrade All Out of Date

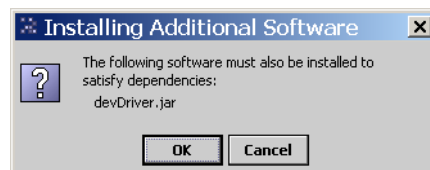
Whenever one or more local items are newer than in the opened JACE platform, the **Software Manager** enables an **Upgrade All Out of Date** button. This allows you to flag *all* out-of-date software to be upgraded. Unlike other action buttons, specific item(s) do not need selection first.

When you click it, the status of all out-of-date software changes to "Upgrade to <version>," and the button becomes unavailable. If needed, you can still make additional changes, such as choosing new software items to install.

Install

This button is available in the **Software Manager** when you have one or more modules selected with a status of "Not Installed." When you click it, the status of the selected modules changes to "Install <version>," and the button changes to **Cancel Install**.

Note: If a selected module has dependencies on modules not already installed (or also flagged to install), a dialog appears explaining additional software is needed, as shown in [Figure 1-64](#). After you click **OK** from this dialog, the additional modules are flagged, the status of all affected modules changes to "Install <version>," and the button changes to **Cancel Install**.

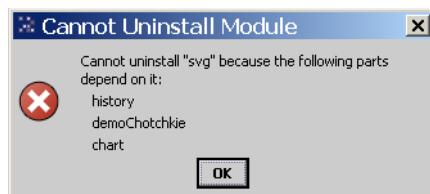
Figure 1-64 Installing Additional Software dialog

Uninstall

This button is available in the **Software Manager** when you have one or more *installed* modules selected (status of either "Up to Date" or "Out of Date"). If the selected module(s) are not dependencies of other installed modules, when you click Uninstall the module(s) status changes to "Uninstall <version>," and the button changes to **Cancel Uninstall**.

Note: If other installed modules have dependencies on one or more modules you selected, a dialog appears explaining the uninstall cannot occur, as shown in [Figure 1-65](#). You can then decide if you want to reflag another uninstall, selecting also all modules that are dependent.

Figure 1-65 Cannot Uninstall dialog



Re-Install, Upgrade, Downgrade

In the [Software Manager](#), when you have one or more *installed* software items selected, the “install” button changes to show one of these options.

- **Re-Install** appears if the installed item is the same version as your locally available one.
- **Upgrade** appears if the installed item is an earlier version than your locally available one.
- **Downgrade** appears if the installed item is a newer version than your locally available one.

When you click this button, the software’s status correspondingly changes to either “Re-Install <version>”, “Upgrade <version>”, or “Downgrade <version>”, and the button changes to **Cancel <action>**, for example: **Cancel Re-Install**.

Commit and Reset

In the [Software Manager](#), when you have one or more *pending* actions in place on software items, the **Commit** button is available. This is how you initiate the software action.

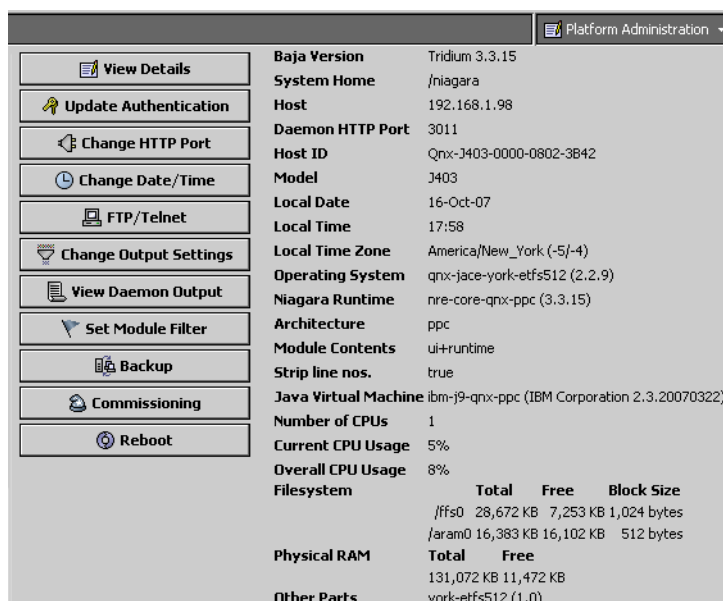
At any time before you commit, you can also click the **Reset** button. This removes all pending actions in place on software items, and makes the **Commit** button unavailable again.

Platform Administration

Platform Administration is one of several [platform views](#). This view provides access to various platform daemon (and host) settings and summary information. As shown in [Figure 1-66](#), available functions appear as “buttons” on the left side, and summary information is listed in the right side. Typical use is when commissioning a new JACE, or when troubleshooting platform or host problems.

Note: During a platform connection, upon first access to Platform Administration, a small delay occurs while downloading data about that platform’s installed modules. You may briefly see a “Loading Modules” dialog before the main view ([Figure 1-66](#)) appears.

Figure 1-66 Platform Administration view



Note: Some functions vary by platform, see [“About platform differences”](#) on page 1-4.

The following sections provide more details:

- [Types of Platform Administration functions](#)

- [View Details](#)
- [Update Authentication](#)
- [Change HTTP Port](#)
- [Change Date/Time](#)
- [FTP/Telnet](#)
- [Change Output Settings](#)
- [View Daemon Output](#)
- [Set Module Filter](#)
- [Backup](#)
- [Commissioning](#)
- [Reboot](#)



Caution *Reboot does exactly what it says, regardless of the opened platform. This is a drastic action to take on any Niagara host. See “[Reboot](#)” on page 1-53 for more details.*

Types of Platform Administration functions

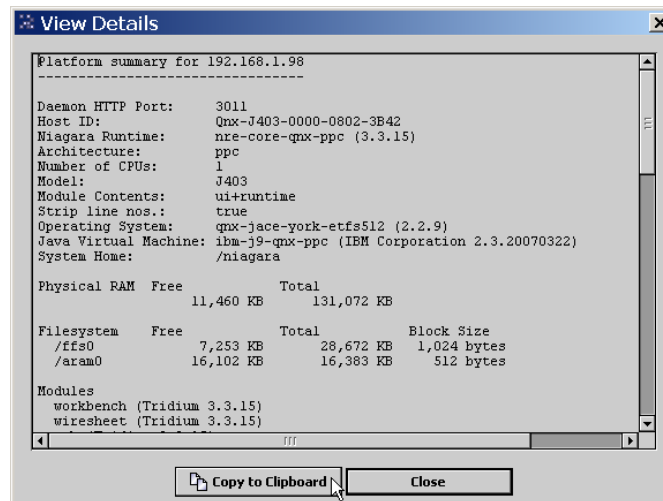
The following list summarizes platform administration functions, by button in the view:

- [View Details](#)
Provides platform summary data, available to the Windows clipboard. Includes all summary information shown in main [Platform Administration](#) view, plus installed modules, and so on.
- [Update Authentication](#)
For dialogs to change *platform login access* (user name and password). In [QNX-based](#) platforms this is simple, as there is only *one* platform administrator. [Win32-based](#) platforms offer a choice of a single (*digest*) platform account, or use of Windows OS user accounts (*basic* authentication).
- [Change HTTP Port](#)
For a dialog to change the HTTP port for the host’s platform daemon from (default) port 3011 to some other port.
- [Change Date/Time](#)
For a dialog to change the hosts’s current date, time, and time zone, as used by that host’s OS.
- [FTP/Telnet](#)
([QNX-based](#) only) For a dialog to enable/disable both FTP and Telnet access to the JACE, or change the default port number used by each one.
- [Change Output Settings](#)
Provides a dialog to change the log level of different processes that can appear in the platform daemon output.
- [View Daemon Output](#)
Provides a window in which you can observe debug messages from the platform daemon in real time, including the ability to pause.
- [Set Module Filter](#)
Provides a dialog to change the module content level of the host. Used very infrequently.
- [Backup](#)
Make a complete backup of all configuration on the connected host platform, including all station files as well as other Niagara configuration.
- [Commissioning](#)
One way to launch the Commissioning Wizard, as an alternative to right-clicking on Platform in the Nav tree.
- [Reboot](#)
Provides a method to reboot a JACE platform, which restarts all software including the OS and JVM, then the platform daemon, then (if so configured in the [Application Director](#)) the installed station. If you click this, a confirmation dialog appears. If you answer yes, the JACE is rebooted and the platform connection drops.

View Details

This selection from the main [Platform Administration](#) view lists more platform information than shown in the main view. Included in the **View Details** window ([Figure 1-67](#)) is all installed modules, lexicons, licenses, and certificates, in addition to all stations (each station line lists configuration for autostart and autorestart, plus current status).

Figure 1-67 View Details dialog in Platform Administration



Generally, information in this view is helpful when troubleshooting or asking for technical support. Buttons include:

- **Copy to Clipboard** puts all details in the dialog on your PC's Windows clipboard.
- **Close** exits the dialog, same as Windows close control (contents copied remain on clipboard).

Update Authentication

This selection from the main [Platform Administration](#) view lets you change that platform's authentication. This affects the login used to access the host's platform daemon. Depending on the type of platform currently opened ([QNX-based](#) or [Win32-based](#)), update authentication provides different dialogs, as follows:

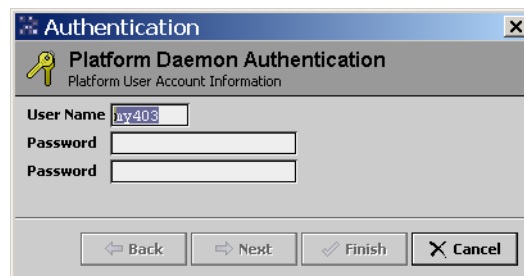
- QNX-based platforms: [Digest platform authentication](#)
- Win32-based platforms, either:
 - [Basic platform authentication](#) or
 - [Digest platform authentication](#)

Digest platform authentication

Digest platform authentication is the only method for a [QNX-based](#) host, and an alternative for a [Win32-based](#) host. The associated Authentication dialog lets you change the *single* platform account *credentials* (user name and password), as shown in [Figure 1-68](#).

Note: Credentials are case-sensitive. For example, *MyJACE4* and *MyJace4* are not the same.

Figure 1-68 Platform authentication dialog for digest authentication



The following sections provide more details:

- [User Name](#)
- [Password](#)
- [Usage Notes](#)

User Name In digest authentication, platform user name can be as follows:

- If [QNX-based](#) host, a maximum of 8 alphanumeric characters (a - z, A - Z, 0 - 9), where the first character must be alphabetic, and following characters either alphanumeric or underscore (_).
Note: Starting in AX-3.2, the maximum limit increased from 8 to 14 alphanumeric characters.
- If [Win32-based](#) host, any number of alphanumeric characters, including hyphens and underscores.

Password In digest authentication, platform password for both QNX-based and Win32-based hosts can be any combination of alphanumeric characters, including common punctuation (! @ # \$ %). This permits a “strong password,” if desired.

Usage Notes In digest authentication, when changing credentials (user name or password, or both), your new credentials become immediately effective when you click **Finish**.

If you previously had “Remember these credentials,” selected (the default) in the Authentication login dialog, the cached credentials are automatically updated. For related details, see “[Credentials manager](#)” in the *User Guide*.

Basic platform authentication

A [Win32-based](#) platform can use either digest *or* basic (native Windows OS user based) authentication for Niagara platform access.

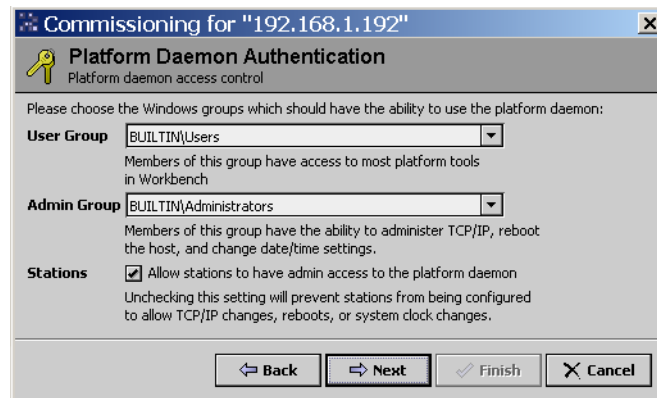
- [Digest platform authentication](#) provides good protection against password eavesdropping. However, there is *only one level* of platform login access, using a single platform user account.
- Basic platform authentication provides integration with existing Windows installations, and provides two [levels of platform access](#). However, it does not protect against password eavesdropping. For any Win32-based host, including a JACE-NX, when you update platform authentication, a dialog asks you to select one of the two methods, as shown in [Figure 1-69](#).

Figure 1-69 Authentication dialog for Win32 Niagara host



- If you select digest authentication, upon **Next** you go to the authentication dialog to set the single platform login account ([Figure 1-68](#) on page 45). There is no linkage between Windows OS users accounts and the platform administrator.
- If you select basic authentication, upon **Next** you go to a different dialog ([Figure 1-70](#)) where you can assign one existing Windows user group to each of the two possible [levels of platform access](#).

Figure 1-70 Basic platform authentication dialog, group selection



In addition, a “**Stations**” checkbox allows you to disable *any* station user from changing TCP/IP settings, system time, or rebooting the host by accessing the station’s [PlatformServices](#).

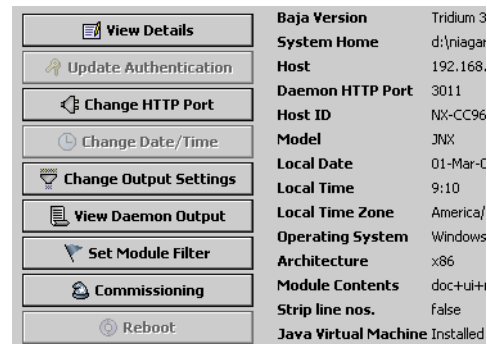
Note: In general, if a [Win32-based](#) JACE, you should leave the **Stations** checkbox enabled, as shown in [Figure 1-70](#). However, if an *AxSupervisor* (PC) platform, you may wish to clear the **Stations** checkbox, particularly if the local IT department has host access concerns.

Levels of platform access [Basic platform authentication](#) provides *two* levels of platform access, which are determined by a user’s group membership(s). The levels of platform access are:

- **User**
Platform access at this level allows full use of most Workbench platform views. This includes the ability to install or delete licenses and stations (including the one running), also to install, re-install, or upgrade the platform dist file and/or modules, and to start, re-start, or stop a station.
- **Admin**
Full access. This includes all user-level platform operations, plus the ability to configure host TCP/IP settings and dialup configuration, change platform authentication, change platform daemon HTTP port, change host date/time settings, use the [File Transfer Client](#), and reboot the host.

Note: When platform-connected at the user level (vs. admin), some platform views are read only. This includes views for [Dialup Configuration](#), [TCP/IP Configuration](#), and [User Manager](#). In addition, some [Platform Administration](#) view buttons are unavailable, as shown in [Figure 1-71](#).

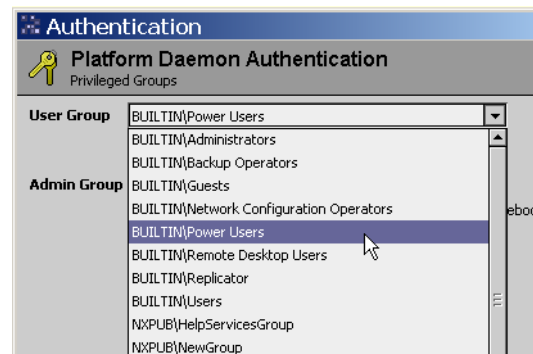
Figure 1-71 Platform Administration view if user-level platform login



Platform access to a remote [Win32-based](#) host (JACE-NX) also provides a [User Manager](#) view in which you can manage Windows users and groups local to that host.

Privileged group selections For both platform access levels (user and admin), you can select from a full list of user groups known to Windows on that host, as shown in [Figure 1-72](#).

Figure 1-72 Group selections include Windows built-in user groups



Groups include Windows “built-in” user groups (include “BUILTIN” or “NT AUTHORITY” prefix), as well as any locally-defined user groups. If the remote host has been added to a Windows *domain*, groups defined in that domain are also listed and available.

If a user has membership in *both* assigned Windows user groups, upon successful platform login they have admin-level platform access.

Note: Default group selections for a Niagara Win32 installation (either Workbench/AxSupervisor installation or a factory-shipped JACE-NXS/NX) are as follows:

- User Group — BUILTIN/Users
- Admin Group — BUILTIN/Administrators

Change HTTP Port

This selection from the main [Platform Administration](#) view lets you change the HTTP port monitored by the host’s platform daemon. By default, HTTP port 3011 is monitored for platform client connections. This is a different port than any used for station (Fox) connections. For more details, see [“About a platform connection”](#) on page 1-2.

If needed, you can change the daemon monitored port to another HTTP port. You may choose to do this for specific firewall reasons, or perhaps for additional security. As shown in [Figure 1-73](#), you can type in the new port number in the Port field, which enables the **OK** button.

**Caution**

If there is a firewall on the host or its network, before changing this port make sure that it will allow traffic to the new port. For JACE-NXS related details, see the section [“Review the Windows XP Firewall”](#) in the *JACE-NXS NiagaraAX Install and Startup Guide*.

Figure 1-73 Update Platform Daemon HTTP Port dialog



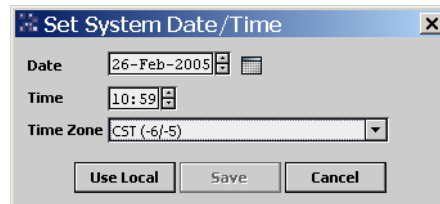
When you click **OK**, the platform daemon restarts, and you lose your platform connection (note this does not affect the operation of any running station). The platform icon is ghosted in the Nav tree, showing the new HTTP port number (:n) in parenthesis. Double-click the ghosted platform for the platform login dialog.

Note: Before closing the host (removing it from the Nav tree), carefully note the new (non-default) port number you entered. You must always specify that port number whenever reopening this platform. If the host is running AX-3.1 or later, you can check this port number in a station running on the host, by opening its Config, [PlatformServices](#) property sheet.

Change Date/Time

This selection from the main [Platform Administration](#) view lets you change the date and time in the Niagara platform, as well as specify its time zone ([Figure 1-74](#)). The Save button becomes available after you change one or more fields in the dialog, or when you click [Use Local](#).

Figure 1-74 Set System Date/Time dialog



Upon **Save**, any change is processed by that host's operating system.

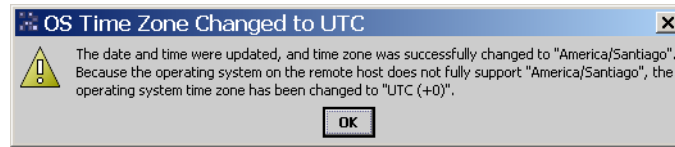
The three dialog fields are as follows:

- **Date**
Click in a day-month-year position to select, then click up/down controls, or click and type in numerals directly, or click the calendar icon for a popup dialog to select the date from a calendar.
- **Time**
Always displays in 24-hour format. Click in a hour or minute position to select, then click up/down controls, or click and type in numerals directly.
- **Time Zone**
Provides a drop-down selection list of all available time zones in NiagaraAX. Each time zone provides a text description, and in parenthesis the “hour offset” from UTC (and if daylight savings time is used) the “offset plus daylight savings.” For example: America/New_York (-5 , -4).
For more time zone details, see the appendix [“Time Zones and NiagaraAX”](#) on page B-1.

Note: Although rare, in some cases a platform's OS may not support a particular time zone. This relates only to a few time zones with daylight savings switch-over. For example, America/Santiago (-4 , -3) is not supported in Win32-based hosts because the start and end daylight savings dates are exact dates, versus a week number and weekday (e.g. 2nd Sunday).

Upon clicking **Save** for a time zone not fully supported by the host's OS, a popup dialog ([Figure 1-75](#)) explains that the platform's OS time zone was set to UTC (+0) during the update.

Figure 1-75 OS time zone changed to UTC dialog



Use Local

Typically, if your Workbench PC's current date/time setting are accurate, you click the "Use Local" to synchronize the remote host's date, time, and time zone with your Workbench PC. Upon **Save**, the remote host will have the identical settings.

Note: To keep time synchronized across multiple Niagara platforms, configure the [TimeSyncService](#) in the station running on each platform, as appropriate.

FTP/Telnet

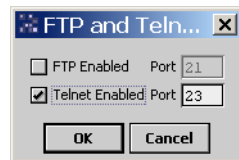
This [Platform Administration](#) view selection is available for QNX-based JACE only. (For Win32-based platforms, you configure/enable FTP and Telnet within local Windows TCP/IP configuration.)

As factory-shipped, a QNX-based JACE has the FTP and Telnet service disabled — this is recommended, especially if the platform is exposed to the public Internet. However, in some cases you may wish to enable one or both services, perhaps to facilitate debugging.

