

Technical Document

ADR Smart Client Driver Guide

May 16, 2012



Niagara^{AX} OpenADR Smart Client Driver Guide

Copyright © 2012 Tridium, Inc.

All rights reserved.

3951 Westerre Pkwy., Suite 350

Richmond

Virginia

23233

U.S.A.

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 Notices

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft, Excel, Internet Explorer, Windows, Windows Vista, Windows Server, and SQL Server are registered trademarks of Microsoft Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. 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. 2012.

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

PREFACE

Preface

Document change log

- May 16, 2012, Initial release document.

Related documentation

- *NiagaraAX-3.x User Guide*
- *NiagaraAX-3.x Drivers Guide*
- *NiagaraAX-3.x OpenADR - DRAS Simple Client Driver Guide*

CHAPTER 1

Overview

This guide is written for users that are familiar with Open Automated Demand Response (OpenADR) specifications. Use this document to help you install and configure the NiagaraAX ADR Smart Client Driver.

The Automated Demand Response Smart 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 (pricing, load, and reliability) from the server and makes the signal information available as Niagara points.

Note: *The functionality of the driver is limited to receiving DR signal and making the signal information available as Niagara points. Defining and implementing control strategies for building automation in response to the DR signal received is outside the scope of this driver.*

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

For more information about demand response events, see the *OpenADR DRAS Simple Client Driver Guide*.

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.
- If the driver is placed in a JACE running within a network firewall, the driver cannot establish communication with the DRAS server. An alternative is to place the JACE in a public network or to open appropriate firewall ports in the WAN.
- Caching op mode and event info schedule will help when there is a temporary loss of connectivity to the server. If the ADR client connection to the server is unavailable for a long time, updating active event might not make sense. Because there is no way for the client to come to know any intermediate changes done in the server during the network failure.
- Updating Active Event with the schedule - will be done only for the currently active event. The driver will not pick the next event automatically from the list of events. For example, for the following events:
 - Event 1- has a start time 8:00 P.M, end time - 9:00 P.M
 - Event 2- has a start time 9:00 P.M, end time - 10:00 P.M

If the network connection gets the last signal at 8:30 p.m., then the driver updates Active Event with the information for "Event1". The driver would not make "Event2" since the next active event will

not occur until 9:00 p.m. This is implemented this way because event order is decided by the server's internal priority mechanism. During network failure, if the event priority is changed in the server, the client operating with old event order will be misleading.

- When the station is restarted during network failure, "Active Event" and all events under Programs node will get cleared. This is for the same reason explained in point 2 above.
- Manual configuration is not fully implemented in DRAS server. As of now only opmode value can be configured in the server. Driver has been developed and tested only against the current functionalities supported by the server
- Akuacom DRAS server does not support configuring all 7 event info types. Driver is developed and tested with 3 event info types -PRICE_ABSOLUTE, PRICE_RELATIVE and LOAD_AMOUNT
- Akuacom DRAS server does not support configuring event info schedule for PRICE_RELATIVE in CPP program
- Driver communicates to DRAS server through REST2 interface. But event cannot be acknowledged using this interface. Currently using REST interface to send acknowledgement back. This can be edited by using Acknowledgement URL in Adr Device.
- The top-node of event xml received by the client is considered to be the event with the most priority. Two different client accounts associated to the same programs-events can receive the events in different order of priority. But the driver sticks to same thumb-rule of keeping top event info - as the high priority event. This means client C1 will display E1 as active where as client C2 will display E2 as active.

Important terms and abbreviations

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

Automated Demand Response (ADR)

A "smart" client driver communicates with a demand response automated server, receives demand response signals and makes the signal information available as Niagara points.

Building Management System (BMS)

The building management system that is controlled through DR signals received by the DRAS client.

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.

HTTP

Hyper Text Transfer Protocol.

HTTPS

Hyper Text Transfer Protocol Secured.

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.

Event

A signal containing electricity pricing/demand information.

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.

SSL

Secure Socket Layer.

CHAPTER 2

Installing the ADR Smart Client Driver

You can install the ADR Smart Client Driver under a NiagaraAX station Drivers node.

The ADR Smart Client connects to a DRAS server. The DRAS server and ADR Smart Client communication uses secured socket layer protocol (SSL). The ADR Smart Client depends on the *crypto* module, which helps establish secured communication.

