

Technical Document

OpenADR - DRAS Simple Client Driver Guide

March 1, 2011



Confidentiality Notice

The information contained in this document is confidential information of Tridium, Inc., a Delaware corporation ("Tridium"). Such information and the software described herein, is furnished under a license agreement and may be used only in accordance with that agreement.

The information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and, except as permitted under the below copyright notice, is not to be released to, or reproduced for, anyone else.

While every effort has been made to assure the accuracy of this document, Tridium is not responsible for damages of any kind, including without limitation 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 latest product specifications can be found by contacting our corporate headquarters, Richmond, Virginia.

Trademark Notice

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} Framework, and Sedona Framework 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.

Copyright and Patent Notice

This document may be copied by parties who are authorized to distribute Tridium products in connection with distribution of those products, subject to the contracts that authorize such distribution. It may not otherwise, 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.

© Tridium, Inc. 2011.

All rights reserved. The product(s) described herein may be covered by one or more U.S or foreign patents of Tridium.

CONTENTS

| | |
|--|------------|
| Confidentiality Notice | i |
| Trademark Notice | i |
| Copyright and Patent Notice | i |
| Preface | iii |
| Document change log | iii |
| Related documentation | iii |
| OpenADR – DRAS Simple Client Driver overview | 1–1 |
| Target platform and software requirements | 1–2 |
| Limitations and known issues | 1–2 |
| Important terms and abbreviations | 1–3 |
| Installing the OpenADR – DRAS Simple Client Driver..... | 2–1 |
| Install the Crypto Service | 2–1 |
| Install the DRAS Network | 2–3 |
| Install the DRAS Device | 2–3 |
| DRAS Simple Client Driver Architecture..... | 3–1 |
| About the DRAS Network | 3–1 |
| About the DRAS Device | 3–3 |
| About the DRAS Device Manager view | 3–4 |
| About the DRAS Program | 3–4 |
| About the DRAS Event | 3–5 |
| Dras Component and Plugin Guides | 4–1 |
| Components in the dras module | 4–1 |
| Plugins in the dras module | 4–1 |
| dras-DrasNetwork | 4–1 |
| dras-DrasDevice | 4–1 |
| dras-DrasSslProvider | 4–1 |
| dras-DrasProgram | 4–2 |
| dras-DrasDeviceManager | 4–2 |

PREFACE

Preface

Document change log

Following are a list of document changes

- March 1, 2011, Initial release document.

Related documentation

The following documents are related to the content in this document and may provide additional information on the topics it covers:

- *NiagaraAX-3.x User Guide*
- *NiagaraAX-3.x Drivers Guide*

CHAPTER 1

OpenADR – DRAS Simple Client Driver overview

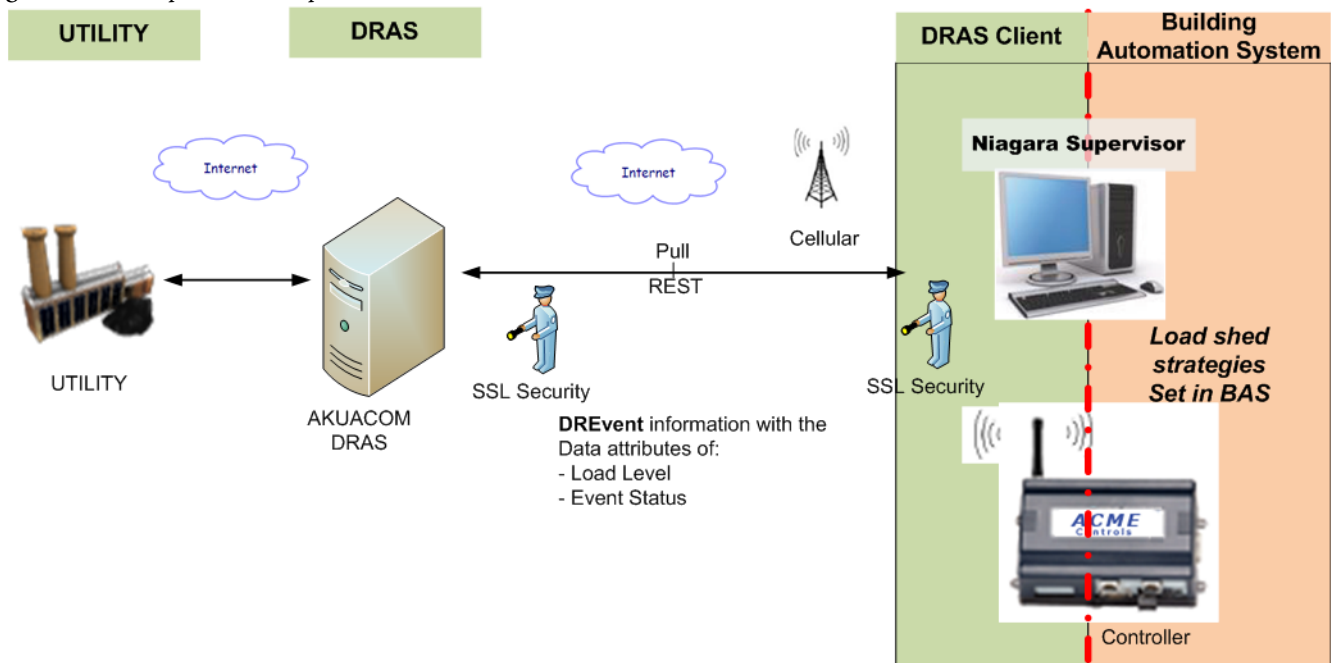
This guide is written for users that are familiar with OpenADR specifications. Use this document to help you install and configure the OpenADR – DRAS Simple Client Driver.