You can also change the TCP/IP port used by each service from the "well-known" port to some other port. However, be sure that any firewalls being used on your network will allow traffic to that port.

[Figure 1-76](#) shows the FTP and Telnet dialog with Telnet enabled (both services showing "well-known" ports).

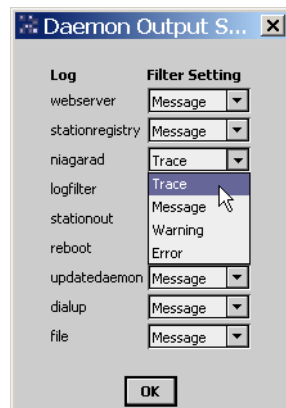
Figure 1-76 FTP and Telnet dialog



Change Output Settings

This selection from the main [Platform Administration](#) view lets you "tune" the amount and content of the platform daemon output ([View Daemon Output](#)). You can do this by changing the [log filter settings](#) of the various daemon processes ([Figure 1-77](#)).

Figure 1-77 Daemon Output Settings dialog



By default, all daemon processes have a "Message" log filter level, and include the following:

- webservice — HTTP server for platform client connection
- stationregistry — Daemon management of stations, including startup, shutdown, and watchdog actions.
- niagarad — Main platform daemon process
- logfilter — Tracks changes made in this dialog

- stationout — Daemon process to station
- reboot — Daemon process
- updatedaemon — Daemon process
- dialup — Daemon process to dialup daemon
- file — Daemon file transfers

Log filter settings

For any item, use the **Filter Setting** drop-down to select one of the following:

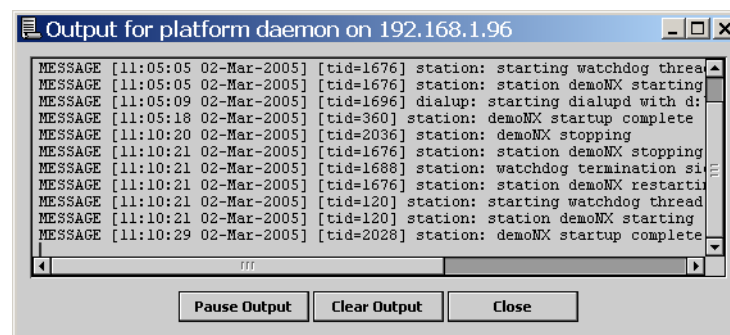
- **Trace**
Returns all message activity (*verbose*). This includes all transactional messages, which may result in too many messages to be useful. Be careful using Trace!
- **Message**
(Default) Returns informational “MESSAGE”s, plus all “ERROR” and “WARNING” types.
- **Warning**
Returns only “ERROR” and “WARNING” type messages (no informational “MESSAGE”s).
- **Error**
Returns only “ERROR” type messages (no “WARNING” or informational “MESSAGE”s).

View Daemon Output

This selection from the main [Platform Administration](#) view lets you examine standard output from the host's *platform daemon* ([Figure 1-78](#)), in *real time*. It is available for troubleshooting purposes.

Note: *Output is different from the output of a running station, as seen in the [Application Director](#).*

Figure 1-78 Example Output for platform daemon



Depending on the [log filter settings](#) set in platform administration's **Daemon Output Settings** dialog ([Change Output Settings](#)), the activity level in the output window will vary. Output is “non-modal,” meaning that you can leave this window open and still do other Workbench operations (including change output settings).

As needed, use the scroll bars to navigate through messages, which will have headings “TRACE,” “MESSAGE,” “WARNING,” or “ERROR,” depending on message type. Each message includes a timestamp and a thread id number.

Use the Windows copy shortcut (CTRL + C) to copy text of interest to the Windows clipboard.

Click **Pause Output** to freeze the output from updating further (no longer in real time). When you do that, note that the button changes to **Load Output**. This means that daemon messages are still collected. When you click **Load Output**, the display loads the collected messages and continues again in real time.

Click **Clear Output** to clear all collected messages from the current daemon output window. This not a “destructive clear,” as another (or new) daemon output window retains daemon messages.

Set Module Filter

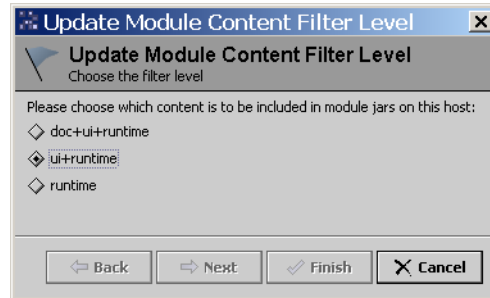
This selection from the [Platform Administration](#) view lets you globally change the “module content level” of the connected platform. This determines how much file space is consumed by installed Niagara modules.

For any [Win32-based](#) host (providing it has hard drive for file storage), you may want the *fullest* possible content level, meaning including all documentation (doc+ui+runtime).

For a [QNX-based](#) JACE, with more limited flash-based file storage, you may wish to change its module content level. Clicking this button produces the **Update Module Content Filter** dialog ([Figure 1-79](#)).

Note: Typically, you specify the module content level once during initial JACE commissioning, then never change it again.

Figure 1-79 Update Module Content Filter Level dialog



Module content level is *one* of the following, from largest to smallest:

- doc+ui+runtime — Typically appropriate *only* for [Win32-based](#) platforms.
- ui+runtime — Appropriate if the JACE is to run a Web Service.
- runtime — Typically best for any [QNX-based](#) platform *not* running Web Service.

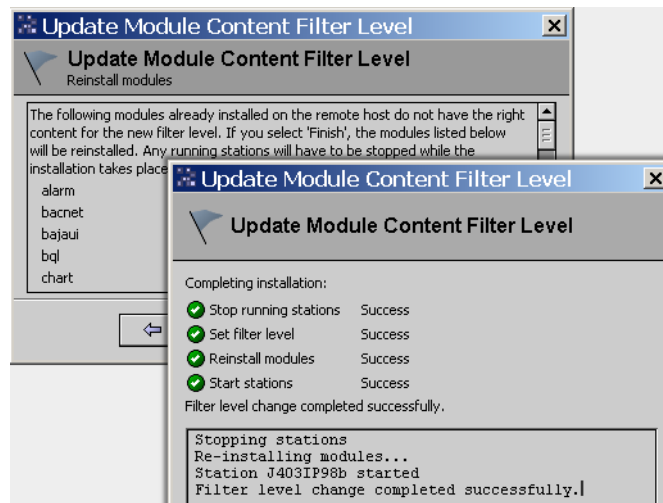
[Operations from a change in module content level](#) differ according to the “direction” of the change.

Operations from a change in module content level

Depending on how you change the module content filter level, operations on the platform vary:

- If you *restrict* the content level (say, go from “ui+runtime” to “runtime”), modules already installed are *not* automatically re-installed (to reduce storage). You simply click the **Finish** button to close the dialog, and platform/station operation is otherwise unaffected. However, if you later re-install existing modules, or install new modules, the new content filter level is applied—typically with resulting savings in storage space.
Therefore, if “freeing” storage space is the goal when restricting module content, after changing the content level, you should *re-install* existing modules. Do this using the [Software Manager](#).
- If you *increase* the content level (say, from “runtime” to “ui+runtime”), this typically requires modules to be re-installed in that platform. In this case, the dialog provides a **Next** button and explains that this automatically occurs, with the station first stopped as a result, as shown in [Figure 1-80](#). Note that if a [QNX-based](#) platform, any module installation also automatically results in a *reboot* of that host platform.

Figure 1-80 Dialog messages resulting from increasing module content level



Backup

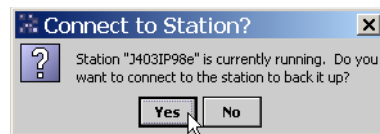
This selection from the [Platform Administration](#) view performs a complete backup of the connected JACE, saved as a *dist file* on your PC. The backup dist contains the entire station folder plus the specific NRE config used by that JACE platform, including license(s) and certificate(s). The dist also contains pointers to the appropriate NRE core, Java VM, modules, and OS. If ever needed, you restore a backup dist using the platform [Distribution File Installer](#) view.

Note: *The backup dist file also contains the TCP/IP configuration of the host when it is backed up. When restoring the backup, you can select to restore these settings, or retain the TCP/IP settings currently in use by the target host. See [“Restoring a backup dist”](#) on page 1-25.*

You can perform a backup with a station running on the target host, or when no station is running.

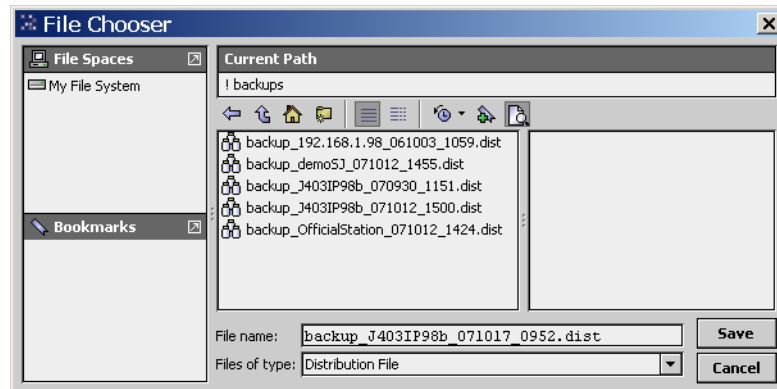
- If the JACE is running station, a confirmation dialog appears to connect to it ([Figure 1-81](#)). This routine uses that station’s **BackupService** to perform an “online backup.” (If the station is not already open in Workbench, you must then logon as a station user.)
- If no station is running on the JACE, the platform daemon performs its own “offline backup.”

Figure 1-81 Backup with station running, station connection



After connection to the station (or if no station is running), the **File Chooser** appears ([Figure 1-82](#)) for you to navigate to the target location to save the backup dist file, and to rename if desired.

Figure 1-82 File Chooser to select target folder and dist file name



By default, the Backup function automatically creates (if not already present) a `backups` subdirectory under your Niagara build directory. The default *name* for a backup file uses a format of:
`backup_stationName_YYMMDD_HHMM.dist`

For example, “`backup_J403IP98b_071012_1500.dist`” for a backup made of station “J403IP98b” on October 12, 2007 at 3:00 pm.

After you click **Save** the backup starts.

- If the station is running, a Fox Backup job is performed. A notification popup appears in the lower right of your display when the backup is done. This job is recorded in the station’s **BackupService** and visible in that component’s **BackupManager** view. Details are also available by accessing the job in the station’s **Job Service Manager**.
- If doing an “offline backup” (no station running), the platform daemon provides another progress dialog during the backup to a dist file, as shown in [Figure 1-83](#). Upon completion, you can click **Close** to return to the [Platform Administration](#) view, or click **Details** to see another popup with a log of actions performed in the backup ([Figure 1-84](#)).

Figure 1-83 Backup from Platform Administration, no station running

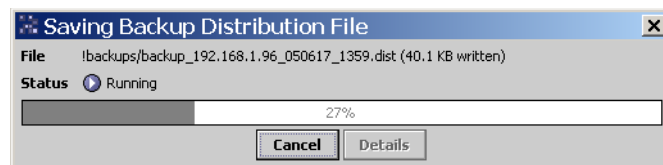
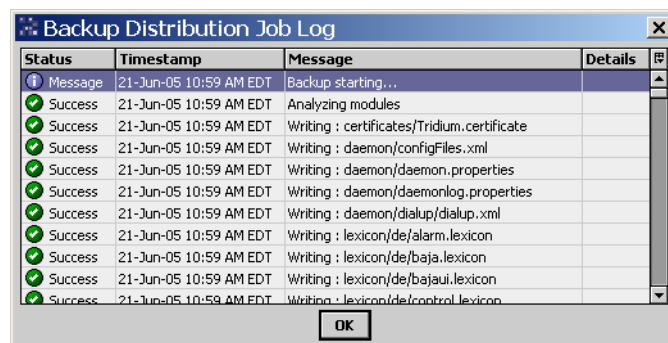


Figure 1-84 Available Details from backup using platform daemon (no station running)



Commissioning

This selection from the [Platform Administration](#) view launches the Commissioning Wizard, an ordered sequence of various platform views. Included views are a subset of those listed in “[Types of platform views](#)” on page 1-3.

Typically, you use the Commissioning Wizard for the *initial* Niagara installation and startup of a JACE controller, as well as for any future *upgrade*. For more details, see “[About the Commissioning Wizard](#)” in the *JACE NiagaraAX Install & Startup Guide*.

Note: The Commissioning Wizard is intended for a remote JACE only. Please note that this button is unavailable whenever you are connected to your “localhost” (AxSupervisor) platform.

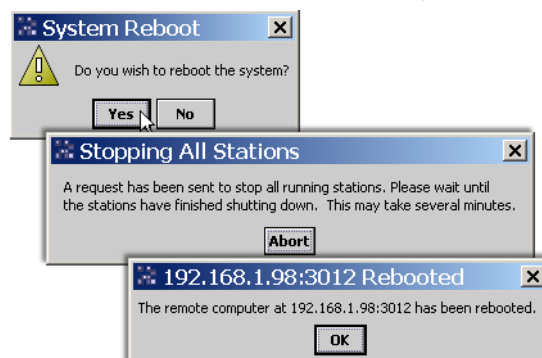
Reboot

This selection from the [Platform Administration](#) view *reboots the host* of the connected platform. One confirmation dialog appears, after which the daemon attempts to stop any running station before issuing the final reboot ([Figure 1-85](#)).



Caution For any [Win32-based host](#), never use reboot in place of restart station (from Application Director), unless there is a specific need for it! Reboot is a drastic action to take on any Niagara host.

Figure 1-85 Reboot performs operating system reboot



A reboot restarts the host OS, Java VM, platform daemon, and finally the Niagara station (providing that it is configured to “Auto-restart,” see “[Application Director](#)” on page 1-7).

When the platform reboots, your Workbench platform connection to it is dropped. Depending on the platform type, it may take from several seconds to a couple of minutes before you can connect again.

Note: *Reboot is intended for a remote JACE only. Please note that this button is unavailable whenever you are connected to your “localhost” (AxSupervisor) platform.*

Station Copier

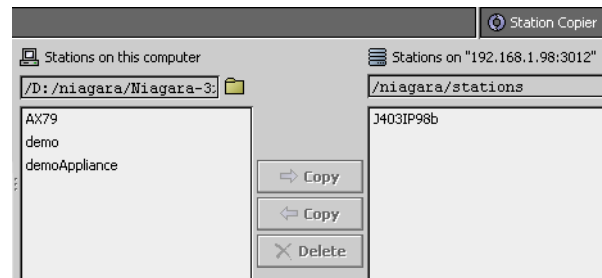
The Station Copier is one of several [platform views](#). You use it to *install a station* in a remote platform, as well as make a *local backup* of a remote station (copy to your PC). You can also delete stations, either locally or remotely.

Note: *In AX-3.1 or later, you do not see this platform view when opening a local platform connection at your AxSupervisor PC—usage has always been intended for remote JACE hosts only.*

As shown in [Figure 1-86](#), this view is split into two main areas:

- Local stations on your PC (left side)
- Remote stations on the opened platform (right side)

Figure 1-86 Station Copier view



Note: *By default, contents of the <NiagaraRel>\stations folder is shown on the PC (left) side. If you have station folders located elsewhere, you can click the folder icon for a “Change Directory” dialog, and point the Station Copier to that location. That alternate location is remembered the next time you access the Station Copier.*

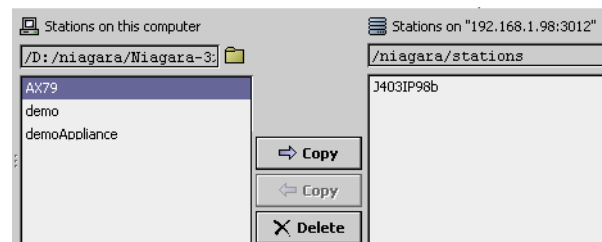
The following sections provide more details:

- [Station copy direction](#)
- [Station Transfer Wizard](#)
- [Deleting stations](#)

Station copy direction

The copier works in either direction. In other words, click a station on *one* side (to copy to the other side). When you click a station, the station is selected (highlighted) and the appropriate **Copy** button, by direction, becomes enabled to clarify the source and target. See [Figure 1-87](#).

Figure 1-87 Copy direction by station side selection



To perform the following station operations, you would:

- Click in *left* side for a copy from *local-to-remote*.
This means *installing* a station (is described as “installing” in remaining document sections).
- Click in *right* side for a copy from *remote-to-local*.
This means *backing up* a station (is described as “backing up” in remaining document sections).

When you click a **Copy** button, a [Station Transfer Wizard](#) guides you the rest of the way.

Station Transfer Wizard

This wizard assists with any station copy (installing or backing up) by presenting a number of *steps*. The exact steps vary by the direction of copy, as well your selections in wizard step dialogs. In each step, click **Next** to advance to the next step. As needed, click **Back** to return to a previous step and make changes, or click **Cancel** to exit from the wizard (no station copy performed).

Note: Use **Cancel** if you need to select a different station to copy; this reruns the wizard.

The wizard's **Finish** button is enabled only in the final step. When you click **Finish**, the related operations begin, and you see progress updates in the wizard's **Transferring station** dialog. When complete, you click **Close** in that dialog to exit the wizard.

The following sections describe all possible steps in the Station Transfer Wizard:

- [Name step](#)
- [Delete step](#)
- [Content step](#)
- [Disposition step](#)
- [Station settings step](#)
- [Details step](#)
- [Modules step](#)
- [Stop station step](#)
- [Review step](#)

Note: In the unlikely case where the source station `config.bog` file is currently in use ("locked"), the wizard opens in a state where you must **Cancel** to exit (no other steps are given).

- If installing a station, the source `config.bog` is locked if it contains unsaved changes (it is being edited elsewhere in Workbench). After saving changes, you can try the copy again.
- If backing up a station, the source `config.bog` is locked if currently in process of being saved. You can retry the copy later.

Name step

The first step in the **Station Transfer Wizard** is to confirm the name (or type a new name) for the copied station directory.

Figure 1-88 Station Transfer Wizard dialog, name step



Default name is the station directory being copied. If you rename the station, it will be identical to the source (copied) station in every way *except* name of its station directory.

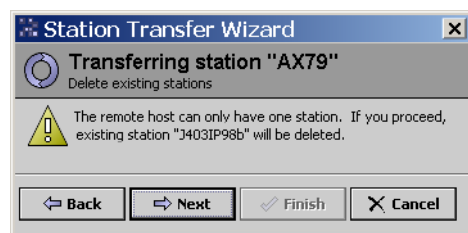
Delete step

Note: This step is skipped for any station backup, or if a station install in either of these cases:

- No existing station exists.
- The existing station is named the same as the one you are installing.

This step occurs because *all* JACE platforms have a support limit of one (1) installed station. The delete step simply cautions you that the existing station will be deleted ([Figure 1-89](#)).

Figure 1-89 Station Transfer Wizard dialog, delete step



Note: The entire remote station directory (all subdirectories and files) is deleted when the station install starts. If unsure, it may be best to **Cancel**, then backup the remote JACE station first.

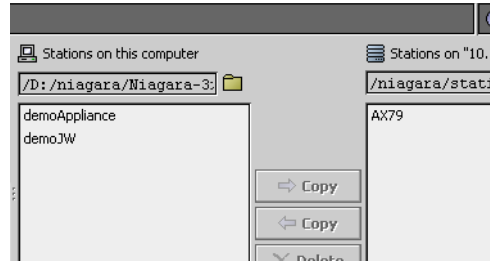
The next step is the [content step](#).

Content step

Note: This [wizard](#) step is skipped if the source station consists of only a `config.bog` file.

After the [name step](#) and possibly [delete step](#), the wizard asks you to select what station files to copy, with the default selection being “all” files and folders under that station directory ([Figure 1-90](#)).

Figure 1-90 Station Transfer Wizard dialog, content step



The three possible selections are:

- Copy files from selected directories (not shown if source station has no subdirectories). If you select this, a later “[details step](#)” allows you to select the source subdirectories.
- Copy every file in the station directory and its subdirectories.
- Copy only the “config.bog” station database file.

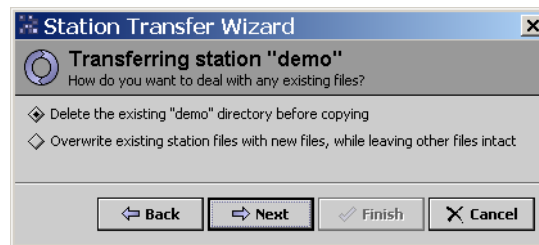
The next step is the [disposition step](#).

Disposition step

Note: This [wizard](#) step occurs only when an identically-named target station already exists.