Open Automated Demand Response (OpenADR) is an Open specification that describes demand response management. This specification helps achieve efficient energy usage by providing an infrastructure where energy utility services send Demand Response (DR) signals to consumers. The DRAS Simple Client Driver is an OpenADR driver for Niagara that can communicate with the Demand Response Automated Server (DRAS) at the Akuacom energy utility company (California, USA). When properly configured, the driver receives demand response signals from the server.

Consumers choose to be *participants* in the event communication process. The communication process helps participants understand the energy rates and dynamic pricing that are based on peak hour load. Participants may decide to shed energy loads based on their own needs and the current cost of electricity. The OpenADR specification does not specify the implementation requirements but rather provides an open specification for the interested parties to implement.

The following illustration shows an example scenario where a utility communicates demand - response information with the DRAS server. The DRAS communicates with a NiagaraAX station client participant, sharing Load Level and Event Status data.

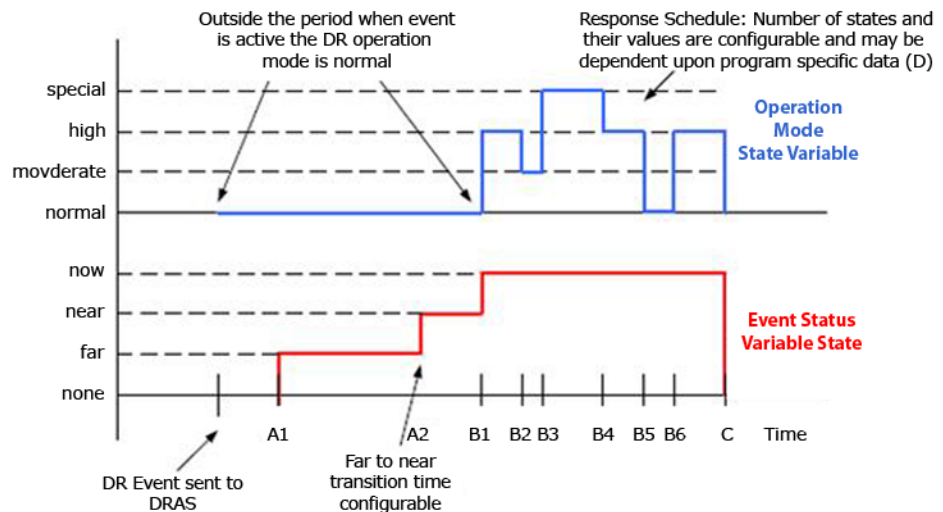
Figure 1-1: Example DRAS Simple Client Driver communications



Based on this simple communication, the participant can choose to implement load shed strategies. These load shed strategies are controlled by the participant's building automation system.

The following figure shows some different Operation Mode transitions as they occur during the ACTIVE period of the DR event. These transitions of the Operation Mode value could be caused either by the EventInfoInstance of the DR event changing values or by a *transition* from one OperationStateSpec to another within a ResponseSchedule.

Figure 1-2 Example DR Event state mode



DR Event State Model (Simple Client View)

Source: Lawrence Berkeley National laboratory/ Akuacom

For more information about event status, see the Event Status property information in the *About the DRAS Event* section.

To learn more about OpenADR refer to the following:

- **OpenADR website:**
<http://www.openadrcollaborative.org/>
- **Open Automated Demand Response Communications Specification (Version 1.0)**
<http://openadr.lbl.gov/pdf/cec-500-2009-063.pdf>

Target platform and software requirements

- **NiagaraAX Platform**
This driver may be used in NiagaraAX-3.5.25.2 (or higher) stations that are running on either QNX® or Microsoft® Windows® based controller platforms.

Limitations and known issues

- This driver has been developed and tested against Akuacom servers only.
- The event XML format used by this driver for communication is specific to Akuacom DRAS servers.

Important terms and abbreviations

The following terms are important for understanding the information in this document.

Demand Response (DR)

Changes in electric use by demand-side resources from their normal consumption patterns in response to changes in the price of electricity, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized.

Demand Response Automation Server (DRAS)

The OpenADR-compliant demand response server that is owned by Akuacom.

ISO

Independent system operator.

RTO

Regional transmission organization.

Duration of Event

The length of an Emergency or Economic Demand Response Event, in hours.

Economic Demand Response Event

An event in which the demand response program sponsor directs response to an economic market opportunity, rather than for reliability or because of an emergency in the energy delivery system.

Emergency Event

An abnormal system condition (for example, system constraints and local capacity constraints) that requires automatic or immediate manual action to prevent or limit the failure of transmission facilities or generation supply that could adversely affect the reliability of the Bulk Electric System.

Emergency Demand Response Event

The period of time during which participants in a Demand Response Program must reduce load. The Emergency Demand Response Event is announced by the program sponsor in response to an Emergency Event declared by it or by another entity such as a utility or RTO/ISO. Demand Response Program sponsors, utilities and RTO/ISOs typically declare these emergency events.

Program

The demand response arrangements between retail or wholesale entities and their retail or wholesale customers. Examples of these arrangements include: critical peak pricing, critical peak pricing with load control, direct load control, interruptible load, load as a capacity resource, regulation, non-spinning reserves, spinning reserves, demand bidding and buy-back, time of use pricing, real-time pricing, system peak response transmission tariff, peak time rebate, and emergency demand response.

Participant

An organization that buys and sells under the rules and guidelines of a demand response program.

REST

Representational State Transfer - an architectural style used in web based communication.

CHAPTER 2

Installing the OpenADR – DRAS Simple Client Driver

You can install the OpenADR – DRAS Simple Client Driver under a NiagaraAX station Drivers node.

The DRAS server and client communication uses secured socket layer protocol (SSL). The DRAS module depends on the *crypto* module, which helps establish secured communication.

The following sections describe how to install and configure the OpenADR – DRAS Simple Client Driver. Perform the tasks in the order listed below:

- “Install the Crypto Service” on page 2-1
- “Install the DRAS Network” on page 2-3
- “Install the DRAS Device” on page 2-3

Install the Crypto Service

The Crypto Service (CryptoService) is provided as part of the crypto module and is available in the crypto palette.

PREREQUISITE

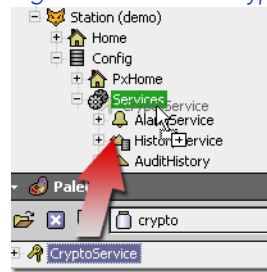
- The target platform must be running NiagaraAX-3.5.25.2 or later.
- The target platform must have the “adr” and “crypto” modules installed.
- The target platform must have the “adr” and “crypto” features in its license.

The following task assumes that the user has established a connection to the target platform using NiagaraAX-3.5.25.2 or later.

TASK

1. From the WorkbenchAX main menu, select **Window > Side Bars > Palette**.
*ADDITIONAL INFORMATION: If the Side Bar pane is not open, select **Window > Side Bars > Show Side Bar** from the main menu.*
2. In the Palette side bar, click the **Open Palette** menu control.
STEP RESULT: The Open Palette dialog box opens.
3. In the Open Palette dialog box, select the “crypto” module and click the **OK** button.
STEP RESULT: The crypto palette displays in the palette side bar.
4. From the crypto palette, drag and drop the CryptoService component onto the target station’s Services node (**Station > Config > Services**).