If the target station already exists, a disposition step asks what is to be done with it ([Figure 1-91](#)).

Figure 1-91 Station Transfer Wizard dialog, disposition step



The two possible selections are:

- Delete existing station directory before starting the copy
- Overwrite existing station files with new files, while leaving other files intact.

If you previously selected “copy everything” from the [content step](#), the default pre-selection is the first (delete existing station directory). Otherwise, the second selection (overwrite) is pre-selected.

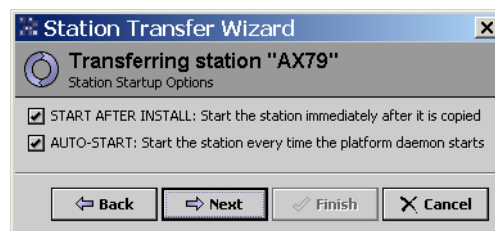
The next step is the [station settings step](#).

Station settings step

Note: This [wizard](#) step is skipped for any station backup.

This step specifies whether to *start* the station after it is copied, also its Auto-Start setting ([Figure 1-92](#)).

Figure 1-92 Station Transfer Wizard dialog, station settings



By default, both items are pre-selected:

- **START AFTER INSTALL:** Start the station immediately after it is copied.
- **AUTO-START:** Start the station every time the platform daemon starts.
(The latter is one of two station settings for any station, as specified in the [Application Director](#) view by using [start checkboxes](#).)

Typically, you enable both and continue to the next step, either the [details step](#) or [modules step](#).

Details step

Note: This [wizard step](#) is skipped unless you selected “copy selected directories” in the [content step](#).

This step provides a tree ([Figure 1-93](#)) to select station subdirectories (folders) to include in the copy. By default, all selectable folders are both expanded and selected, while unselectable folders are not (note that if present, a station’s [history](#) folder is unselectable).

Figure 1-93 Station Transfer Wizard dialog, details step



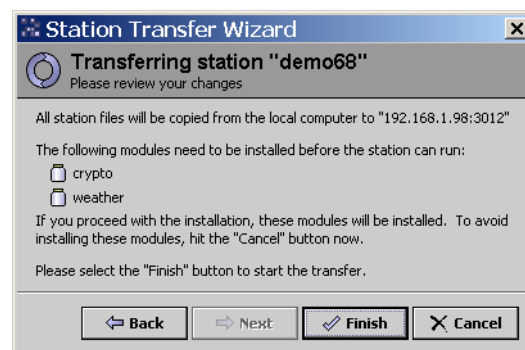
For any selectable folder, click to toggle it as either selected (with X) or unselected (no X).

Modules step

Note: This [wizard step](#) is skipped for any station backup, or when all modules required by the station are already in the remote target platform. In this case, you see either the [stop station step](#) or [review step](#) instead.

This step displays a list of missing modules required for the station to run ([Figure 1-94](#)), and tells you that they will be installed during the station copy operation.

Figure 1-94 Station Transfer Wizard dialog, modules step



If included, this is the *final* step before the copy process starts. At this point, you click either:

- **Finish** — Start the local-to-remote copy, including installation of required modules. Progress updates appear in a [Transferring station](#) dialog.
- **Cancel** — Exit from the [Station Transfer Wizard](#), then rerun selecting another source station.

Stop station step

Note: You see this [wizard step](#) only if installing a “same-named” station, and in a previous [disposition step](#), you selected to “overwrite existing station files.”

This step simply reminds you that the station must be stopped while it is copied ([Figure 1-95](#)).

Figure 1-95 Station Transfer Wizard dialog, stop station step



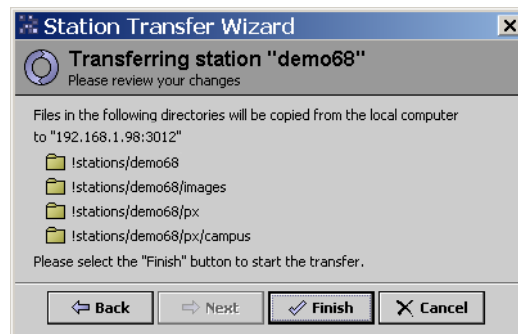
Click Next to go to the [review step](#).

Review step

Note: This [wizard step](#) is skipped when installing a station where additional modules are required. (Instead, the [modules step](#) provides the **Finish** button.)

This step provides a summary of choices from previous steps, and a **Finish** button to begin the station copy process. As shown in [Figure 1-96](#), if you selected only *specific* station subdirectories to copy (from the [details step](#)), they are listed.

Figure 1-96 Station Transfer Wizard dialog, review step

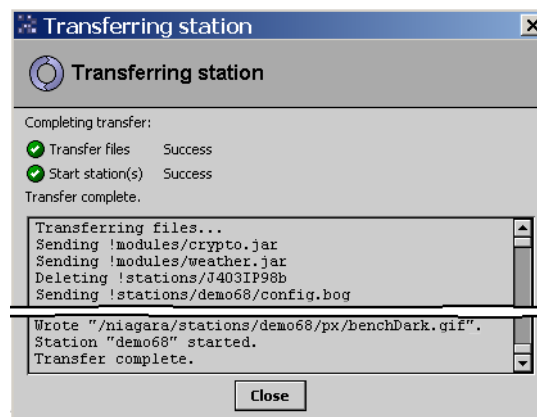


If needed, click **Back** to make changes, or click **Finish** to begin the copy process and observe progress in the [Transferring station](#) dialog.

Transferring station

After clicking **Finish** in the wizard's [modules step](#) or [review step](#), the station copy process begins and updates appear in a this dialog, as shown in [Figure 1-97](#).

Figure 1-97 Station Transfer Wizard, Transferring station (copy) process



Depending on the type of copy, the following operations may be included in this process:

- If installing a station (copy local-to-remote):
 - Stop all stations — whenever modules require to be installed.
 - Stop one station — any JACE where same station is being reinstalled.
 - Delete station(s) — if you chose to delete station in the [disposition step](#), or if a station needs to be deleted to stay under maximum number of stations (only one for any JACE platform)
 - Transfer files — includes station and module files (actual copy portion).
 - Start station — if a station had required to be stopped (module installation), or if you chose to

start the station in the [station settings step](#). Note that if a [QNX-based](#) platform, the process will finish with a host reboot.

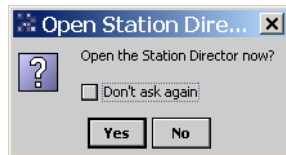
- If backing up a station (copy remote-to-local):
 - Save station — whenever remote station is currently running.
 - Transfer files — includes station and module files (actual copy portion).

Note: A popup explaining that the existing station must be stopped may appear for a few seconds. Following, and during execution of the various operations, a **Cancel** button is available. If you click **Cancel** before all operations complete, the installation (or backup) is not valid.

After all operations are finished, a **Close** button is available and the last update in the dialog is “Transfer complete.” Click **Close** to exit the [wizard](#).

By default, after *installing* a station, the wizard exits with a popup ([Figure 1-98](#)) asking if you wish to switch to the [Application Director](#) platform view. Because it is a good idea to observe a station’s output upon first startup, you typically select **Yes**.

Figure 1-98 Switch to Application Director popup

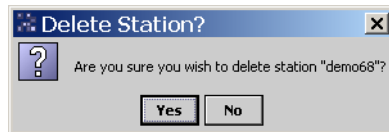


Note: To always automatically switch to the [Application Director](#) after installing a station, click the checkbox to “Don’t ask again” before selecting **Yes**. Then, you do not see this popup again.

Deleting stations

The [Station Copier](#) lets you delete *any* station, either on your local PC (left side) or a remote platform (right side). A confirmation dialog ([Figure 1-99](#)) appears when you select a station and click **Delete**.

Figure 1-99 Confirm delete station dialog



Note: Be careful when deleting stations, as there is no undo. Furthermore, please note the following:

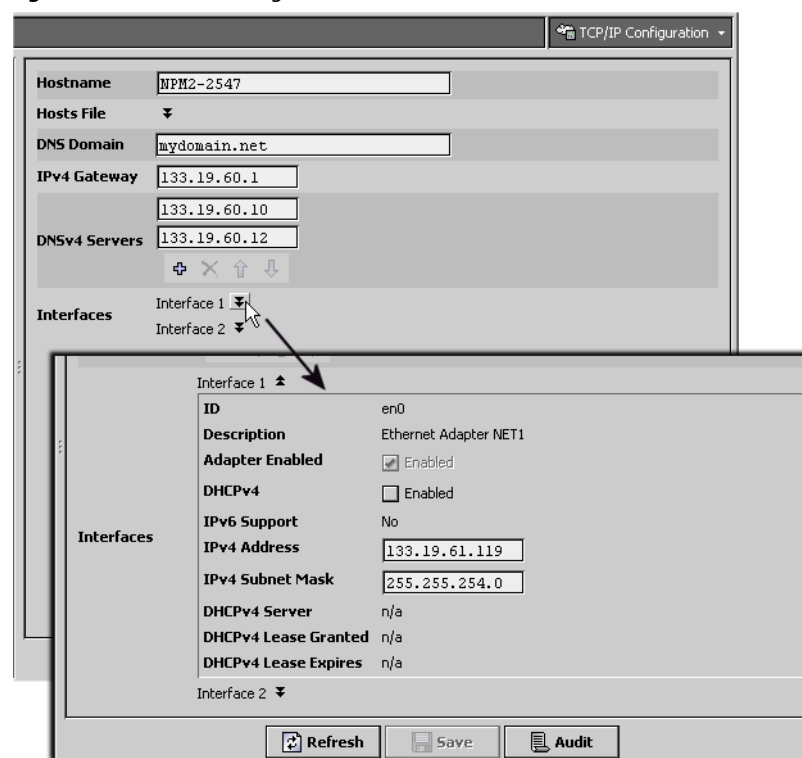
- The entire selected station directory gets deleted, including all subdirectories and file contents.
- Special notification does *not* occur if you choose to delete a *running* station (you may briefly see a “stop station” popup, with opportunity to **Abort**).
- Also in general (as a precaution), *before* deleting a remote station, it is generally recommended to make a backup copy first. If desired, when backing up you can rename it using some “temp” convention to flag it for later housekeeping.

TCP/IP Configuration

TCP/IP Configuration is one of several [platform views](#). Typically, you use it to initially configure a remote JACE’s TCP/IP settings. Some JACE models have only a single Ethernet port (interface) for configuring TCP/IP, but if equipped with two Ethernet ports, both interfaces are accessible ([Figure 1-100](#)).

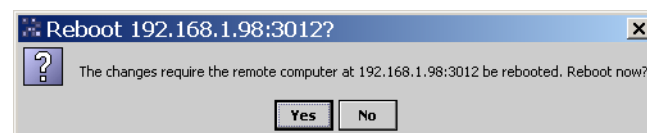
Note: Starting in AX-3.3, *Workbench* shows many TCP/IP property field descriptors with an “IPv4” or “v4” qualifier, in preparation for eventual support for separate “IPv6” properties. Support for IPv6 may be available in the next release of NiagaraAX.

Property descriptions provided in this section are listed with the “v4” descriptor seen in *Workbench* AX-3.3, with the previous descriptor following in parentheses (). For example: IPv4 Address (IP Address)

Figure 1-100 TCP/IP Configuration shows available Ethernet interface(s)

If connected to your localhost (PC) platform, this view also provides access to your PC's Windows TCP/IP settings. However, you typically use the Windows Control Panel for making these changes.

Note: Regardless of platform connection, if you make any changes in this view, a reboot of that platform is necessary before those changes are effective. A popup dialog appears whenever you click **Save** (Figure 1-101), asking if you wish the platform daemon to reboot that host now. When making multiple changes, wait until entering all changes before you click **Save, Yes**.

Figure 1-101 Reboot confirmation dialog from TCP/IP configuration change

More details on the [TCP/IP Configuration](#) view are in the following sections:

- [TCP/IP Host fields](#)
- [TCP/IP DNS fields](#) (host-level for QNX-based platforms only)
- [TCP/IP Interface fields](#)

TCP/IP Host fields

The [TCP/IP Configuration](#) platform view provides access to the platform's TCP/IP host settings. The available host fields are as follows:

- **Hostname**
Synonymous with “computer name,” this is a string that can be processed by a DNS server to resolve to an IP address. By default, hostname for any [QNX-based](#) platform is “localhost”.
Note: On Win32 systems, this is also the computer name. Be aware that changing it will have important consequences with regards to the computer's identification in its workgroup or domain!
If using hostnames in your network, you should assign each Niagara platform a *unique* hostname.
- **Hosts File**
The hosts file is a standard TCP/IP hosts file, where each line associates a specific IP address with a known hostname. To review and edit, you click on the expand control to see all entries.
 - To *add* an entry, click at the end of the last line and press Enter. Then type the IP address, at least one space, then the known hostname.
 - To *delete* an entry, drag to highlight the entire line, then press Backspace.

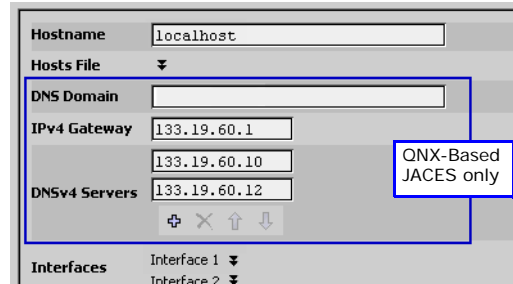
Click the expand control again to collapse the Hosts File editor.

TCP/IP DNS fields

If connected to a [QNX-based](#) platform, the [TCP/IP Configuration](#) platform view provides access to DNS and gateway settings as “host-level” parameters, as shown in [Figure 1-102](#).

Note: If a [Win32-based](#) host, DNS and gateway settings are available under each Interface section. See “[TCP/IP Interface fields](#)” on page 1-61.

Figure 1-102 TCP/IP Configuration for QNX-based platform includes DNS



The available host-level fields for a QNX-based platforms are as follows:

- **DNS Domain**
The TCP/IP Domain Name System (DNS) domain this host belongs to, if used.
- **IPv4 Gateway (Gateway)**
The IP address for the device that forwards packets to other networks or subnets. A valid gateway address is required in multi-station (JACE) jobs to allow point discoveries under NiagaraNetworks.
- **DNSv4 Servers (DNS Servers)**
The IP address for one or more DNS servers (if available), where each can automate associations between hostnames and IP addresses. Included are icon-buttons to Add (to enter IP address of server), Delete, and move Up/Down (to set the DNS search order).

TCP/IP Interface fields

For each enabled Ethernet port on the connected platform, the [TCP/IP Configuration](#) platform view provides an expandable Interface section. Some JACE platforms offer two Ethernet ports (LAN1 and LAN2), while others may have a single Ethernet port. The available TCP/IP Interface fields are as follows:

- **ID**
A read-only OS identifier for the hardware interface, such as “en0” if a [QNX-based](#) platform, or if a [Win32-based](#) platform, a 128-bit GUID (globally unique identifier).
- **Description**
A read-only text string such as “Ethernet Adapter”.
- **Adapter Enabled**
Read-only checkbox that indicates whether the Ethernet port is usable.
- **DHCPv4 (DHCP)**
A checkbox to specify DHCP (Dynamic Host Configuration Protocol) instead of static IP addressing. Successful use requires a DHCP server installed on your network. If enabled, other interface fields such as IP Address and Subnet Mask become read-only, as these are assigned by the DHCP server after the platform reboots.
Note: In general (for stability), static IP addressing is recommended over DHCP.



Caution Do not enable DHCP unless sure that your network has one or more DHCP servers! Otherwise, the JACE may become unreachable over the network.

- **IPv6 Support**
Read-only indicator if host platform supports IPv6 (in initial AX-3.3 release, is typically “No”).
- **DNS Domain**
([Win32-based](#) hosts) The TCP/IP Domain Name System (DNS) domain this host belongs to, if used.
- **IPv4 Address (IP Address)**
The “static” IP address for this host, unique on your network.

Note: If a JACE with two enabled LAN ports, the LAN1 IP address and LAN2 IP address must be on different subnets, otherwise the ports will not function correctly. For example, using a typical “Class C” subnet mask of 255.255.255.0, setting Interface 1=192.168.1.99 and Interface 2=192.168.1.188 is an invalid configuration, as both addresses are on the same subnet.

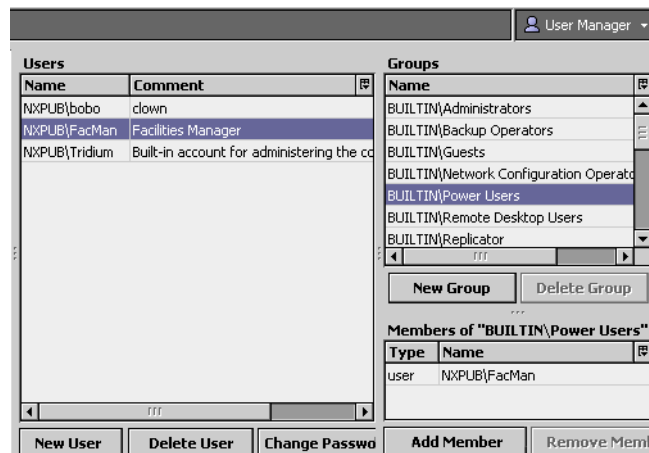
- **IPv4 Gateway (Gateway)**
(Win32-based hosts) IP address for the device that forwards packets to other networks or subnets.
- **IPv4 Subnet Mask (Subnet Mask)**
The “static” IP subnet mask used by this host.
- **DHCPv4 Server (DHCP Server)**
Applies only if DHCP is enabled. Shows read-only address of the DHCP server from which this host last obtained its IP address settings.
- **DHCPv4 Lease Granted (DHCP Lease Granted)**
Applies only if DHCP is enabled. Shows a read-only timestamp of when the DHCP lease started.
- **DHCPv4 Lease Expires (DHCP Lease Expires)**
Applies only if DHCP is enabled. Shows a read-only timestamp of when the DHCP lease will expire, and will need renewal.
- **DNSv4 Servers (DNS Servers)**
(Win32-based hosts) The IP address for one or more DNS servers, each of which can automate associations between hostnames and IP addresses. Included are icon-buttons to Add (to enter IP address of server), Delete, and move Up/Down (to set the DNS search order).

User Manager

The User Manager is one of several [platform views](#), available only when connected to a remote [Win32-based](#) JACE. This view allows you to manage Windows OS user and group accounts local to that host (which otherwise would require accessing “Administrative Tools” in Windows on that host).

Note: You require “admin-level” platform access in order to change any user settings. When connected to the platform using a “user-level” login, you can review settings, but none of the buttons in this view are available, nor are “drag and drop” actions possible. See “[Levels of platform access](#)” on page 1-46 for related details.

Figure 1-103 User Manager for remote Win32-based host



As shown in [Figure 1-103](#) above, the view has two main sides.

- Users are listed in a *users table* on the *left* side.
- Groups are listed in a *groups table* the *right* side. In addition, a lower “membership” table shows all members of any currently selected group.

Buttons below each side provide popup dialogs in which you can add or delete a user or group, or change password for a selected user.

The following sections provide more details:

- [Users management](#)
- [Groups management](#)

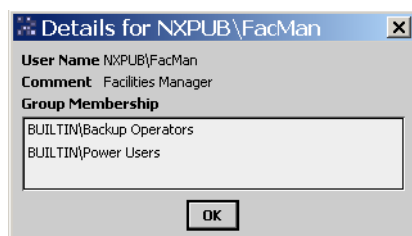
Users management

In the users side of the [User Manager](#), click in the users table and buttons below to perform various Windows user management tasks. You can [review](#), [add](#), and [delete](#) users, and [change passwords](#). You can also [drag and drop](#) users into groups.

Review user

Double-click any existing user in the [User Manager](#) for a Details dialog, as shown in [Figure 1-104](#).

Figure 1-104 Details dialog for Windows user

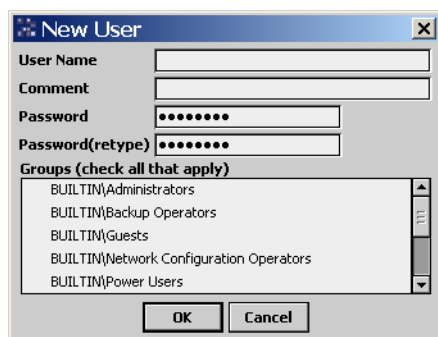


This displays the user's account name, comment, and group memberships (including domain groups).

Add user

Click the **New User** button in the [User Manager](#) for a **New User** dialog, as shown in [Figure 1-105](#).

Figure 1-105 New User dialog



In this dialog you must type a user name and password (text in both password fields must match). You can also type a comment, typically a full user name or description. Click in the groups checklist to designate which groups the new user should have membership.

When you click **OK**, the new user is added and appears in the user table.