Figure 2-1 Add the CryptoService under the Services node.



STEP RESULT: The CryptoService component displays under the Services node.

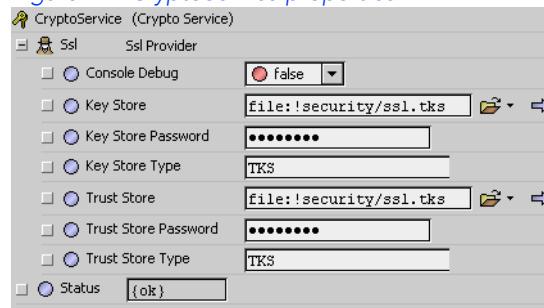
5. In the nav tree, double-click on the CryptoService component.

STEP RESULT: The CryptoService property sheet view displays in the view pane.

6. In the CryptoService property sheet view, expand the Ssl node and check the following properties, as required:

NOTE: For the DRAS Simple Client Driver, default settings, shown below, should never be changed.

Figure 2-2 CryptoService properties.



- **Key Store**
The path to the file that contains the key entries for the DRAS client. By default, this string value is *file:!/security/ssl.tks*, which is under the security folder of the Niagara installation directory.
- **Key Store Password**
The string password by which the keystore file is protected. The password is: “tridium”.
- **Key Store Type**
The type of keystore file. This is preconfigured information. NiagaraAX key store file is of type “TKS”.
- **Trust Store**
The file path containing the certificate entries. The entries in this file validate the secured server to which the client connects. By default this slot points to *SSL.tks* file located under the security folder of the Niagara installation directory.
- **Trust Store Password**
The string password by which the trust store file is protected. The default password is “tridium”.
- **Trust Store Type**
Type of the trust store file. This is preconfigured information. Niagara trust store file is of type “TKS”.

7. With all properties set as desired, click the **OK** button.

STEP RESULT: The property settings are saved.

RESULT:

The CryptoService installation is complete.

Install the DRAS Network

The DRAS Network is provided as part of the dras module and is available in the DRAS palette or from the Driver Manager's New dialog box.

PREREQUISITE

- The target platform must be running NiagaraAX-3.5 or later.
- The Crypto Service is installed and configured.
- A platform connection exists between your WorkbenchAX platform and the target platform.

TASK

1. In the WorkbenchAX nav tree, double-click on the Drivers node.
STEP RESULT: The Driver Manager view displays.
 2. At the bottom of the Driver Manager view, click the **New** button.
STEP RESULT: The New dialog box displays.
 3. In the New dialog box, choose Dras Network from the Type to Add option list, enter "1" in the Number to Add field, and click the **OK** button.
STEP RESULT: A second New Dialog box displays.
 4. In the New Dialog box, edit the name of the network, if desired, set the Enabled property to true and click the **OK** button.
STEP RESULT: The DrasNetwork entry appears in the Driver Manager view and the DrasNetwork component appears in the nav side bar under the Drivers node.
-

RESULT:

The DrasNetwork installation is complete.

Install the DRAS Device

The Dras Device component is provided as part of the dras module and is available in the DRAS palette or from the Dras Device Manager's New dialog box.

PREREQUISITE

- The target platform must be running NiagaraAX-3.5 or later.
- The DrasNetwork is installed.
- A platform connection exists between your WorkbenchAX platform and the target platform.

NOTE: The DRAS client is a "device-level" component. Each DRAS device represents a participant configured in the DRAS server.

TASK

1. In the WorkbenchAX nav tree, double-click on the DrasNetwork node.
STEP RESULT: The Dras Device Manager view displays.
2. At the bottom of the Dras Device Manager view, click the **New** button.
STEP RESULT: The New dialog box displays.
3. In the New dialog box, choose Dras Device from the Type to Add option list, enter "1" in the Number to Add field, and click the **OK** button.
STEP RESULT: A second New Dialog box displays.

4. In the New Dialog box, edit the name of the device, if desired, set the Enabled property to true and click the **OK** button.