Delete user

In the [User Manager](#), click to select one or more users (press Ctrl and click to select multiples). Then click the **Delete User** button for a confirmation dialog, as shown in [Figure 1-106](#).

Figure 1-106 Confirm Delete dialog



When you click **OK**, the selected user(s) is deleted and removed from the user table.

Change user password

In the [User Manager](#), click to select a user, then click the **Change Password** button for a popup dialog ([Figure 1-107](#)).

Figure 1-107 Change Password dialog



You must type the current user's password, then the new password twice (text in *both* new password fields must match). When you click **OK**, the password for that user is changed to your new password.

Drag and drop

In the [User Manager](#), you can “drag and drop” rows from the users table on top of a row in the groups table. This adds the selected user(s) to the target group, without any popup dialog.

Or, if a single group is already selected, you can drag and drop user rows into the lower membership table for that group. This adds the selected user(s) to that group, and updates the membership table.

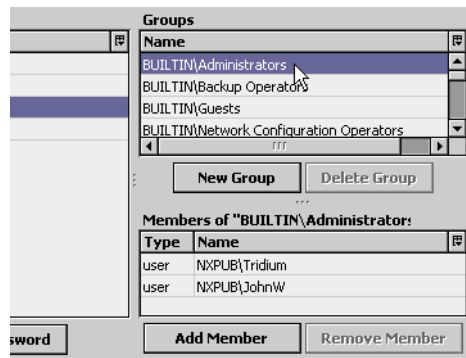
Groups management

In the groups side of the [User Manager](#), click in the groups table, membership table, and buttons below to perform various Windows group management tasks. You can [review](#), [add](#), and [delete](#) groups, and in any group, you can [add](#) or [remove](#) members.

Review group

Click any existing group in the [User Manager](#) to see user members in the table below ([Figure 1-108](#)).

Figure 1-108 Select group to see membership

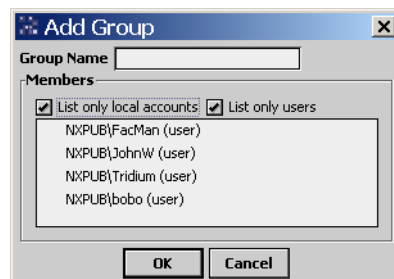


All users for the selected group are shown.

Add group

Click the **New Group** button in the [User Manager](#) for a popup dialog, as shown in [Figure 1-109](#).

Figure 1-109 New Group dialog



In this dialog you must type a name for the new group. Click in the users checklist to designate which Windows users the new group should have as members.

Note: By default, the users checklist is “filtered” to reduce entries as follows:

- **List only local accounts** — Any domain users and groups do not appear.
- **List only users** — Groups do not appear.

As needed, click these checkboxes to add or remove these choices in the users checklist.

When you click **OK**, the new group is added and appears in the groups table.

Delete group

Note: You cannot delete any Windows “Built-In” group.

In the [User Manager](#), click to select one or more groups (press Ctrl and click to select multiples). Then click the **Delete Group** button for a confirmation dialog, as shown in [Figure 1-106](#).

Figure 1-110 Confirm Delete dialog



When you click **OK**, the selected group(s) is deleted and removed from the groups table.

Add member

Select (click) a group in the [User Manager](#), then the **Remove Member** button for a popup ([Figure 1-111](#)).

Figure 1-111 Add Member dialog



Only users not already members of this group are listed. Click in the users checklist to designate which Windows users the group should have as members. By default, as in the **New Group** dialog ([Figure 1-109](#)), the users checklist is filtered to *not list* domain users and groups and local groups. If needed, click these checkboxes to add or remove these choices in the users checklist.

When you click **OK**, the group’s membership is updated with the member(s) you added.

Note: You can also *drag and drop* users (rows in users table) onto groups (rows in groups table).

Remove member

Select (click) a group in the [User Manager](#), then in the membership table, select one or more users. With the user(s) selected, click the **Remove Member** button for confirmation dialog ([Figure 1-112](#)).

Figure 1-112 Confirm Remove dialog



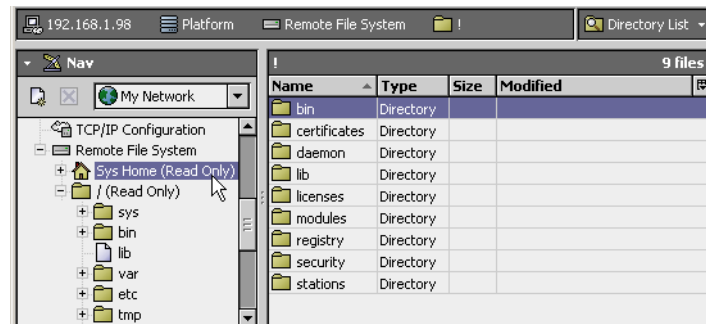
When you click **OK**, the selected user(s) is removed that group’s membership.

Remote File System

The Remote File System view is one of several [platform views](#). It provides a read-only view of the remote platform’s file system.

For [QNX-based](#) platforms (only), it also provides browse capability of the file system root (/) of the remote host, using the Nav tree side bar of Workbench ([Figure 1-113](#)).

Figure 1-113 Remote File System for QNX-based host



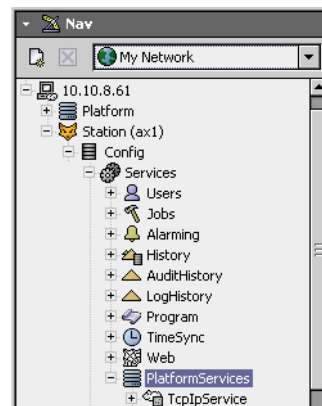
About Platform Services

This section explains the platform access available in a running *station*—in other words, the *station's perspective* on its host platform. Unlike the various [platform views](#), a platform connection is *not needed* to access platform services. Instead, you need only a standard station (Fox) connection.

Platform services do *not* provide the full set of functions available in Workbench as when you have a [platform connection](#). For example, you cannot install or upgrade software, or transfer stations and files. However, a number of important configuration views for the platform are made available under a station's PlatformServices.

Under **Config, Services**, every *running* Niagara station has a **PlatformServices** container ([Figure 1-114](#)).

Figure 1-114 Example JACE station's PlatformServices



Note: When engineering station security, be careful about assigning user permissions to PlatformServices and its child service components. In general, you should regard this portion of the station as most critical, as it allows access to items such as host licenses and TCP/IP settings. Furthermore, right-click actions on the PlatformService include "Restart Station" (note that if a [QNX-based](#) JACE, this results in a host reboot!). For details about station security, see "[About Security](#)" in the User Guide.

PlatformServices is *different from all other components* in a station in the following ways:

- It acts as the *station interface* to specifics about the *host platform* (whether JACE or a PC).
- It is built *dynamically* at station *runtime*—you do not see PlatformServices in an offline station.
- Changes you make to PlatformServices and all child services are *not stored in the station database*. Instead, changes are stored in other files on that platform, such as its `platform.bog` file.

Note: Do not attempt to edit `platform.bog` directly; always use PlatformServices' views!

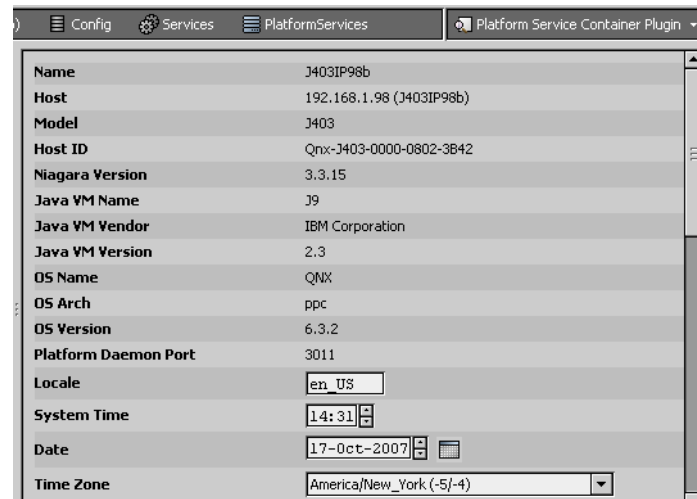
The following sections provide additional details:

- [PlatformServiceContainer parameters](#)
- [PlatformServiceContainer actions](#)
- [SystemService \(under PlatformServices\)](#)
- [Platform service types](#)
- [Using platform services in a station](#)
- [PlatformServices binding and link caveats](#)

PlatformServiceContainer parameters

In addition to acting as a container, the [PlatformServices](#)Container provides various status and configuration entries for the host platform. In the Nav tree, double-click PlatformServices to access the “Platform Service Container Plugin” which lists these entries, as shown in [Figure 1-115](#).

Figure 1-115 PlatformServicesContainerPlugin view (many entries not shown)



Included are many read-only status values as well as configuration parameters. Each is described in separate sections as follows:

- [PlatformServiceContainer status values](#)
- [PlatformServiceContainer configuration parameters](#)
- [Additional PlatformServiceContainer properties \(AX-3.2 and later\)](#)

Note: By default, any [PlatformServices](#)Container also provides three right-click [actions](#).

PlatformServiceContainer status values

Status values in a station's [PlatformServices](#)Container include the following:

- **Name**
Name of running station.
- **Host**
IP address of host platform.
- **Model**
Model of host platform type, such as “J403,” “JNXS,” or “Workstation.” See “[Models of platforms](#)” on page 1-6 for further details.
- **Host ID**
NiagaraAX host identifier, a string unique to this one machine.
- **Niagara Version**
Version and build number of the NiagaraAX distribution running in the host platform.
- **Java VM Name**
Java virtual machine used, e.g. “J9” ([QNX-based](#) hosts) or “Java HotSpot(TM) Client VM.”
- **Java VM Vendor**
Vendor for Java VM, e.g. “IBM Corporation” (J9) or “Sun Microsystems Inc.” (Java HotSpot).
- **Java VM Version**
Version of Java VM, e.g. “2.3” (J9) or “1.6.0_02-b06” (Java Hotspot).
- **OS Name**
Operating System name, such as “QNX” or “Windows XP.”
- **OS Arch.**
Machine architecture for OS, such as “ppc” ([QNX-based](#) hosts) or “x86” ([Win32-based](#) hosts)
- **OS Version**
Operating System version, such as “6.3.2” (QNX) or “5.1” (Windows XP)
- **Platform Daemon Port** (AX-3.1 or later only)
Port number on which the platform daemon that started the station is listening (3011, or another port number). This can prove useful in case you changed the platform port (see “[Change HTTP Port](#)” on page 1-47), but then forgot what the new port is.

Note: In the container plugin, most of the remaining entries are [configuration](#) parameters. However a few status values are also mixed in, and are described below.

- **Number of CPUs**
Number of CPUs used in the host platform (typically 1).
- **Current CPU Usage**
Percentage of CPU utilization in the last second.
- **Overall CPU Usage**
Percentage of CPU utilization since the last reboot.
- **Filesystem**
File storage statistics for the host, including total file space, available (free) space, and file block size (minimum size for even the smallest file). For a [QNX-based](#) host, it may look similar to:

	Total	Free	Block Size
/ffs0	28,672 KB	7,253 KB	1,024 bytes
/aram0	16,383 KB	16,081 KB	512 bytes
- **Physical RAM**
Current total and free RAM statistics for the host. For a [QNX-based](#) host, it may look similar to:

Total	Free
131,072 KB	9,912 KB
- **Hardware Jumper Preset**
(Appears only if [QNX-based](#) host) Either true or false—indicates whether or not the mode jumper is installed for “serial shell mode” access. Read at boot time only. See “[About JACE serial shell mode](#)” in the *JACE NiagaraAX Install & Startup Guide*.
- **EFW Overlays**
(Appears only if a CompactFlash-based JACE-NXS host) Status of the Enhanced Write Filter overlays which protect the CompactFlash card from excessive writes. Two overlays are listed, one for OS, and one for History. The current **state** can either be “Enabled,” “Disabled,” or “Unknown.” The state should always say “Enabled” for both overlays under normal operation. The **Command** should normally be “Commit.” Other possible values are “noCmd” and “CommitAndDisable.” **Ram Usage** is the amount of RAM used by the overlay, and should “cap out” at some maximum number during normal operation. The history overlay theoretical maximum is about 5% larger than the size of the History partition, about 85MB.
Also see the *JACE-NXS NiagaraAX Install & Startup Guide*.

PlatformServiceContainer configuration parameters

Configuration properties of a station's [PlatformServices](#)Container are listed below. If needed, you can change any in the container plugin view (property sheet)—click **Save** to write to the host platform.

Note: It is recommended that you leave engine-related parameters and other advanced settings at default values, unless you have been directed otherwise by Systems Engineering.

- **Locale**
Determines locale-specific behavior such as date and time formatting, and also which lexicons are used. A string entered must use the form: language [“_” country [“_” variant]]. For example, U.S. English is “en_US” and traditional Spanish would be “es_ES_Traditional”. For details, see Sun documentation at <http://java.sun.com/j2se/1.4.2/docs/api/java/util/Locale.html>.
- **System Time**
Current local time in host.
- **Date**
Current local date in host.
- **Time Zone**
Current local time zone for host. For more details, see “[Time Zones and NiagaraAX](#)” on page B-1.
- **Engine Watchdog Policy**
Note: The *engine watchdog* is a platform daemon process, to which the station periodically reports its updated engine cycle count. The watchdog purpose is to detect and deal with a “hung” or “stalled” station, and is automatically enabled when the station starts.
The Engine Watchdog Policy defines the response taken by the platform daemon if it detects a station engine watchdog timeout. Watchdog policy selections include:
 - Log Only — Generates stack dump and logs an error message in the system log.
 - Terminate — (Default) Kills the VM process. If “restart on failure” is enabled for the station (typical), the station is restarted.
 - Reboot — Automatically reboots the host JACE platform. If “auto-start” is enabled for the station, the station is restarted after the system reboots.

- **Engine Watchdog Timeout**
Default is 1 minute, and range is from 0 ms to infinity. If the station's engine cycle count stops changing and/or the station does not report a cycle count to the platform daemon within this defined period, the platform daemon causes the VM to generate a stack dump for diagnostic purposes, then takes the action defined by the Engine Watchdog Policy.
- **Enable Station Auto-Save**
Either Enable (default) or Disable. Allows for "auto save" of running station to "config_backup_<YYMMDD>_<HHMM>.bog" file at the frequency defined in next property. Auto-saved backup files are kept under that station's folder.
- **Station Auto-Save Frequency**
Default is every 24 hours for any QNX-based host, or every (1) hour if Win32-based host. Range is from 1 to many hours.
- **Station Auto-Save Backups to Keep**
Starting in AX-3.1 the default value for QNX-based hosts is 0 (none), and in most cases, this is appropriate for any QNX-based host. Oldest of kept backups is replaced upon next manual save or auto-save backup, once the specified limit is reached. In Win32-based hosts, the default is 3, and typically (unless a CompactFlash-based JACE-NXS) can be safely adjusted up, if desired.
- **RAM Disk Size**
In MB, where default is 16 for JACE-4/5 series, or 8 for JACE-2 series. Specifies size of RAM disk used to store history and alarm files.
- **Failure Reboot Limit**
(Only if an AX-3.1 or later QNX-based host) Limits the number of system reboots that can be triggered by station failures, within the Failure Reboot Limit Period, below (if the host is so configured using the **Application Director**, see "Start checkboxes" on page 1-11). Default value is 3.
- **Failure Reboot Limit Period**
(Only if an AX-3.1 or later QNX-based host) Specifies the repeating frequency of the Failure Reboot Limit period, with default value at 1 hour.
Note: These two "Failure Reboot" settings are also adjustable (in any version of QNX-based host) within that JACE's !daemon/daemon.properties file, in the following two properties:
 - failureRebootLimit=x (where x is integer, default is 3)
 - failureRebootLimitPeriod=y (where y is long in milliseconds, default is 3600000)
- **Minimum Free Heap Size**
Specifies the *minimum* free Java heap size, in MB, against which the station compares (tests) for low memory conditions, that is excessive Java heap. The default value for JACE-4/5s is 8 MB, and for a JACE-2 is 3 MB.
This test automatically runs once a minute. If the heap free byte count is less than the defined minimum free heap size, a "low memory warning" appears in *all Workbench views* of the station. The warning is a yellow message box overlayed on any new view accessed, or on any current view that is refreshed. This warning is removed when the heap free byte count rises above the defined minimum size—such as might occur if enough components are deleted from the station.
Note: All memory statistics, including those for heap, are accessible on a station opened in Workbench, via the Resource Manager view of the Station component.

Additional PlatformServiceContainer properties (AX-3.2 and later)

Starting in AX-3.2, a station running on a QNX-based platform has the following additional PlatformServiceContainer properties, as shown in Figure 1-116.

Figure 1-116 Additional AX-3.2 and later PlatformServiceContainer properties for QNX-based host.

Hardware Jumper Present	false		
Failure Reboot Limit	3 [1 - max]		
Failure Reboot Limit Period	00001h 00m [0ms - +inf]		
RAM Disk	Min Free	Size	Status
	5 % [0 - 100]	16 [1 - max] MB	Ok
Java Heap	Min Free	Max	Free Status
	8 MB	48 MB	37 MB Ok
Open File Descriptors	Min Free	Max Open	Free Status
	50 [50 - max]	2000	1925 Ok
Free RAM	Min Free		Status
	1024 [512 - max] KB		Ok
Disk Space	Min Free		Status
	10 % [0 - 100]		Ok

Refresh Save

These properties are in addition to the other [configuration](#) properties described above, where each has a read-only **Status** field on the far right side, plus the following configuration properties:

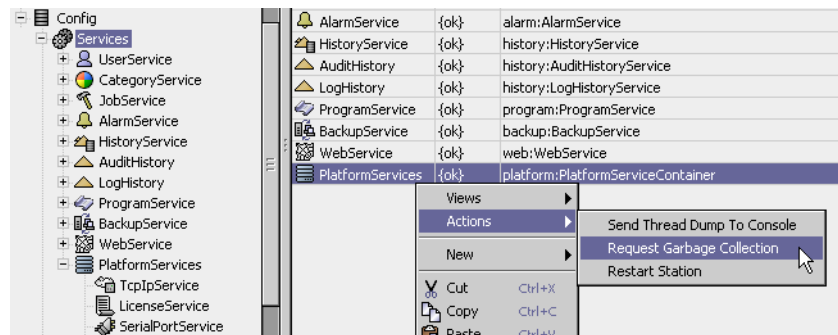
- **RAM Disk Size**
Has two configurable fields:
 - Min Free — minimum allowable free size in %. If status is not Ok, a “Low RAM disk space” warning is overlayed in all Workbench views of the station.
 - Size —in MB, specifies the amount of RAM disk used to store history and alarm files, where default is 32 for JACE-6 series, 16 for JACE-4/5 series, or 8 for JACE-2 series.
- **Java Heap**
Has one configurable “Min Free” field, in MB. Used by the station to compare (test) for low memory conditions, that is excessive Java heap. Also shown is a read-only Max field, in MB, for the heap size. If status is not Ok, a “Low memory” warning is overlayed in all Workbench views of the station.
- **Open File Descriptors**
Has one configurable “Min Free” field, related to number of files (and/or open sockets). Specifies the maximum amount of file descriptors that can be used. That is, the read-only “Max Open” number minus the “Min Free” amount. File descriptors are used for histories, modules, and Fox connections. If exceeded a “Station has too many open files or sockets” warning is overlayed in all Workbench views of the station.
- **Free RAM**
Has one configurable “Min Free” field, in KB. Specifies the minimum RAM that can be left free during station operation. If status is not Ok, a “Low free RAM” warning is overlayed in all Workbench views of the station.
- **Disk Space**
Has one configurable “Min Free” field, in %. Specifies the minimum percentage of disk storage that can be left free during station operation. Below this amount, a “Platform running low on disk space” warning is overlayed in all Workbench views of the station.

Note: Current statistics for memory, heap, and file descriptors (fd) are accessible on a station opened in Workbench, via the **Resource Manager** view of the Station component.

PlatformServiceContainer actions

The PlatformServicesContainer also provides three (right-click) actions, as shown in [Figure 1-117](#).

Figure 1-117 PlatformServicesContainer actions.



These actions are described as follows:

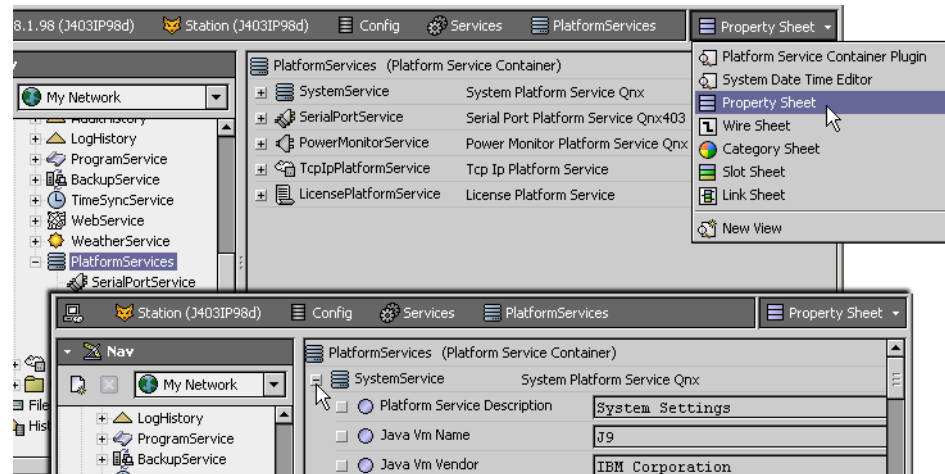
- **Send Thread Dump to Console**
Causes that host's platform daemon to have the station send a VM thread dump to its [station output](#) (console), equivalent to the "Dump Threads" command in the [Application Director](#). Typically used only during troubleshooting.
Note: Apart from *Application Director (platform access)* to view station output, you can also view a "snapshot" of station output in a browser. Do this via the "stdout" link in the **spy** utility, at URL `http://<hostIP>/ord?spy:/stdout`
- **Request Garbage Collection**
Causes the JVM running the station to perform garbage collection. This results in a "best effort" towards releasing unused objects and making more memory available on the "heap". Note that current heap and memory statistics for any running station are available on the [ResourceManager](#) view of the station component.
- **Restart Station**
Produces a popup confirmation dialog. Applies directly to a station running in a [Win32-based](#) platform, where it is equivalent to issuing a "Restart" command from the [Application Director](#) (station is saved on its host, then restarted). If issued to a station running on a [QNX-based](#) platform, this results in a host *reboot* (station restart not available unless host is rebooted).

Note: Also, most child services under the [PlatformServices](#) container have an available "Poll" action, which refreshes their property values. See "[Platform service types](#)" for a listing of possible child services.

SystemService (under PlatformServices)

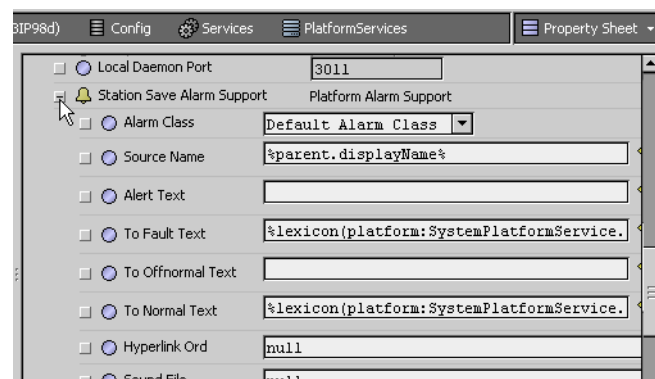
[PlatformServices](#) also contains a child "SystemService" container, accessible from its property sheet as shown in [Figure 1-118](#). Unlike other child services, SystemService does not appear in the Nav bar.

Figure 1-118 SystemService from property sheet of PlatformServices.



When you expand SystemServices, you see most of the same properties available in the default Platform Service Container Plugin view (see "[PlatformServiceContainer parameters](#)" on page 1-67). In addition, starting in AX-3.2 there is container slot "Station Save Alarm Support," as shown in [Figure 1-119](#).

Figure 1-119 Station Save Alarm Support expanded in property sheet of SystemService.



These properties allow you to configure the alarm class and other parameters to use for “station save” alarms. Such an alarm may occur, for example, if there is insufficient disk space to complete the save. Properties work the same as those in an alarm extension for a control point. For property descriptions, see the *User Guide* section [“About alarm extension properties”](#).

Note: *Other platform warnings from defined limits, such as for low memory, low disk space, and so on (see [Figure 1-116](#) on page 70 for related properties) are not really alarms—they simply generate a yellow overlay in the lower right corner when viewing the station in Workbench. If you need actual alarms, you can link from an appropriate boolean slot of the `SystemService` component (for example, “LowHeap”) into other persisted station logic in another area of the station.*

Platform service types

In addition to the `SystemService` found under its property sheet, the `PlatformServicesContainer` has various child services, of which different types are listed below.

Note: *Some platform services are intended to support installations where all configuration must be done using only a browser connection (and not Workbench connected to a JACE’s platform daemon). Examples include types `TcpIpService` and `LicenseService`.*

The complete list of visible platform service types is as follows:

- **TcpIpService**
Provides access to the same configuration using the platform’s [TCP/IP Configuration](#) view.
- **LicenseService**
Provides access to the same configuration using the platform’s [License Manager](#) view.
- **SerialPortService**
Allows review of available serial ports on the host platform. For [QNX-based](#) JACEs with configurable serial ports (e.g. JACE-403), this is where you *configure port usage*. For a related procedure, see [“JACE serial port configuration”](#) in the JACE NiagaraAX Install & Startup Guide.
- **PowerMonitorService**
Currently applies to [QNX-based](#) JACEs, providing configuration and status of the JACE’s battery monitoring and AC power-fail shutdown routines. See [JACE power monitoring](#) for details. This service also applies to some JACE-NXS models, see [JACE-NXS power monitoring](#).
- **DialupService**
Provides status of the platform’s dialup daemon and allows “blockout” of dialup functions from station logic, if desired. For details, see [“About the dialup daemon and service”](#) on page 1-17.
- **HardwareMonitorService**
Applies to JACE-NXS or JACE-NX station only. Provides status of several internal environmental variables, including alarm limit configuration. See [JACE-NX hardware monitoring](#) and [JACE-NXS hardware monitoring](#) for details.

Using platform services in a station

Apart from configuration usage, some platform services under the `PlatformServicesContainer` provide status values that you can further incorporate. Typically, each value also provides built-in alarm features. Usage is typical for the following:

- [JACE power monitoring](#)
- [JACE-NXS power monitoring](#)
- [JACE-NX hardware monitoring](#)
- [JACE-NXS hardware monitoring](#)

JACE power monitoring

Currently, through the `PowerMonitorService`, any [QNX-based](#) JACE provides status monitoring of the following items, via “Boolean” type slots:

- **AC power**
 (“Primary Power Present” slot) — True whenever AC power is currently supplied to the JACE.
- **Battery level**
 (“Battery Good” slot) — True if last JACE test of NiMH backup-battery was good.
 Also included is a “Time of Last Test” slot that provides a timestamp for the last battery test.

If needed, you can make Px bindings or links to these slots (however, see [“PlatformServices binding and link caveats”](#) on page 1-73). In addition to these read-only status slots, the `PowerMonitorService` provides related *configuration* slots, which you typically review at commissioning time. For more details and a related procedure, see [“JACE power monitoring configuration”](#) in the JACE NiagaraAX Install & Startup Guide.

Note: As new QNX-based JACE platforms are introduced, additional power-monitoring capabilities may be present in the station's `PowerMonitorService`. For example, the Security JACE (SEC-J-201) has two backup batteries: the NiMH battery like a JACE-2, plus a 12V sealed lead-acid (SLA) battery that provides system operation for some duration during a power outage. Separate slots exist for the monitoring and alarming of both batteries.

JACE-NXS power monitoring

Note: Often, a hard-drive based JACE-NXS does not have the special SITOP DC UPS module (with battery accessory), in which case its `PowerMonitorService` will have no application. The following slots do apply to the CompactFlash-based JACE-NXS, which will be so equipped.

Currently, through the `PowerMonitorService`, a station running in a JACE-NXS provides status monitoring of the following items, via “Boolean” type slots:

- **DC power**
 (“Primary Power Present” slot) — True whenever DC power is currently supplied to the UPS.
- **Battery level**
 (“Battery Ok” slot) — True if last UPS backup-battery test was good.
- **UPS Connectivity**
 (“Ups Talking” slot) — value is “talking” if last JACE attempt to communicate to the UPS was successful. Another possible state is “no comm.” Note that a USB cable must be connected between the JACE-NXS and the SITOP UPS module.

If needed, you can make Px bindings or links to these slots (however, see “[PlatformServices binding and link caveats](#)” on page 1-73). In addition to these read-only status slots, this `PowerMonitorService` provides related *configuration* slots, which you typically review at commissioning time. For more details and a related procedure, see “[Power monitoring JACE-NXS configuration](#)” in the *JACE-NXS NiagaraAX Install & Startup Guide*.

JACE-NX hardware monitoring

A station in the (discontinued) Win32-based JACE-NX provides status monitoring of various *internal* hardware parameters, via “Float” type slots under its `HardwareMonitorService`.

A few of the items monitored include:

- **CPU temperature**
 (“Cpu Temp” slot) — Value in degrees (C or F) of the mainboard under the JACE-NX's CPU.
- **CPU fan Speed**
 (“Cpu Fan Speed” slot) — RPM value of the CPU fan inside the JACE-NX.
- **Chassis fan Speed**
 (“Sys Fan Speed” slot) — RPM value of the JACE-NX's rear-mounted chassis fan.

Also included are various internal voltage values (Vtt, CPU core, Vcc 3 and 5, +12, -12, Vsb).

If needed, you can make Px bindings or links to these slots (however, see “[PlatformServices binding and link caveats](#)” on page 1-73). In addition to these read-only status slots, the `HardwareMonitorService` provides related *configuration* slots, which you typically review at commissioning time. For more details and a related procedure, see “Hardware monitoring in the JACE-NX” in the *JACE-NX NiagaraAX Install & Startup Guide*.

JACE-NXS hardware monitoring

A station in the Win32-based JACE-NXS provides status monitoring of *internal* hardware parameters, via “Float” type slots under its `HardwareMonitorService`.

The two items monitored are:

- **CPU temperature**
 (“Cpu Temp” slot) — Value in degrees (C or F) of the mainboard under the JACE-NXS's CPU.
- **Board temperature**
 (“Board Temp” slot) — Value in degrees (C or F) of the space inside the chassis.

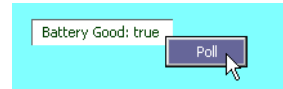
If needed, you can make Px bindings or links to these slots (however, see “[PlatformServices binding and link caveats](#)” on page 1-73). In addition to these read-only status slots, the `HardwareMonitorService` provides related *configuration* slots, which you typically review at commissioning time. For more details and a related procedure, see “[Hardware monitoring JACE-NXS configuration](#)” in the *JACE-NXS NiagaraAX Install & Startup Guide*.

PlatformServices binding and link caveats

Because any station's `PlatformServices` are dynamically built upon startup, if binding its slots to Px widgets (or linking to other station components), be aware of the following limitations/guidelines:

- Subscription behavior is unique to a station's PlatformService slots, in that property values initially load, but do *not automatically update*. To explicitly refresh such properties, you must invoke the "poll" action of the container for those properties.
For example, if on a Px page you bind a BoundLabel to the PowerMonitorService's "Battery Good" slot, it will display text as "true" or "false." However, this value does not update until the user right-clicks for the "Poll" action, which forces a fresh read.

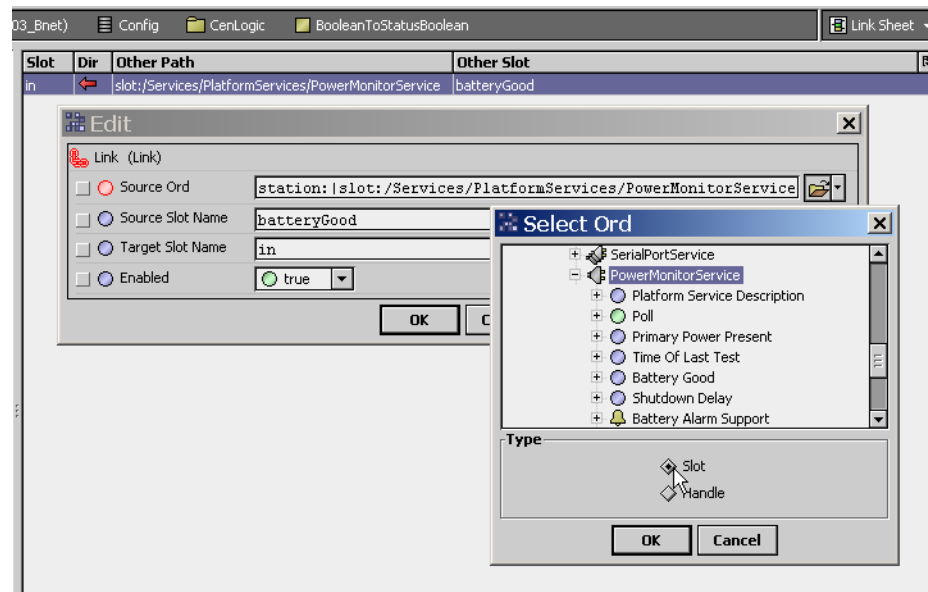
Figure 1-120 Poll action on bound PlatformService property



- Links from PlatformServices (and child slots) to other station components must use a source ord "slot path", versus "handle". Otherwise, after a station restart or host reboot, handle-sourced links may be lost. An example link being edited to use slot path is shown in Figure 1-121.

Note: Consider the "update limitation" before linking PlatformService slots into other components that provide control logic. Linked slot values may well be outdated shortly after station startup, yet still "subscribed" and not marked as "stale."

Figure 1-121 From LinkSheet of target component, editing link to use slot path for source ord.



However, note that the station's plugins (views) for the PlatformService *do* provide updated property values, as they work in concert with the special polling used for platform-resident data.

CHAPTER 2

Platform Component Guides

Component Guides provides summary information on platform-related components.

Component Reference Summary


Summary information is available on components in the following modules:

- [platDialup](#)
- [platform](#)
- [platLon](#)
- [platPower](#)
- [platPowerNxs](#)
- [platSerialQnx](#)
- [platSerialWin32](#)
- [platSysmonNx](#)
- [platSysmonNxs](#)

Components in platDialup module

- [DialupPlatformService](#)


platDialup-DialupPlatformService

 [DialupPlatformService](#) is the station's interface to the platform's [dialup daemon](#). This service is found under the running station's [PlatformServiceContainer](#). For more details, see “[DialupService](#)” on page 1-18.


Components in platform module

- [DaemonFileSpace](#)
- [DaemonSession](#)
- [LicensePlatformService](#)
- [Platform](#)
- [PlatformAlarmSupport](#)
- [PlatformServiceContainer](#)
- [SystemPlatformServiceQnx](#)
- [SystemPlatformServiceWin32](#)
- [TcpIpPlatformService](#)


platform-DaemonFileSpace

 [FileSpace](#) implementation for the files which can be accessed using the platform daemon.


platform-DaemonSession

 [DaemonSession](#) represents a platform connection to a host made in Workbench. In the Nav tree view, the [DaemonSession](#) icon is labeled **Platform**, and is directly under the host to which the platform session is in progress. The default view is the **Nav Container View**, which provides a table of all the various platform views (see [Figure 1-1](#) on page 1).


platform-LicensePlatformService

 The LicensePlatformService provides station access to the host platform's license(s) and certificate(s). This service is found under the running station's [PlatformServiceContainer](#). From the default plugin (view), you can perform the same operation as from the [License Manager](#) view using a platform connection.

platform-Platform

 Platform provides a component model of the Niagara Platform: OS, VM, framework, modules, etc. The Platform is available under system and in any connected station.

platform-PlatformAlarmSupport

 PlatformAlarmSupport is a container slot that appears for each alarmable value under a Platform Service, such as (for a QNX-based JACE), the [PowerMonitorPlatformServiceQnx](#), or for a JACE-NXS, the [HardwareMonitorNxsPlatformServiceWin32](#) (internal JACE-NXS temperature, etc).

For a QNX-based JACE, example PlatformAlarmSupport components include:


- **Battery Alarm Support**
To configure how “low battery level” alarms are handled in the station.
- **Power AlarmSupport**
To configure how “AC power loss” alarms are handled in the station.

For a JACE-NXS, example PlatformAlarmSupport components include:


- **Cpu Temperature Alarm Support**
Under the HardwareMonitorService, to configure how “Cpu Temperature Fail” alarms are handled in the station.
- **Ups Not Found Alarm Support**
Under the PowerMonitorService, to configure how “UPS not responding” alarms (if applicable) are handled in the station.

Properties under each PlatformAlarmSupport container are used to designate the station's Alarm Class to be used, and also to populate the alarm record when the specific alarm occurs. These properties work in the same fashion as those in an alarm extension for any control point.

platform-PlatformServiceContainer

 PlatformServiceContainer provides a container for a station's PlatformService instances. The [PlatformServiceContainerPlugin](#) is its primary view. The PlatformServiceContainer is available in the platform Module. For more details, see “[PlatformServiceContainer parameters](#)” on page 1-67.


platform-SystemPlatformServiceQnx

 SystemPlatformServiceQnx is the Qnx implementation of SystemPlatformService. For more details, see “[SystemService \(under PlatformServices\)](#)” on page 1-71.

Reboot action

Reboot action causes system reboot.


platform-SystemPlatformServiceWin32

 SystemPlatformServiceWin32 is the Win32 implementation of SystemPlatformService. For more details, see “[SystemService \(under PlatformServices\)](#)” on page 1-71.

Reboot action

Reboot action causes a system reboot. This is not available in Win32 systems if the platform authentication setting labeled “Stations - allow stations to have admin access to platform daemon” is disabled. See [Figure 1-70](#) on page 46 and the discussion following it for more details.


platform-TcpIpPlatformService

 TcpIpPlatformService provides station access to the host platform's TCP/IP settings. This service is found under the running station's [PlatformServiceContainer](#). From the default plugin (view), you can perform the same operation as from the [TCP/IP Configuration](#) view using a platform connection.


If a Win32 host and the platform authentication setting labeled “Stations - allow stations to have admin access to platform daemon” is disabled, TCP/IP properties in this view are read-only. See [Figure 1-70](#) on page 46.

Components in platLon module

platLon-LonPlatformServiceQnx

 A LonPlatformServiceQnx is used to access the native lonworks Driver system interface. The LonPlatformServiceQnx is available as lon under Platform in a Qnx JACE.


platLon-LonPlatformServiceWin32

 A LonPlatformServiceWin32 is used to access the native lonworks Driver system interface. The LonPlatformServiceWin32 is available as lon under Platform in a Win32 JACE.


Components in platPower module

- [JaceSlaBattery](#)
- [Npm2NimhBattery](#)
- [NpmDualBatteryPlatformService](#)
- [NmpExternalSlaBattery](#)
- [PowerMonitorPlatformServiceQnx](#)


platPower-JaceSlaBattery

 JaceSlaBattery is the “Battery” slot under the PowerMonitorService in a JACE-4/5 series station’s PlatformServices container. This slot simply indicates the host JACE platform uses a sealed lead acid type (SLA) battery.

platPower-Npm2NimhBattery


 Npm2NimhBattery is the “Battery” slot under the PowerMonitorService in a JACE-2 series (NPM2) station’s PlatformServices container. This slot simply indicates the host JACE platform uses a nickel-metal hydride (NiMH) battery.

platPower-NpmDualBatteryPlatformService


 NpmDualBatteryPlatformService (PowerMonitorService) applies only to station running in a Security JACE (SEC-J-201), and is used to monitor primary power status and backup battery levels in *both* the onboard NiMH battery as well as the sealed-lead acid (SLA) battery. A few configuration parameters allow changing the shutdown delay time, as well as alarm source configuration for all three types of alarms (low NiMH battery level, low SLA battery level, primary power lost).

This PowerMonitorService is found only under the [PlatformServiceContainer](#) in a station running on the Security JACE. See “[Using platform services in a station](#)” on page 1-72 for related details.

platPower-NpmExternalSlaBattery

 NpmExternalSlaBattery is one of two “Battery” slots under the PowerMonitorService in a Security JACE’s station’s PlatformServices container. This slot simply indicates the host JACE platform uses a sealed-lead acid (SLA) battery, *in addition to* the onboard NiMH backup battery.


platPower-PowerMonitorPlatformServiceQnx

 PowerMonitorPlatformServiceQnx (PowerMonitorService) is used to monitor the primary power status and backup battery level in most QNX-based JACEs. A few configuration parameters allow changing the shutdown delay time, as well as alarm source configuration for both types of alarms (low battery level, primary power lost).

This PowerMonitorService is found under the [PlatformServiceContainer](#) in a station running on any QNX-based JACE *except* for the Security JACE (SEC-J-201), which uses the [NpmDualBatteryPlatformService](#) instead. See “[Using platform services in a station](#)” on page 1-72 for related details.

Components in platPowerNxs module

platPowerNxs-PowerMonitorPlatformServiceNxswin32

 PowerMonitorPlatformServiceNxswin32 (PowerMonitorService) is used to monitor the status of primary power, UPS communications, and UPS battery condition for a JACE-NXS. Configuration parameters allow changing the shutdown delay time, as well as alarm source configuration for all three types of alarms (low battery level, primary power lost, UPS communications).