STEP RESULT: The DrasDevice entry appears in the Dras Device Manager view and the DrasDevice component appears in the nav side bar under the Drivers node.

RESULT:

The DrasDevice is installed. Default settings should be sufficient for the device to be enabled. For more information about property settings, see “Related Links”.

RELATED LINKS:

“About the DRAS Device” on page 3-3

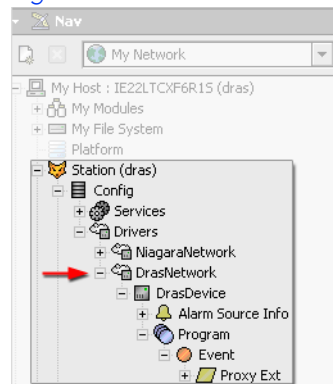
CHAPTER 3

DRAS Simple Client Driver Architecture

The DRAS client uses the standard NiagaraAX network architecture. See *About Network architecture* in the *NiagaraAX Drivers Guide* for more details. The DRAS components are the station interface to DR events coming from DRAS servers. The DRAS device forms a “participant”, which resides under the DRAS Network which in turn is located under the station’s “Drivers” container.

Hierarchically, the component architecture is: DRAS Network, DRAS Device, DRAS Program and DRAS Event.

Figure 3-1 Dras network hierarchy in the nav tree



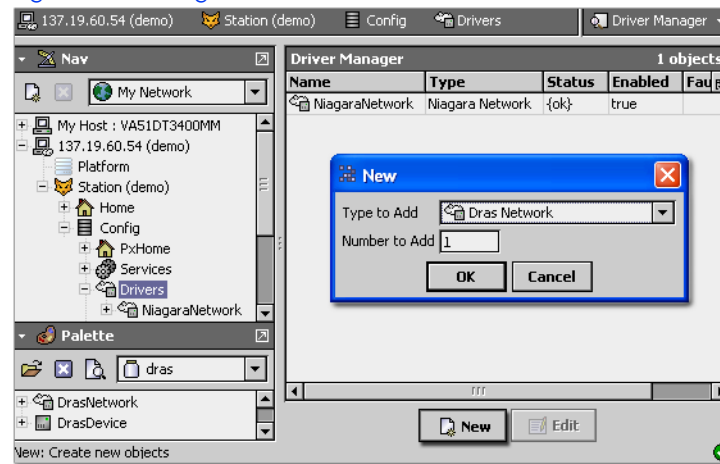
The following sections describe the major parts of the DRAS architecture:

- *About the DRAS Network*
- *About the DRAS Device*
- *About the DRAS Device Manager view*
- *About the DRAS Program*
- *About the DRAS Event*

About the DRAS Network

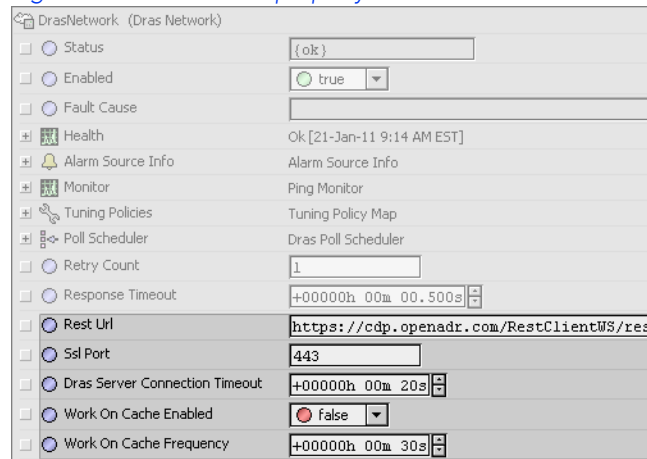
The DRAS Network is the top-level container component for the DRAS client in a NiagaraAX station. You can add the DRAS network from the “Driver Manager” view, using the **New** button. You can also copy the DRAS network from the “dras” palette onto Drivers node in the nav tree.

Figure 3-2 Adding the Dras Network



The DRAS Network component has most of the typical properties that other network components share. For more details about the common properties, see “Common network components” in the *NiagaraAX Drivers Guide*.