The PowerMonitorService is found under the [PlatformServiceContainer](#) in a station running on any JACE-NXS. See “[Using platform services in a station](#)” on page 1-72 for related details. For specific details, see the section “Power monitoring configuration in JACE-NXS” in the *JACE-NXS NiagaraAX Install & Startup Guide*.

Note: *This service applies to any CompactFlash-based JACE-NXS, which includes the special “SITOP” DC UPS and UPS battery modules. However, if a hard drive-based JACE-NXS (installed without this UPS option), you can safely ignore this service, and its contained slots.*


Components in platSerialQnx module

- [SerialPortPlatformServiceQnx](#)
- [SerialPortPlatformServiceQnx403](#)
- [SerialPortPlatformServiceQnx404](#)


platSerialQnx-SerialPortPlatformServiceQnx

 SerialPortPlatformServiceQnx is the station's interface to the platform's serial port configuration, such as used by a JACE-2 series host. This service is found under the running station's [PlatformServiceContainer](#) as the **Serial Port Service**.

platSerialQnx-SerialPortPlatformServiceQnx403


 SerialPortPlatformServiceQnx403 is the station's interface to the platform's serial port configuration, in this case specific to a JACE-403 series host. This service is found under the running station's [PlatformServiceContainer](#) as the **Serial Port Service**.

platSerialQnx-SerialPortPlatformServiceQnx404

 SerialPortPlatformServiceQnx404 is the station's interface to the platform's serial port configuration, in this case specific to a JACE-545 series host. This service is found under the running station's [PlatformServiceContainer](#) as the **Serial Port Service**.


Components in platSerialWin32 module

platSerialWin32-SerialPortPlatformServiceWin32

 SerialPortPlatformServiceWin32 is the station's interface to the platform's serial port configuration, used by any Win32 (Windows XP) based host, such as a JACE-NXS or AxSupervisor PC. This service is found under the running station's [PlatformServiceContainer](#) as the **Serial Port Service**.

Components in platSysmonNx module

platSysmonNx-HardwareMonitorNxPlatformServiceWin32


 HardwareMonitorNxPlatformServiceWin32 (HardwareMonitorService) is the station's interface to internal environmental parameters in the host JACE-NX, such as CPU temperature, fan speeds, and various voltages. This service appears under the running station's [PlatformServiceContainer](#) as the **Hardware Monitor Service**.

See “[Using platform services in a station](#)” on page 1-72 for related details. For specific details, see the section “[Hardware monitoring JACE-NX configuration](#)” in the *JACE-NX NiagaraAX Install & Startup Guide*.

Components in platSysmonNxs module

- [HardwareMonitorNxsPlatformServiceWin32](#)

platSysmonNxs-HardwareMonitorNxsPlatformServiceWin32

 HardwareMonitorNxsPlatformServiceWin32 (HardwareMonitorService) is the station's interface to internal environmental parameters in any JACE-NXS host, namely the CPU temperature and board temperature. This service appears under the running station's [PlatformServiceContainer](#) as the **Hardware Monitor Service**.

For more details, see the section “Hardware monitoring JACE-NXS configuration” in the *JACE-NXS NiagaraAX Install & Startup Guide*.

CHAPTER 3

Platform Plugin Guides

There are many ways to view plugins (*views*). One way is directly in the tree. In addition, you can right-click on an item and select one of its views. Plugins provide views of components.

Access the following summary descriptions on any plugin by selecting **Help > On View** (F1) from the menu, or by pressing F1 while the view is open.

Plugin Reference Summary


Summary information is provided on views in the following modules:

- [platDaemon](#)
- [platDDNS](#)
- [platDialup](#)
- [platform](#)

Plugins in platDaemon module


- [ApplicationDirector](#)
- [DaemonLogSettingsView](#)
- [DistInstaller](#)
- [DistributionView](#)
- [FileTransferClient](#)
- [LexiconInstaller](#)
- [LicenseManager](#)
- [SoftwareManager](#)
- [SoftwareView](#)
- [PlatformAdministration](#)
- [PlatformTextSummaryEditor](#)
- [StationCopier](#)
- [StationDirector](#)
- [TcpIpConfiguration](#)
- [UserManager](#)

platDaemon-ApplicationDirector


 The Application Director (formerly: **Station Director**) is the AX-3.3 and later platform view that allows you to start, stop, restart, and kill a station on the connected NiagaraAX platform. You also use it to examine standard station output, for troubleshooting and debug purposes. Starting in AX-3.3, along with the renaming to **Application Director**, limited support began also for starting, stopping, and monitoring.

For more details, see “[Application Director](#)” on page 1-7.


platDaemon-DaemonLogSettingsView

 DaemonLogSettingsView allows you to view the log settings.

platDaemon-DistInstaller

 DistInstaller allows you to install distribution files from this computer to the remote host. For more details, see “[Distribution File Installer](#)” on page 1-21.


platDaemon-DistributionView

 Distribution View is the dialog that appears when you double-click a distribution file listed in the platform's [Distribution File Installer](#) view. A number of details is provided about the selected distribution file, including all contents and any dependencies.


platDaemon-FileTransferClient

 The File Transfer Client is the platform view that allows you to *copy* files and/or folders between your Workbench PC and the remote Niagara platform, as needed. For more details, see [“File Transfer Client”](#) on page 1-28.


platDaemon-LexiconInstaller

 Lexicon Installer allows you to install lexicon files (for Niagara localization) on a remote host. For more details, see [“Lexicon Installer”](#) on page 1-29.


platDaemon-LicenseManager

 License Manager allows you to view and install files required for Niagara licensing. For more details, see [“License Manager”](#) on page 1-30.


platDaemon-SoftwareManager

 The Software Manager is the platform view you use to install, upgrade, or remove modules in the connected Niagara platform. For more details, see [“Software Manager”](#) on page 1-36.


platDaemon-SoftwareView

 Software View is the dialog that appears when you double-click an item (for example, module) listed in the platform's [Software Manager](#) view. A number of details is provided about the selected item.


platDaemon-PlatformAdministration

 The Platform Administration view provides access to various platform daemon (and host) settings and summary information. Included are buttons to perform various platform operations. For more details, see [“Platform Administration”](#) on page 1-43.


platDaemon-PlatformTextSummaryEditor

 PlatformTextSummaryEditor is the dialog that appears when you click the export tool button when using the [Platform Administration](#) view. Setup in this dialog allows you to include/exclude the platform summary data and the platform daemon console output, as well as limit daemon output.


platDaemon-StationCopier

 The Station Copier is the platform view that you use to install a station in a remote JACE host, as well as make a local backup copy of a remote JACE station onto your Workbench PC. You can also delete stations using this view. For more details, see [“Station Copier”](#) on page 1-54.


platDaemon-StationDirector

 The Station Director is the pre-AX-3.3 platform view that allows you to start, stop, restart, and kill a station on the connected NiagaraAX platform. You also use it to examine standard station output, for troubleshooting and debug purposes. Starting in AX-3.3, this view was renamed to the **Application Director**, but functionality for stations (NiagaraAX applications) remained unchanged. For more details, see [“Application Director”](#) on page 1-7.


platDaemon-StationTextSummaryEditor

 StationTextSummaryEditor is the dialog that appears when you click the export tool button when using the [Application Director](#) view. Setup in this dialog allows you to include/exclude the platform summary data, platform daemon console output, station console output, as well as limit both the daemon and station output.

platDaemon-TcplpConfiguration

 TCP/IP Configuration is the platform view you use to configure a remote JACE host's TCP/IP settings. Typically, you make initial settings when you first commission the JACE, where this view is one step in the platform's Commissioning Wizard. For more details, see [“TCP/IP Configuration”](#) on page 1-59.


platDaemon-UserManager

 The User Manager is a platform view available only for Win32 based hosts (e.g. JACE-NX). It allows you to manage Windows OS user and group accounts local to that host, which otherwise would require accessing “Administrative Tools” in Windows on that host.

For more details, see “[User Manager](#)” on page 1-62.


Plugin in platDDNS

platDdns-DdnsConfigurationView


 Ddns Configuration allows for DNS IP addresses to be dynamically updated. For more details, see “[Ddns Configuration](#)” on page 1-13.

Plugins in platDialup

platDialup-ConfigureDialup

 Dialup Configuration is to configure a host’s modem and “captive network” settings. For more details, see “[Dialup Configuration](#)” on page 1-14.

platDialup-DialupEditor

 DialupEditor allows you to configure the dialup parameters for the connected host.


Plugins in platform module

- [ActivityView](#)
- [LicensePlatformServicePlugin](#)
- [PlatformServiceContainerPlugin](#)
- [PlatformServiceProperties](#)
- [SystemPlatformServicePlugin](#)
- [SystemPlatformServiceQnxPlugin](#)
- [TcpIpPlatformServicePlugin](#)


platform-ActivityView

The ActivityView allows you to view the items in the platform.

platform-LicensePlatformServicePlugin

 LicensePlatformServicePlugin allows you to manage the host’s licenses and certificates under a station’s [PlatformServiceContainer](#).

platform-PlatformServiceContainerPlugin

 PlatformServiceContainerPlugin allow you to view and edit information about the Platform. It includes the following:


- System - System Settings
- LON - LON settings
- Comm - Serial Communications

For more details, see “[About Platform Services](#)” on page 1-66.

platform-PlatformServiceProperties

 PlatformServiceProperties allows you to configure a remote host’s serial port properties or power monitoring properties under a station’s [PlatformServiceContainer](#).

platform-SystemPlatformServicePlugin

 System PlatformService Plugin allows you to view and edit information about the system.

Reboot

Reboot allow you to reboot the system.

platform-SystemPlatformServiceQnxPlugin

- SystemPlatformServiceQnxPlugin provides station access to platform parameters on QNX-based hosts.

platform-TcplpPlatformServicePlugin

- TcpIpPlatformServicePlugin allows you to manage the host's TCP/IP settings under a station's [PlatformServiceContainer](#). Settings include Hostname, Domain, the Hosts file and Interfaces.

APPENDIX A

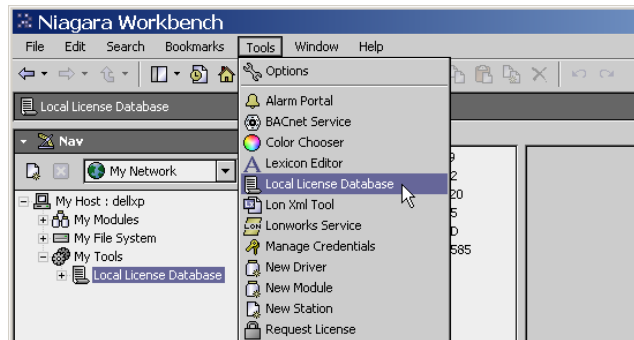
License Tools and Files

This appendix provides details about the Workbench tools related to NiagaraAX license files, including license-management enhancements starting in AX-3.3. Also included are details on the *contents* of license files.

The following subsections are included:

- **License-related Workbench tools**
Unlike platform views (which require a [platform connection](#)), or equivalent [PlatformServices](#) plugin views (requiring a station connection), Workbench tools are available whenever running full Workbench. Find Workbench tools on the Tools menu, as shown in [Figure A-1](#).
 - [Workbench License Manager](#) tool (AX-3.3 or later only)
 - [Request License](#) tool (any NiagaraAX version)

Figure A-1 Tools menu in AX-3.3 Workbench

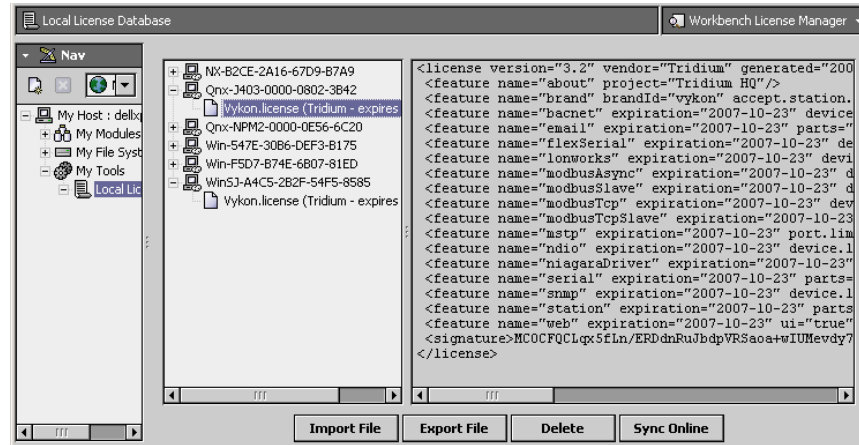


- **License management changes starting in AX-3.3 (in addition to Workbench License Manager):**
 - [local license database](#)
 - [license archive \(.lar\) files](#)
- [About NiagaraAX license files](#)
 - [Items common to all license files](#)
 - [JACE hardware features](#)
 - [Driver attributes](#)
 - [Driver types](#)
 - [Applications](#)

Workbench License Manager

Starting in AX-3.3, the **Workbench License Manager** view is available via the Workbench Tools menu, by selecting **Local License Database**.

Figure A-2 Workbench License Manager



As shown in [Figure A-2](#), this view lets you browse and manage the contents of your “local license database.”

Note: For details about the license database structure, see [“About the local license database”](#) on page A-6.

This view provides a two-pane window into all the license files and parent “host ID” folders, where

- Left pane provides tree navigation, where you can expand folders and click (to select) license files.
- Right pane shows the text contents of any selected license file.

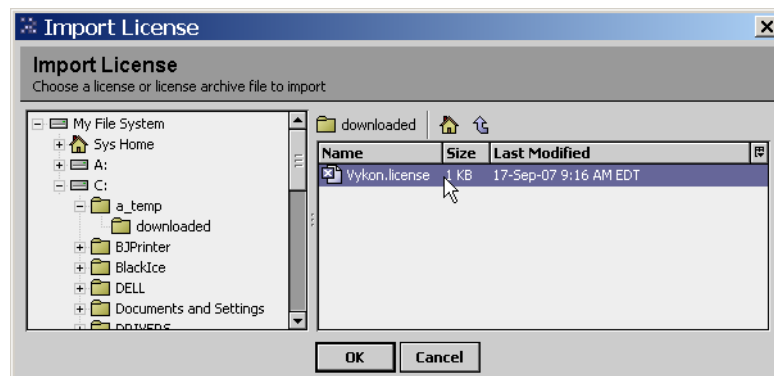
Buttons at the bottom of this view provide a way to manage the contents of your local license database, and are described as follows:

- **Import File** — Always available, this allows you to add license file(s) from a local license file or license archive (.lar) file.
- **Export File** — Always available, this allows you to save all licenses (or any selected licenses) locally, as a license archive file.
- **Delete** — This allows you to delete licenses from your Workbench local license database.
- **Sync Online** — Typically available if you have Internet connectivity. This lets you *update* all licenses (or any selected licenses) in your local license database with the *most current* versions, via the online licensing server.

Import File

The **Import File** button in the **Workbench License Manager** is always enabled, and produces the **Import License** dialog for you to navigate to a source file (.license or .lar), as shown in [Figure A-3](#). Note only these two types of files appear for selection.

Figure A-3 Import License dialog to find local license file or license archive file



Select a license file and click **OK** to add to (or update in) your local license database. A popup dialog ([Figure A-4](#)) informs you of success, and the license(s) are added or updated in your database.

Figure A-4 Import success



Note: If any of the license(s) you select to import are older than the ones currently in your local database, meaning that the “generated” attribute timestamp is earlier, newer license(s) in your local license database are not overwritten. However, the same “Import successful” message popup appears for such file import operations.

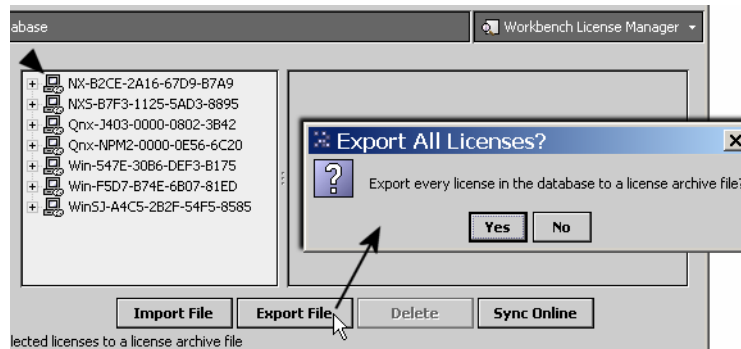
Export File

The **Export File** button in the [Workbench License Manager](#) allows you to save any number (or all) licenses in your local license database locally on your Workbench PC, as a license archive (.lar) file.

Note: The license archive format allows you to easily share saved .lar files (however named) among multiple Workbench PCs without overwriting a license file for a different host platform. You can use the [Import File](#) command in the Workbench License Manager to add/update licenses in a license archive, or the equivalent [Import](#) command when in the platform [License Manager](#) (or similar License Platform Service Plugin). For more details, see [“About license archive \(.lar\) files”](#) on page A-7.

If you click **Export File** without first selecting any licenses (and/or) host IDs, every license in your local license database will be included in the archive, as noted in a confirmation dialog. See [Figure A-5](#).

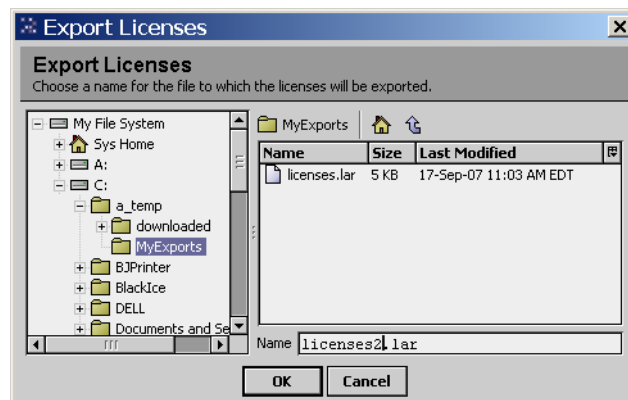
Figure A-5 Export All Licenses confirmation dialog



Or, you can select one or more entries in the left pane (host IDs or license files) to include only those selected (highlighted) licenses to be in the exported archive file.

When you click **Yes** (if all) or **Export File** for selected licenses, an **Export Licenses** dialog ([Figure A-6](#)) lets you navigate to the spot to save the .lar file.

Figure A-6 Export Licenses dialog



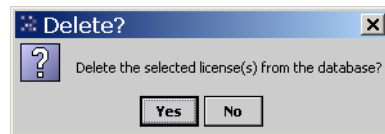
By default, a license archive file is saved in the root of your Niagara release directory. Use the dialog’s navigation controls to specify another target folder or drive, as needed. Before saving, you can also *rename* the license archive file, to make it more identifiable. For example, instead of: licenses.lar, you could rename it MyJaceNxs.lar.

Upon export of license(s) to a license archive file, a popup dialog appears, as shown in [Figure A-7](#).


Figure A-7 Export file success

Delete

The **Delete** button in the [Workbench License Manager](#) is enabled when you have one or more host IDs and/or license files selected in the left pane, and produces a confirmation dialog to delete licenses from your local license database, as shown in [Figure A-8](#).

Figure A-8 Delete licenses confirmation

Click **Yes** to delete the license(s), or **No** to leave the local license database unchanged.

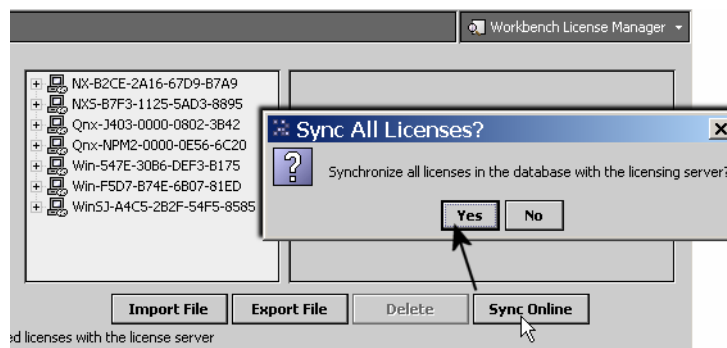
Note: Following a delete, you may need to click the  Refresh button in order to update the left pane contents. Note that if the selected “host ID” folder contained only a `.license` file, the entire folder is removed with a delete. However, if the folder contained other files (or subfolders), only the `.license` file is actually deleted, but it will no longer appear in the left pane.

Sync Online

The Sync Online feature in the [Workbench License Manager](#) allows you to update any number (or all) licenses in your local license database with the most current license, available *online* from the *licensing server*. This feature requires Internet connectivity from your Workbench PC.