Figure 3-3 DrasNetwork property sheet view



In addition to the common properties, note the following properties:

- Rest Url**
 This property has special importance because it represents the secured URL to connect to the DRAS server. The connection uses the Representational State Transfer (REST) software architectural style.
- Ssl Port**
 Port 443 is designated for standard SSL communications.
- Dras Server Connection Timeout**
 This property specifies the maximum time to wait for a response after a basic message request before determining a failure.
- Work on Cache Enabled**
 The purpose of this property is to provide a way to update point values when communication with the server is not available. Cached event op mode schedules are files that have been previously communicated to and stored in on the client. When set to true, this property allows for the use of these cached values. Cached values are used at an interval set by the Work on Cache Frequency value.
- Work on Cache Frequency**
 This is the frequency at which the cached schedule values are used.

- **DRAS Network status**
As with most “field bus” drivers, the status of a DRAS Network is either the normal “ok” or less typical “fault” (fault might result from a licensing error). The Health property contains historical timestamp properties that record the last network status transitions from “ok” to any other status. The “Fault Cause” property further explains any fault status.
- **DRAS Network monitoring**
The DRAS Network’s monitor routine verifies child DRAS Client component(s)—the “pingable” device in the DRAS driver. For general information, see “About Monitor” in the *NiagaraAX Drivers Guide*.
- **DRAS Network views**
The DRAS Network’s default view is the DRAS Device Manager, equivalent to the Device Manager in most other drivers. You use this view to add DRAS Device components to the station. See “About the DRAS Device Manager view” on page 3-4. Other standard views are also available on the DRAS Network. However, apart from the DRAS Device Manager, you typically access only its property sheet

About the DRAS Device

The DRAS device represents the client connection to DRAS server through a participant account. Unlike most other drivers, the DRAS device maps to a participant configured in DRAS server. All the devices under the DRAS network connect to the DRAS server with the REST URL configured in the network’s property sheet. As the “device-level” component in the DRAS driver architecture, it is similar to most driver’s device components— see “Common device components” in the *NiagaraAX Drivers Guide* for general information.

The following DRAS properties are available in the DrasDevice property sheet view:

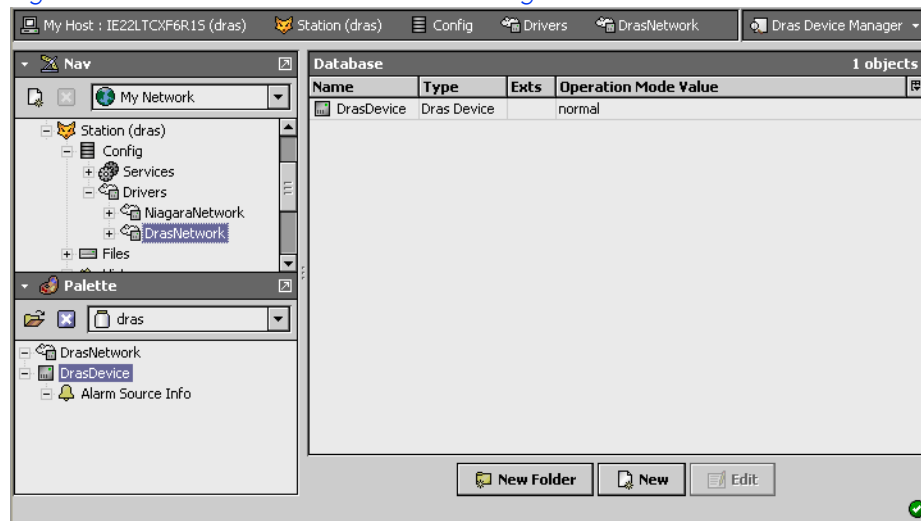
- **Status**
Status of connection to the DRAS server with the configured participant account. Possible status flags include:
 - **ok**
Normal communications, no other status flags
 - **disabled**
Enabled property is set to false, either directly or in DRAS Network. While status is disabled, while disabled ideally no subscription notification should happen.
 - **Fault**
Typically, this a licensing issue.
 - **down**
Error communicating to the DRAS server. If status was previously ok (without changing Participant account configuration), this may mean the server host is now unreachable, or the server program is not running.
- **Enabled**
Either true (default) or false. Can be set directly or in parent DRAS Network. See Status disabled description above.
- **Health**
Contains properties including timestamps of last “ok” time and last “fail” time, plus a string property describing last fail cause.
- **Fault Cause**
If status has fault, describes the cause.
- **Alarm Source Info**
Alarm source info under the DRAS device has no significance in this driver.
- **User Name**
This is the user name of the participant account configured in DRAS server.