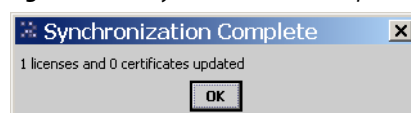
Note: For related details, see “[About the licensing server](#)” on page 1-35.

If you click **Sync Online** without first selecting any licenses (and/or) host IDs, every license in your license database will be included in the sync request, as noted in a confirmation dialog. See [Figure A-9](#).

Figure A-9 Sync All Licenses confirmation dialog

Or, you can select one or more entries in the left pane (host IDs or license files) to include only those selected (highlighted) licenses to be included in the sync request.

When you click **Yes** (if all) or **Sync Online** for selected licenses, an immediate request is sent to the licensing server. Intermediate popup dialogs may briefly appear while the sync request is handled. The operation concludes with a **Synchronization Complete** dialog, which summarizes the number of licenses and certificate files updated in your Workbench local license database. See [Figure A-10](#).

Figure A-10 Synchronization Complete dialog

If all licenses (and certificates) were already up-to-date, this dialog will say “0 licenses and 0 certificates updated”.

Request License

The “**Request License**” item on the Workbench Tools menu (Figure A-1 on page 1) simply opens a “license request form” in your Workbench PC’s default browser. By default, the only pre-filled field in this form is the host ID of your Workbench PC. See Figure A-11.

Figure A-11 License request form in browser (from Workbench, Tools > Request License)

The screenshot shows a web browser window with the address bar displaying `http://axlicensing.tridium.com/license/request?params=YnJ`. The page title is "License Request" and features the Niagara Central logo. The form is divided into three numbered sections:

- Section 1:** "This instance of Niagara will be licensed to the hardware platform with the following:"
 - Host ID*:** A text field containing the pre-filled value `Win-F5D7-B74E-6B07-81ED`.
 - License Key:** An empty text field.
- Section 2:** "The license request will be validated against an order. Please check your packing slip for the following information. Please Note: If you provide a license key with the request, only the Part or Item Number field is required."
 - Purchase Order Number*:** An empty text field.
 - Distributor*:** An empty text field.
 - Part or Item Number*:** An empty text field.
- Section 3:** "Please also provide the following information to help process your request:"
 - Name*:** An empty text field.
 - Company*:** An empty text field.
 - Email*:** An empty text field.

At the bottom of the form is a "Request License" button. A footnote at the bottom left states: "* Required fields".

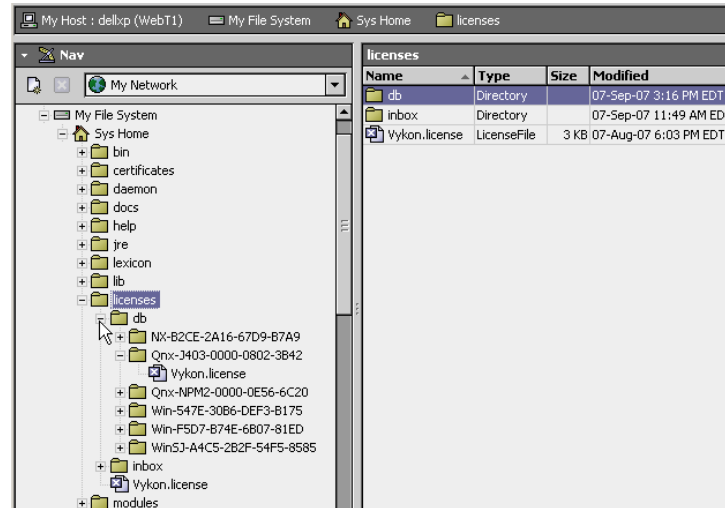
Typically, your Workbench PC is already licensed. Otherwise, you would not be able to successfully start Workbench, and then select **Request License** from the “Tools” menu.

However, you could use this as quick method to request a license for *another* PC on which you have installed NiagaraAX. In that case, you could substitute (type in) the host ID for the other PC in this form, along with other pertinent information.

About the local license database

Starting in AX-3.3, any Workbench PC (including an AxSupervisor) has a “local license database” that is a *structured* collection of subdirectories and file under its Niagara release (system home) `! /licenses/db` directory. Each subdirectory has a unique NiagaraAX “host ID” name, matching that for some remote host platform. [Figure A-12](#) shows an example of this license database structure, as viewed in the Workbench Nav tree.

Figure A-12 Workbench local license database (AX-3.3 and later) is everything under `! /licenses/db`



Your local license database is created and managed *automatically* by Workbench, and updated whenever you perform license operations from platform connections, PlatformService plugins, or when using Workbench tools such as the Workbench License Manager. Note that you can see the same directory/file structure when looking at this location on your Workbench PC using Windows Explorer.

Note: As in earlier pre-AX-3.3 releases, the license required for your (local) Workbench PC operation is in the root of the licenses folder, named simply by your brand, for example `Vykon.license`. For details on the Workbench License Manager, see [“Workbench License Manager”](#) on page A-2.

Local license database rationale

The local license database design starting in AX-3.3 makes it easier for Workbench to store licenses for multiple host platforms—without inadvertently overwriting one license file with another. No longer do you have to manually create special license folders (subdirectories), and/or rename license files uniquely. The related “license archive” storage file format (`.lar`) also facilitates the exchange of licenses among different Workbench PCs, and is used in updating/synchronizing licenses to the online licensing server, as well as with new provisioning features for Niagara Networks. See [“About license archive \(.lar\) files”](#) on page A-7.

Local license inbox

In addition to the `! /licenses/db` folder, there is also a `! /licenses/inbox` folder. The inbox allows “drag and drop” importing into your license database of both individual license files and “license archive” (`.lar`) files, which may have been “saved” or “exported” from other Workbench PCs, or perhaps sent to you from the licensing server.

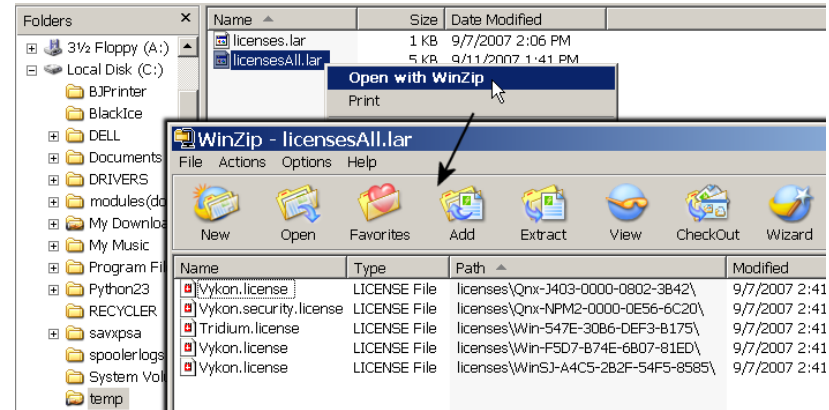
When you restart Workbench after copying license files and/or `.lar` files into your `inbox` subfolder, Workbench automatically creates the appropriate “host ID” named subfolders in your local license database, each containing the corresponding license file. Contents of the `inbox` folder are then deleted.

Note: If you upgraded Workbench (or an AxSupervisor) from an earlier release to AX-3.3, this inbox feature is particularly useful. Simply copy all your license files you previously stored (from whatever locations) into your AX-3.3 Workbench PC’s `! /licenses/inbox` folder, renaming files if necessary (to permit copying). After you restart Workbench, your local license database will be correctly structured. In addition, now you can use the “Sync Online” feature of the [Workbench License Manager](#) to ensure you have the latest version of all your licenses. See [“Sync Online”](#) on page A-4.

About license archive (.lar) files

Starting in AX-3.3, when you use the platform [License Manager](#) view or Workbench's [Workbench License Manager](#) view (under Tools) to *export* one or more license files, they are saved in a compressed (Zip compatible) format known as a *license archive*, that is a file with a “.lar” file extension. Any .lar file is simply a zip of the exported license file(s) that includes the complete “licenses/hostID” folder (subdirectory) structure for any included licenses. See [Figure A-13](#) for an example.

Figure A-13 License archive (.lar) is license file(s) in zip format, including folder paths relative to sys home.



[Figure A-13](#) shows an license archive file in Windows Explorer being opened using WinZip, and its subsequent contents. In this case where the archive contains multiple licenses, it was created by an export performed using the [Workbench License Manager](#) tool. However, if you export a license in the [License Manager](#) (AX-3.3 or above) when platform-connected to a remote host, the license archive file will contain only that one license.

Licensing server use of license archives

Starting around or before the AX-3.3 release timeframe, the NiagaraAX licensing server also uses the “license archive” (.lar) format when it sends out license files, as a file attachment to emails.

- In cases where your Workbench is AX-3.3 or later, you can simply use the **Import File** command in your [Workbench License Manager](#) view (in your Tools menu) to copy in the licenses.lar. This adds or updates those licenses in your [local license database](#). You can then import licenses from your local license database when either platform (or station) connected. Or, if your Workbench PC is Internet-connected while using the platform [License Manager](#) (or a station's equivalent [PlatformServices](#), License Manager view) you can use the **Import** command and select “Import licenses from the licensing server,” which installs the updated license in the host platform, *and* updates your local license database at the same time.
- In cases where your Workbench is an earlier release (say AX-3.1 or AX-3.2), you can unzip the contents of the licenses.lar under your !\\licenses folder, and use the **Install File** command in the License Manager to install licenses.

About NiagaraAX license files

A NiagaraAX license file is a structured XML file that has a .license file extension. It enables a set of vendor specific *features*. Each license file is valid for one specific host platform (JACE, PC), matched by that host's unique *host ID*. License files are “digitally signed” by the vendor to prevent tampering.

The following sections provide more details on the contents of any license file that validates against the Tridium certificate:

- [Items common to all license files](#) (license, about, brand, signature)
- [JACE hardware features](#) (maxheap, mstp, ndio, serial)
- [Driver attributes](#) (name, expiration, device.limit, history.limit, point.limit, schedule.limit, parts)
- [Driver types](#) (many types, including bacnet, lonworks, modbusTcp, obixDriver, niagaraDriver)
- [Applications](#) (crypto, eas, email, genericAppliance, provisioning, station, tenantBilling, web, workbench)

Items common to all license files

license

All license files require an opening `<license >` line, where the last line in the license file is the closing `</license>` tag, and all contents (lines) in between are `<feature >` elements, plus one signature element.

In the first `<license >` line, there are a number of common attributes, described below.

```
<license version="3.2" vendor="Tridium" generated="2007-04-11"
expiration="2008-03-31" hostId="Win-6827-91CB-C49A-6B4B" serialNumber="4856">
```

version version="3.2" - The highest release version of software which can be installed in the JACE. If a newer version of software is installed, the JACE will fail on startup with a license version error.

vendor vendor="Tridium" - This is always Tridium.

generated generated="2007-04-11" - The date upon which the license file was generated.

expiration expiration="2008-03-31" - The expiration date of the license file. After the expiration date the Workbench software will fail start due to a license expired error. Typically, engineering copies of Workbench have expiration dates which expire on an annual basis. License files for actual projects are issued with non-expiring licenses, where this attribute value is "never".

hostId hostId="Win-6827-91CB-C49A-6B4B" - Alphanumeric code unique to the specific host, which is generated upon installation of the NiagaraAX software on a Windows-based platform. QNX-based JACE controllers are assigned a hostId similar to this: hostId="Qnx-NPM2-0000-0E8F-2420". The hostId in the license file must match the hostId of the JACE controller, otherwise the JACE cannot run a station.

serialNumber serialNumber="4856" - Applies to a license for a QNX-based JACE only. Designates its unique serial number assigned from the factory. The serial number in the license file must match the serial number of the JACE..

about

The "about" feature is used to designate optional information, and does not affect station operation in any way. This information can be useful for filtering records when searching the license database. Two attributes in this feature are typically designated when ordering product:

```
<feature name="about" owner="Tridium" project="Tridium Testing"/>
```

owner owner="Tridium" - Optional field to designate the name of a person who is responsible for the project or possibly an end user.

project project="Tridium Testing" - Optional field to designate a project. This grouping should typically be assigned to all JACE controllers used for a particular project.

brand

For any license with vendor="Tridium", the NiCS (Niagara Compatibility Structure) provides a structure (or schema) that OEMs can use to define the various levels and types of Niagara interoperability that their products will support. The NiCS definitions are contained in the this feature item which is checked by a station or tool when it starts up. There are five attributes to the NiCS: BrandID, Station Compatibility In, Station Compatibility Out, Tool Compatibility In, and Tool Compatibility Out. These elements can be combined in a variety of ways to achieve unlimited flexibility, and are described below.

```
<feature name="brand" brandId="Vykon" accept.station.in="*"
accept.station.out="*" accept.wb.in="*" accept.wb.out="*" />
```

brandId brandId="Vykon" - Every licensed station and tool has a Brand Identifier (BrandID). This field holds a text descriptor that the OEM chooses as the identifier for its product line. Each station or tool can have only one BrandID entry.

accept.station.in accept.station.in="*" - A list of brands that this local station will allow NiagaraAX data to come in from. Simply stated from a JACE perspective, "this is the list of brands that I can accept data from". The "*" is a wildcard designation to allow all brands.

accept.station.out accept.station.out="*" - A list of brands that this local station will allow NiagaraAX data to be shared with. Simply stated, "This is the list of brands that I can share data with".

accept.wb.in accept.wb.in="*" - A list of brands that this station will allow to be connected to it for engineering of its application. Simply stated, "This is the list of brands that can engineer me".

accept.wb.out accept.wb.out="*" - A list of brands that this tool is allowed to connect to and engineer. Simply stated, "This is the list of brands that I can engineer".

signature

This element contains a digital signature which is created when the license file is generated. It prevents tampering with the license file. Attempts to edit the license file to enable additional features will render the license file useless. Typically, the signature element is the last element contained in the license, so it is followed by the closing license tag as the last line in the license file.

```
<signature>MCwCFFOdq4wJcYgvhTVtrf0oSyuCDCwJAhRj+H9pNxQGStBnhEkIqK8rONB10g==</signature>
</license>
```

Note: All licenses for any JACE or AxSupervisor platform also require a "station" application feature, in order to run a local station. See the section "[Applications](#)" on page A-12 for related details.

JACE hardware features

maxheap

This feature determines the maximum size of the Java heap for either a JACE-2 or JACE-6 series controller. In the absence of this feature, the maximum heap size is limited to 16 MB for a JACE-2, and 48 MB for a JACE-6. When this license feature is present, the maximum heap size is limited to 48 MB for a JACE-2 and 96 MB for a JACE-6. The feature may not be available on all JACE-2 controllers, earlier models were manufactured with 64 MB RAM chips. Verify the amount of physical memory in a JACE-2 before attempting to order the memory upgrade option.

```
<feature name="maxHeap" expiration="never" parts="NPM-128">
```

mstp

This feature determines how many of the available serial ports may be used for BACnet MS/TP communications. Note that features [bacnet](#) and [serial](#) must also exist in the license file.

```
<feature name="mstp" expiration="2008-03-31" port.limit="1" parts="AX-DEMO"/>
```

port.limit port.limit="1" - This specifies the number of serial ports which may be used for MSTP communications. Typically this number matches the number of physical ports. Some JACE controller models have option card slots, if additional serial cards are added then the port limit may be less than the number of physical ports if the port activation has not been ordered as well.

ndio

This feature enables the NDIO (Niagara Direct Input Output) driver, required to configure and use a JACE controller's I/O hardware. Not all JACE controllers support integrated or distributed I/O, refer to specific JACE controller datasheets to confirm whether this is an available option. Note that in the ndio features line (below), a "device" equates to an "Ndio Board", and that history and schedule limits have no practical application. See Driver Feature Attributes for related details.

```
<feature name="ndio" expiration="never" device.limit="none"
history.limit="none" point.limit="none" schedule.limit="none" parts="AX-DEMO"/>
```

serial

This feature enables the use of JACE serial ports for various drivers, for example aapup or modbusAsync. Note that the JACE license needs this serial feature in addition to any specific driver feature. Only one serial feature line is needed, regardless of number of serial-based drivers. Note that in the case of a JACE used for BACnet MS/TP, it would require this serial feature and driver features [bacnet](#) and [mstp](#).

```
<feature name="serial" expiration="2008-03-31" parts="AX-DEMO"/>
```

Driver attributes

Each driver is enabled by a feature line (element) in the license file. Most of the drivers utilize the same *attributes* within that feature. For simplicity, these common attributes are discussed first, and any unique attribute specific to a particular driver or service will be discussed in that [Driver types](#) section.

```
<feature name="driver name" expiration="expiration date" device.limit="none"
history.limit="none" point.limit="none" schedule.limit="none" parts="AX-DEMO">
```

Note: The various "limit type" attribute values can be either "none" or a numerical (limit) value, for example device.limit=32. Note that a limit value of none means unlimited, whereas a limit value of 0 means none allowed. Although most drivers include all the attributes shown in the feature line above, some attributes may not apply to a specific driver, with the exceptions of point.limit and device.limit attributes, which typically do apply to any driver.

For example, none of the Modbus-related drivers have any history or schedule import/export capability, due to the simplicity of the Modbus protocol. Therefore, "history.limit" and "schedule.limit" attributes have no importance in the feature line of those drivers.

name

Feature name of the driver, often the same as the actual module (.jar file) name, for example bacnet, lonworks, etc.

expiration

Each driver has an expiration date which is typically the same as the expiration property of the license feature. In some cases such as beta testing agreements, individual drivers may be set to expire where the main license file is non-expiring.

device.limit

This attribute designates a license limit on the number of devices which may be added to this specific driver network in the station database. Above this limit, any added device component (and all its child components) will be in fault.

This limit has no impact on the actual physical limitation of a field bus. For example just because the lonworks feature is set to device.limit="none", this does not mean that you can exceed the normal limit of 64 devices per segment.

history.limit

This attribute limits the number of NiagaraAX histories that can be imported from remote histories (logs or trends) into the station's history space, and/or exported from station histories to appear as histories in remote devices. Above this limit, any added history import descriptor (or history export descriptor) will be in fault, and the associated import/export will not be successful.

point.limit

This attribute designates the maximum number of proxy points that may be added to the station database for a particular driver. Above this limit, any added proxy point will be in fault.

schedule.limit

This attribute limits the maximum number of NiagaraAX schedules that can be imported from remote schedules into the station's database, and/or exported from station schedules to appear as schedules in remote devices. Above this limit, any added schedule import descriptor (or schedule export descriptor) will be in fault, and the associated import/export will not be successful.

parts

This is an alphanumeric part code which is automatically assigned when generating the license file and is for Tridium internal use.

Driver types

Each driver type is enabled by a separate feature element (or line, starting with *name* attribute), and has common [Driver attributes](#) as previously described.

Note: *New NiagaraAX drivers are continually developed and offered as products. This section includes some, but not all drivers available. It is included in this section to illustrate how driver features appear in licenses.*

Alphabetically, driver types listed here include [aaphp](#), [aapup](#), [bacnet](#), [bacnetws](#), [dust](#), [fileDriver](#), [lonworks](#), [modbusAsync](#), [modbusSlave](#), [modbusTcp](#), [modbusTcpSlave](#), [obixDriver](#), [opc](#), [niagaraDriver](#), [rdbDb2](#), [rdbOracle](#), [rdbSqlServer](#), and [snmp](#).

aaphp

Enables the American Auto-Matrix Public Host Protocol (PHP) driver. The serial feature is also required.

aapup

Enables the American Auto-Matrix Public Unitary Host (PUP) driver. The serial feature is also required.

bacnet

Enables functionality of the BACnet driver for BACnet/Ethernet and BACnet/IP. If a QNX-based JACE, other features can be added to enable BACnet MS/TP communications over serial ports: [mstp](#) and [serial](#).

```
<feature name="bacnet" expiration="2009-03-31" device.limit="none"
  export="true" history.limit="none" point.limit="none" schedule.limit="none"
  parts="AX-DEMO" />
```

export export="true" - When set to "true" this field enables BACnet server operation. When the field is set to "false" only BACnet client operation is permitted.