- **Password**
This is the password of the participant account configured in DRAS Server. The DRAS device connects to DRAS server with this user name and password.
- **Poll Frequency**
As per the poll frequency, the DRAS device sends a POLL request to the DRAS server to receive the event information. Configure the Normal, Slow, Fast poll frequencies in the DRAS network's poll scheduler as required.
- **Last Poll Time**
This display-only value indicates the time the last poll was made.
- **Poll Wait Count**
This value indicates how many times the poll method can wait for the "Dras Server connection Timeout" to expire before the DrasDevice status is changed to "down".
- **Event Status**
This field represents the current event status of the DR Event received from the server. This can take one of the following values: NONE, FAR, NEAR, ACTIVE. These values are configurable on the DRAS.
- **Operation Mode Value**
This field represents the current operation mode value of the DR event received from the server. This can take one of the following values: NORMAL, HIGH, MODERATE, SPECIAL. These values are configurable on the DRAS.

About the DRAS Device Manager view

As the default view on the DrasNetwork component, the DRAS Device Manager view lets you add one or more "device-level" DRAS Device components to the station database. For general information, see "About the Device Manager" in the *NiagaraAX Drivers Guide*.

Figure 3-4 Dras Network and Dras Device Manager view



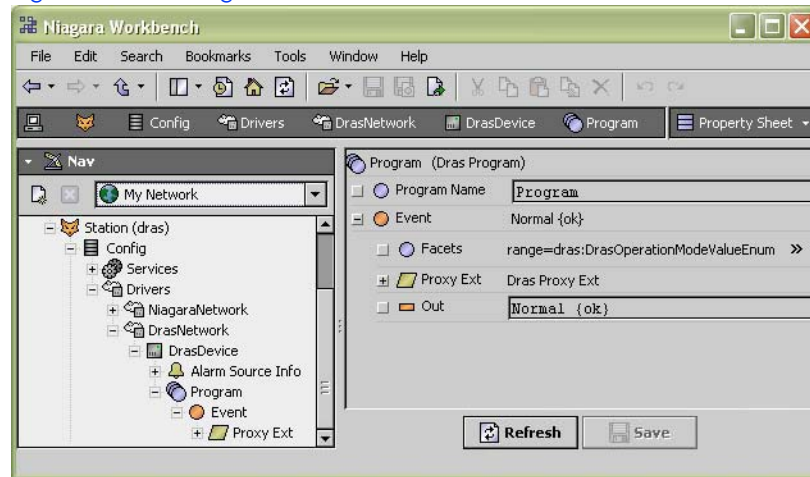
NOTE: Unlike many drivers, the device manager doesn't support device discovery. This is because each DRAS device is modeled as a single participant connection to the DRAS server and a discovery process is not necessary.

About the DRAS Program

This Dras driver supports grouping DR events based on the program. When the device polls for events for the first time, a Program node is created under the device.

NOTE: This Simple OpenADR driver only supports one program and one event under a program.

Figure 3-5 Dras Program

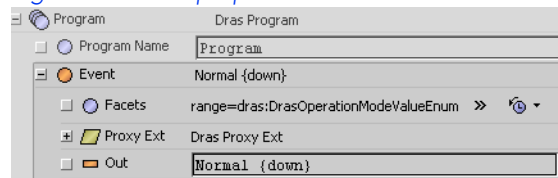


Program has a read only property, the Program Name property. This property's value represents the name of the program that the DR event belongs to. When there are no events coming from the server, this field will have the value "Program". If the Event Status field value is NONE, indicating that there is no pending or active DR event, then this field is not applicable.

About the DRAS Event

When the device polls for a DR event for the first time, a Program object is created with a child component named "Event". Event is modeled as a Niagara Enum point and it has a proxy extension attached to it. The following figure shows the top level Event properties.

Figure 3-6 Event properties



The following properties are displayed under the Event node:

- **Facets**
The standard facet property. The default values are set to match the Out (Operational Mode) values of: normal, high, moderate, and special.
- **Proxy Ext**
The Dras Proxy Ext properties are described in the following section.
- **Out**
This is the "Operation Mode" value. Operation Mode values can be one of the following, as defined by the program for the current time (using the Current Time property value): NORMAL, HIGH, MODERATE, SPECIAL.

Following, is a list of the Dras Event Proxy Ext properties:

- **Event Mod Number**
Modification number of the DR event. Used to indicate if the DR Event has been modified by the Utility. Each time it is modified this number is incremented. If the Event Status field is NONE indicating that there is no pending or active DR event then this field is not applicable.
NOTE: This field is not supported by Akuacom DRAS Server.
- **Event Identifier**
Identifier for the DR event that was created when the DR event was first issued. If the Event Status field is NONE indicating that there is no pending or active DR event then this field is not applicable.

- **DRAS Client ID**
Identifier of the DRAS client that this Event State is being sent to.
- **Event State ID**
This is a transaction ID that is guaranteed to be unique for each instance of the Event State that is generated and sent to the DRAS Client. It is used by the DRAS Client to confirm reception of the Event State from the DRAS.
NOTE: This field is not supported by the Akuacom DRAS Server
- **Schema version**
This is the version number for the Event State schema itself.
- **DRAS name**
Name and/or version of the DRAS that is sending the Event State message.
- **Test Event**
This attribute signifies whether this is a test event or not. Test events may be issued by the Utility/ISO like other DR Events.
- **Offline**
This field signifies whether from the DRAS point of view the DRAS Client is off line. If this field is TRUE then the DRAS Client is off line and if it is FALSE then the DRAS Client is on line.
- **Event Status**
This is the DR event status that is valid at the time specified by the Current Time field.
 - **None**
No events from the DRAS.
 - **FAR**
The DRAS has sent notification of an event but it is not imminent.
 - **NEAR**
The event active state is imminent (typically within 1 minute).
 - **ACTIVE**
The state when an event has started. This is the time between the event start and event end.
- **Current time**
This is the relative time in seconds from the beginning of the DR event window that this particular instance of the Event State is valid for. If the Event Status field is NONE indicating that there is no pending or active DR event then this field is not applicable.
- **Start time**
Date and time for when the DR event begins.
- **End Time**
Date and time for when the DR event ends.
- **Notification Time**
Date and time that the DRAS Client should first be notified of the upcoming DR event.

CHAPTER 4

Dras Component and Plugin Guides

The following sections provide summary descriptions of the components and component views (plugins) that are related to the OpenADR – DRAS Simple Client Driver.

Components in the dras module

The following components are contained in the dras module:


- *dras-DrasNetwork*
- *dras-DrasDevice*
- *dras-DrasSslProvider*
- *dras-DrasProgram*

Plugins in the dras module

The following plugins are contained in the dras module:

- *dras-DrasDeviceManager*


dras-DrasNetwork

 DrasNetwork models a network of Dras objects. It is available in the dras palette.

For more information about Dras components and plugins, see also:

- *Dras Component and Plugin Guides*
- *OpenADR – DRAS Simple Client Driver overview*


dras-DrasDevice

 DrasDevice represents the client connection to DRAS server through a participant account.

For more information about Dras components and plugins, see also:

- *Dras Component and Plugin Guides*
- *OpenADR – DRAS Simple Client Driver overview*


dras-DrasSslProvider

 DrasSslProvider provides the fields used for specifying the file path and security credentials for the Key Store and Trust Store parameters. This object is a child component of the DrasSslProvider component, found in the dras palette.

For more information about Dras components and plugins, see also:

- *Dras Component and Plugin Guides*
- *OpenADR – DRAS Simple Client Driver overview*


dras-DrasProgram

 DrasProgram supports grouping DR events based on the program parameters or values. This component is not located in the dras palette. When the device polls for events for the first time, a Program node is automatically created under the device.

For more information about Dras components and plugins, see also:

- *Dras Component and Plugin Guides*
- *OpenADR – DRAS Simple Client Driver overview*

dras-DrasDeviceManager

 The Dras Device Manager view is the default view of the Dras Network component. This view displays a table view of any added DrasDevice, showing the device Name, Type and Operation Mode Value.

For more information about Dras components and plugins, see also:

- *Dras Component and Plugin Guides*
- *OpenADR – DRAS Simple Client Driver overview*