Note: *When BACnet export is enabled, any station histories and/or schedules that are exported to BACnet do not count towards any [history.limit](#) or [schedule.limit](#) values in the license (if any).*

bacnetws

Provides added functionality as a *BACnet Supervisor* as described in the BACnet Specification B-OWS device profile, and is for PC platforms only (not JACE platforms). The [bacnet](#) feature is also required in the license. More details are available as an appendix in the *NiagaraAX BACnet Guide*.

dust

Enables the Dust Wireless Mesh driver. Details are available in the *NiagaraAX Dust User Guide*.

fileDriver

Enables the File driver, used to import comma or tab delimited text files and convert into Niagara Histories.

lonworks

Enables the Lonworks driver. Utilizing the driver also requires a LON interface on the JACE controller. Some JACE controller models require an optional Lonworks interface card to be installed. More details are available in the *NiagaraAX Lonworks Guide*.

modbusAsync

Enables the Modbus Master Serial driver. The JACE controller operates as the Modbus Master device communicating via an available serial port using either Modbus RTU or Modbus ASCII. The [serial](#) feature is also required.

modbusSlave

Enables the Modbus Slave Serial driver. The JACE controller operates as a Modbus Slave communicating via an available serial port using either Modbus RTU or ASCII to a Modbus Master device. The [serial](#) feature is also required.

modbusTcp

Enables the Modbus Master TCP driver. The JACE controller operate as a Modbus Master device communicating via Modbus TCP/IP.

modbusTcpSlave

Enables the Modbus Slave TCP driver. The JACE controller operates as a Modbus Slave device communicating via Modbus TCP/IP.

obixDriver

Enables the oBIX driver. The driver supports the oBIX protocol, which is M2M (Machine-to-Machine) communications via XML over TCP/IP.

```
<feature name="obixDriver" expiration="2009-03-31" device.limit="none"
export="true" history.limit="none" point.limit="none" schedule.limit="none"
parts="AX-DEMO" />
```

export export="true" When set to "true" this field enables oBIX server operation. When the field is set to "false" only oBIX client operation is permitted.

opc

Enables the OPC client driver, and is only available on Windows-based platforms because of the protocol's dependency of Windows.

niagaraDriver

Enables communication via the Fox protocol to other NiagaraAX stations, and allows creation of a NiagaraNetwork, including proxy points, importing/exporting histories and schedules, and routing alarms.

rdbDb2

Enables the Relational Database Driver using the IBM DB2 database format. This driver allows exporting of histories from the NiagaraAX station to an IBM DB2 database. The driver does not include the DB2 software, which must be purchased separately from a third party source.

```
<feature name="rdbDb2" expiration="never" parts="ENG-WORKSTATION" />
```

rdbOracle

Enables the Relational Database Driver using the Oracle database format. This driver allows exporting of histories from the NiagaraAX station to an Oracle database. The driver does not include the Oracle software, which must be purchased separately from a third party source.

```
<feature name="rdbOracle" expiration="never" parts="ENG-WORKSTATION" />
```

rdbSqlServer

Enables the Relational Database Driver using the Microsoft SQL database format. This driver allows importing and exporting of histories to and from the NiagaraAX station, and to and from a Microsoft SQL database. The driver does not include the Microsoft SQL software, which must be purchased separately from a third party source. The driver does work with the MSDE version which is free from Microsoft; however, the normal Microsoft imposed limitations on the MSDE version still apply.

```
<feature name="rdbSqlServer" expiration="never" history.limit="10" history-
Import="true" parts="ENG-WORKSTATION" />
```

snmp

Enables the SNMP (Simple Network Management Protocol) driver, which allows sending and receiving SNMP messages.

```
<feature name="snmp" expiration="2008-03-31" device.limit="none"
history.limit="none" point.limit="500" schedule.limit="none" parts="AX-DEMO" />
```

Applications

Alphabetically, application types include [crypto](#), [eas](#), [email](#), [genericAppliance](#), [provisioning](#), [station](#), [tenantBilling](#), [web](#), and [workbench](#). However, applications [station](#), [web](#), and [workbench](#) have special importance, and are summarized first.

station

Enables a station to be run, and is typically present in any JACE platform, as well as an AxSupervisor.

```
<feature name="station" expiration="2008-03-31" resource.limit="none"
parts="AX-DEMO" />
```

The station feature may not be present in a license for an engineering workstation (PC), unless specifically ordered with it.

resource.limit `resource.limit="none"` - If the `resource.limit` flag is specified (in kRUs), then the station displays a warning on startup if the actual resource units exceed the limit resource units. If the limit is exceeded by 110% then the station will not boot at all. This limit is normally only specified on SoftJACE license files.

web

The web feature must be present to start the WebService in a running station (to access the web server via an HTTP connection). If not licensed, the server is set to fault with appropriate `faultCause`. If the feature is enabled, then WebServlets and spy pages are always available.

Note: *Full Workbench can connect to a station (via Fox connection) even if the web feature is disabled.*

```
<feature name="web" expiration="2008-03-31" ui="true" ui.wb="true"
ui.wb.admin="true" parts="AX-DEMO" />
```

ui `ui="true"` - If the `ui` flag is false, then all access to ServletViews via `/ord` are disabled (with exception to spy pages).

ui.wb `ui.wb="true"` - If the `ui.wb` flag is false, then all user accounts using an `WbProfile` are disabled (including thin client views, but with exception to spy pages). Only thin client accounts operate when `ui.wb` is false; other accounts do not automatically degrade. Views which aren't licensed return 403 Forbidden.

ui.wb.admin `ui.wb.admin="true"` - If the `admin` flag is false, then all views which have an `admin` flag set in their required permissions are unavailable using the `wb` applet.

workbench

The workbench feature must be present to start the full version of Workbench (for example, a copy of Tridium's WorkPlaceAX or an OEM-specific Workbench-based application). If the `admin` flag is false, then all views requiring `admin` access are unavailable. This feature is included for PC platforms only, with the sole exception of a SoftJACE.

```
<feature name="workbench" expiration="2008-03-31" admin="true" parts="AX-DEMO" />
```

crypto

Enables "secure socket layer" (ssl) operation.

```
<feature name="crypto" expiration="never" ssl="true" parts="ENG-WORKSTATION" />
```

eas

This feature enables the Vykon Energy Suite application and the associated reports, data points and meters.

```
<feature name="eas" expiration="2008-03-31" allCostReports="true"
allE2Reports="true" brand="vykon" costMeter.limit="none"
dataPoint.limit="none" parts="AX-DEMO"/>
```

allCostReports allCostReports="true" - If set to "true" all Cost Profiler Reports are enabled. When set to "false" there are additional feature items for each of the specific Cost Profiler Reports that are enabled.

allE2Reports allE2Reprot="true" - If set to "true" all E2 Profiler Reports are enabled. When set to "false" there are additional feature items for each of the specific E2 Profiler Reports that are enabled.

costMeter.limit costMeter.limit="none" - Designates the maximum number of Cost Profiler Meters that may be configured in the VES Application, where "none" means unlimited.

dataPoint.limit dataPoint.limit="none" - Designates the maximum number of E2 Profiler Points that may be configured in the VES Application, where "none" means unlimited

email

This features enables the NiagaraAX station to send email messages to an SMTP server. If the feature is not present then the service and all its incoming and outgoing accounts are marked as fault. No email can be sent or received if not licensed.

```
<feature name="email" expiration="2008-03-31" parts="AX-DEMO"/>
```

genericAppliance

This feature enables the NiagaraAX Generic Appliance Application.

```
<feature name="genericAppliance" expiration="never" vendor.name="" parts="GA-
GENERIC"/>
```

provisioning

Enables the operation of provisioning, typically used to automate routine maintenance of a NiagaraAX system such as JACE software upgrades, file distribution and backups. It applies to an AxSupervisor platform only. In AX-3.1 and AX-3.2, provisioning was done with the ProvisioningService, sourced from the **provisioning** module. Starting in AX-3.3, provisioning moved to a BatchJobService and "network extension model (e.g. a "ProvisioningExt" under the NiagaraNetwork), sourced respectively from modules **batchJob** and **provisioningNiagara**.

```
<feature name="provisioning" expiration="2008-03-31" parts="AX-DEMO"/>
```

tenantBilling

Enables the NiagaraAX Tenant Billing Application.

```
<feature name="tenantBilling" expiration="never" tenant.limit="none" parts="S-
TBS-AX"/>
```

tenant.limit tenant.limit="none" - Designates the maximum number of tenants that may be configured in the station database, where "none" means unlimited.

APPENDIX B

Time Zones and NiagaraAX

Platform configuration of a NiagaraAX host includes specifying its time zone. This affects both real time clock accuracy used in station control, as well as how timestamps appear in items like histories and alarms. This appendix provides details on time zone selection in all revisions of NiagaraAX, as well as the AX-3.3 and later improvements based upon a “historical time zone database.”

The following main sections are included:

- [Time zones and terminology](#)
- [Selecting a time zone in NiagaraAX](#)
- [About timezones.xml](#) (pre-AX-3.3)
- [About the historical time zone database \(AX-3.3 and later\)](#)

Time zones and terminology

A time zone is a region in the world that uses the same standard time, often referred to as the *local time*. There are many different time zones, owing to the combinations of geographic locations and political/cultural differences. Time zones calculate their local time as an offset from [UTC](#) (Coordinated Universal Time). In addition, many time zones apply [DST](#) (Daylight Saving Time).

UTC

Coordinated Universal Time (UTC) is the recognized atomic-clock standard of reference time, largely replacing GMT (Greenwich Mean Time) as reference time. Time zones are commonly expressed as negative or positive offsets from UTC time.

DST

Daylight Saving Time (DST) is used as a means of maximizing daylight hours during normal waking hours, and is used by many (but not all) time zones. DST is a twice-yearly event acting upon local time, as follows:

- Start of DST adds an offset (typically 1 hour) to local time. During this period of the year, local time may be called “daylight time.”
- End of DST removes the DST offset from local time. During this period of the year, local time may be called “standard time.”

Any time zone using DST has specific rules that define the exact days and times when DST starts and ends. These rules vary widely from zone to zone, since DST policies are set by national and regional governments. Also, DST policies are subject to *change* for this same reason—as in the recent 2007 change for all U.S. time zones that observe DST.

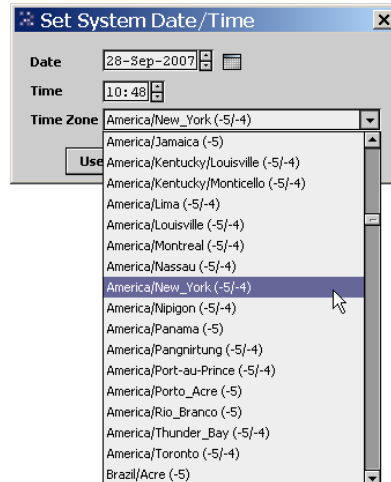
In the 2007 U.S. DST changes, the DST start time was changed to “first Sunday on or after the 8th in March” (from “first Sunday on or after the 1st in April” for 2006 and prior years). The DST end time was changed to “first Sunday on or after the 1st in November” (from “last Sunday in October” for 2006 and prior years).

Note: *A change in DST rules for a time zone can cause issues in NiagaraAX when displaying historical data (histories and alarm records), particularly when applying new (current) DST rules to records collected using prior (old) DST rules. Starting in AX-3.3, these issues have been overcome using a “historical timezones” mechanism. For more details, see [“About the historical time zone database \(AX-3.3 and later\)”](#) on page B-4.*

Selecting a time zone in NiagaraAX

Since the first AX-3.0 release, platform configuration of a NiagaraAX host includes the setting of date and time, which includes specifying its local time zone. Typically, this is done using either a platform connection and the **Platform Administration** view (function “[Change Date/Time](#)”), or possibly after a station is installed and running, using the one of the station’s [PlatformServices](#) views (“Platform Service Container Plugin” or “System Date and Time Editor”).

Figure B-1 Selecting time zone from *Change Date/Time* selection in Platform Administration View



Time zones appear on the selection list using a “Zone ID (hours UTC offset/hours [DST + UST] offset)” format. Some examples:

```
America/Chicago (-6,-5)
Europe/Berlin (+1,+2)
Asia/Tokyo (+9)
```

Note there is no DST observance in Japan, so the selection with zone ID “Asia/Tokyo” shows only the UTC offset of +9 hours.

This selection list of available time zones is from a “*time zone database*” that *resides on that host*.

Depending on NiagaraAX release level, this database is sourced differently, as follows:

- Host platforms running AX-3.0, -3.1, or -3.2 use a `!lib/timezones.xml` file. In this database file, for each known time zone there is *one* definition, starting with its zone ID. If necessary, you can edit the definition of one or more time zones in this `timezones.xml` file (or replace with an edited version of this file). See the next section “[About timezones.xml](#)” for more details.
- Host platforms running AX-3.3 or later use a “historical time zone database”, contained in the file `!lib/timezones.jar`. (Although a `timezones.xml` file is also included, it is not used locally.) In this database, a history of *changes* for applicable time zones are stored, such that *multiple definitions* for a time zone may exist, including past definitions as well as its current definition. Time zones in this database are *not* user editable. However, this time zone database offers advantages for NiagaraAX jobs where accrued histories or alarms have spanned across different time zone definition eras. For more details, see “[About the historical time zone database \(AX-3.3 and later\)](#)” on page B-4.

About timezones.xml

The `timezones.xml` file used by AX-3.0, AX-3.1, and AX-3.2 hosts provides an easily read and edited *single* definition for each time zone, where all values are contained within quotes “*value*”.

For example, consider the following entry for one time zone:

```
<zone id="America/Anchorage" utcOffset="-9h">
  <display name="Alaska Standard Time" short="AKST" dstName="Alaska Daylight Time" dstShort="AKDT"/>
  <dst savings="1h">
    <start time="2:00" weekday="sunday" month="march" week="second"/>
    <end time="2:00" week="first" weekday="sunday" month="november"/>
  </dst>
</zone>
```

where attributes and example values are:

- `zone id="America/Anchorage"` : Name in drop-down selection list when picking a time zone.
- `utcOffset="-9h"` : Time added to UTC for this zone, typically in (h)ours, may be negative, positive,

or 0 (i.e. none).

- `display name="Alaska Standard Time"` : Seen somewhere (but where?) when DST *not* in effect.
- `short="AKST"` : Seen within timestamps (e.g.,alarms and histories) when DST *not* in effect.
- `dstName="Alaska Daylight Time"` : Seen somewhere (but where?) when DST *is* in effect.
- `dstShort="AKDT"` : Seen within timestamps (e.g.,alarms and histories) when DST *is* in effect.
- `dst savings="1h"` : All dst parameters exist only if the time zone has DST. If so, the savings attribute is the DST adjustment amount, typically "1h" (1 hour), in very few time zones is "0.5h".
 - `start time="2:00" weekday="sunday" month="march" week="second" /` : Specifies the time and day when DST begins, often with "week, weekday, month" method (that is, "Nth weekday", as shown here.
 - `end time="2:00" week="first" weekday="sunday" month="november" /` : Specifies the time and day when DST ends, often with a "week, weekday, month" method (that is, "Nth weekday", as shown here.

Note that there are [DST start and end syntax variations](#) different from the ones above.

DST start and end syntax variations

Sometimes a time zone with DST will use a *different* syntax to specify a DST start or stop day, versus the "Nth weekday" syntax shown in the previous [timezones.xml](#) example.

For example, the Baghdad time zone definition uses day ("Exact day") instead of "Nth weekday" to specify a particular date, in this case for the 1st of the month:

```
<zone id="Asia/Baghdad" utcOffset="3h">
  <display name="Arabia Standard Time" short="AST" dstName="Arabia Daylight Time" dstShort="ADT"/>
  <dst savings="1h">
    <start time="3:00 standard" month="april" day="1"/>
    <end time="3:00 standard" month="october" day="1"/>
  </dst>
</zone>
```

A variation is where weekday and day are both used (but *not* week) and the day numeric value includes trailing . . . to indicate "on or after (that day number)," as in time zone definition below. In this case, start time is evaluated as "on the Friday on or after the 26th of March."

```
<zone id="Asia/Tel_Aviv" utcOffset="2h">
  <display name="Israel Standard Time" short="IST" dstName="Israel Daylight Time" dstShort="IDT"/>
  <dst savings="1h">
    <start time="2:00" weekday="friday" month="march" day="26..." />
    <end time="2:00" month="september" day="13"/>
  </dst>
</zone>
```


A similar "on or before" variation can be specified using a leading . . . before a day number, for example, `day="...12"`. Of course many time zones do not even observe DST, so their definitions are noticeably shorter, for example the following one for Honolulu:

```
<zone id="Pacific/Honolulu" utcOffset="-10h">
  <display name="Hawaii Standard Time" short="HST"/>
</zone>
```

Using Workbench to edit and transfer timezones.xml

Remember the [timezones.xml](#) used by any Niagara AX host (JACE) is the one in its *own !lib* folder, so to change it you typically edit a *local copy* (on your Workbench PC) and then *transfer this file* to that location on the remote JACE, using the platform view File Transfer Client.

The following is a quick summary of how to do this using Workbench:

- Step 1 In the Workbench nav tree, expand **My File System** and then **Sys Home, lib** folder.
 - Step 2 In the **lib** folder, double-click `timezones.xml` to open it in the Text File Editor.
 - Step 3 Make whatever changes are needed in the time zone(s) you will pick from, and click the Save tool  when done.
 - Step 4 In Workbench, open a platform connection to the target JACE, and select the File Transfer Client.
 - Step 5 In the File Transfer Client, open both (left and right) sides to list files in the **!lib** folder.
 - Step 6 Transfer your locally edited `timezones.xml` (left side) to the JACE on the right. A "Replace File?" popup dialog will appear, showing a different CRC value. Click **Yes** to transfer the file.
- The time zone database is now updated in the JACE.

Note: Any change to `timezones.xml` is not effective until after the station is restarted.

About the historical time zone database (AX-3.3 and later)

Starting in AX-3.3, any NiagaraAX host uses a time zone database with “historical perspective,” such that display of a station’s timestamped data (histories and alarms) collected in time zones under “prior rules” (typically DST-related) will always display with the original (and correct) collected time.

Issue example and fix

You have a JACE that has been running since AX-3.0, installed in a U.S. time zone that observes DST. Included in the AX-3.0 builds was a `timezones.xml` with “old” DST rules. See the “DST” section for related details. On it you have a station running, that includes an original history for a boolean point with two records like this:

```
01-Nov-06 8:00 AM EST {} {ok} On
01-Nov-06 5:00 PM EST {} {ok} Off
```

The timestamp values shown are correct, this was a piece of equipment that came ON at 8:00am and OFF at 5:00pm, and on November 1st of 2006 it was indeed “Eastern Standard Time” (EST), as Daylight Savings Time had already ended on the last Sunday in October.

Now you upgrade the JACE to AX-3.2, which uses updated 2007 rules in its `timezones.xml` file. Afterwards, when looking at the same two records for the example history, you see:

```
01-Nov-06 9:00:11 AM EDT {} {ok} On
01-Nov-06 6:00:13 PM EDT {} {ok} Off
```

These timestamp values are incorrect, as the DST offset of 1 hour was “retroactively” applied using the new DST rules (even though these rules were not actually in effect at this time). However, those same new rules are required during this period in 2007 to correctly change the system clock and maintain accurate 2007 timestamps.

After upgrading the JACE to AX-3.3, which uses the `timezones.jar` (historical time zone database) instead of `timezones.xml`, when you look again at the same two records for the example history, you see:

```
01-Nov-06 8:00:11 AM EST {} {ok} On
01-Nov-06 5:00:13 PM EST {} {ok} Off
```

The timestamp values are correct, as the historical rules were used instead of the current (new) rules.

Parts of the historical time zone database

The AX-3.3 and later historical time zone database is sourced from the public “Olson Time Zone Database,” and updated (synchronized to it) upon each NiagaraAX build. It is implemented using a “`timezones.jar`” file that contains histories of changed rules for any so-affected timezones, along with changes to the NRE (Niagara Runtime Environment) that support this new method.

There are 2 associated files with historical time zones in an AX-3.3 and later host’s `!lib` folder, described as follows:

- `timezones.jar`: The time zone database, in Java archive format. Contains a collection of binary files, one representing each time zone. Upon each build of NiagaraAX, this `timezones.jar` file is updated (synchronized) to the Olson Time Zone Database to maintain historical accuracy.
- `system.properties`: The file responsible for loading various system settings at NRE (Niagara Runtime Environment) boot time. This file now contains 2 keys pertaining to historical time zones:
 - `niagara.timezone.dbCache`: The maximum number of zones to remain cached in memory when querying the database for a particular zone. This caching is done in an “LRU” fashion, vs. hitting the database repeatedly for the same zones. This method provides performance gains.
 - `niagara.timezone.eraTolerance`: The number of milliseconds to wait before loading a new historical time zone era. The higher the number, the better the performance (yet lower the accuracy). The reverse is true for higher numbers

Please note that time zones in this database are not user editable, unlike with the earlier `timezones.xml` database.

Updating a historical time zone database

Typically, you use the normal release/build upgrade process for a JACE to update its historical time zone database. However, in cases where an updated version of this database becomes available, and you want to *maintain* the current build in an AX-3.3 or later JACE, you can simply *transfer* the newer `timezones.jar` file to the host’s `!lib` folder.

Note: *As when making changes to the `timezones.xml` file in an earlier rev. host (AX-3.0, AX-3.1, AX-3.2), after updating the historical time zone database files, you must restart the station on that host for the updated time zone database to become effective.*