The following sections describe how to install and configure the ADR Smart Client Driver. Perform the tasks in the order listed below:

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

Install the Crypto Service

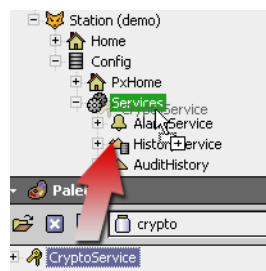
The Crypto Service (CryptoService) is provided as part of the crypto module and is available in the crypto palette.

- 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 tasks assume that the user has established a connection to the target platform using NiagaraAX-3.5.25.2 or later.

- Step 1 From the WorkbenchAX main menu, select **Window > Side Bars > Palette**.
- Note:** If the Side Bar pane is not open, select **Window > Side Bars > Show Side Bar** from the main menu.
- Step 2 In the Palette side bar, click the **Open Palette** menu control.
The *Open Palette* dialog box opens.
- Step 3 In the *Open Palette* dialog box, select the “crypto” module and click the **OK** button.
The crypto palette displays in the palette side bar.
- Step 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.



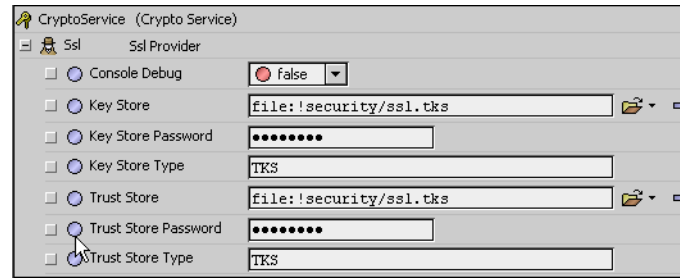
The CryptoService component displays under the Services node.

- Step 5 In the nav tree, double-click on the CryptoService component.
The CryptoService property sheet view displays in the view pane.

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

Note: For the ADR Smart 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 ADR 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”.
- Step 7 With all properties set as desired, click the **OK** button.
The property settings are saved.
The CryptoService installation is complete.

Install the ADR Network

The AdrNetwork is provided as part of the adr module and is available in the Adr palette or from the Driver Manager's **New** dialog box.

- The target platform must be running NiagaraAX-3.5 or later.
 - A platform connection exists between your WorkbenchAX platform and the target platform.
- Step 1 In the WorkbenchAX nav tree, double-click on the Drivers node.
The Driver Manager view displays.
- Step 2 At the bottom of the Driver Manager view, click the **New** button.
The **New** dialog box displays.
- Step 3 In the **New** dialog box, choose Adr Network from the Type to Add option list, enter “1” in the Number to Add field, and click the **OK** button.
A second **New** Dialog box displays.
- Step 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.
The AdrNetwork entry appears in the Driver Manager view and the AdrNetwork component appears in the nav side bar under the Drivers node.
The AdrNetwork installation is complete.

Install the ADR Device

The ADRDevice component is provided as part of the adr module and is available in the ADR palette or from the ADR Device Manager's **New** dialog box. You can add any number of ADRDevices under the network.

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

Note: *The ADR client is a "device-level" component. Each ADRDevice represents a client account configured on the target DRAS server.*

Step 1 In the WorkbenchAX nav tree, double-click on the ADRNetwork node.
The ADR Device Manager view displays.

Step 2 At the bottom of the ADR Device Manager view, click the **New** button.
The **New** dialog box displays.

Step 3 In the **New** Dialog box, edit the name of the device, if desired, set the Enabled property to **true** and click the **OK** button.

The ADRDevice entry appears in the ADR Device Manager view and the ADRDevice component appears in the nav side bar under the Drivers node.

The ADRDevice is installed. Default settings are sufficient for the device to be enabled.

Note: *When configuring the ADRDevice, User name and Password details entered on the ADRDevice property sheet view are mapped to a participant account on the target DRAS server.*

For more information about property settings, see ["About the ADR Device"](#) on page 3-3.

CHAPTER 3

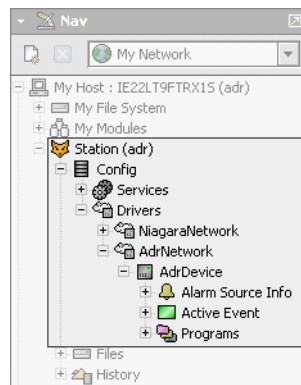
ADR Smart Client Driver Architecture

The ADR Smart Client uses the standard NiagaraAX network architecture. See “About Network Architecture” in the *NiagaraAX Drivers Guide* for more details.

The AdrNetwork forms the container for AdrDevices, the entities responsible for connecting to DRAS servers. Components are the station interface to DR events coming from DRAS servers. The AdrDevice forms a “client”, which resides under the AdrNetwork which in turn is located under the station’s “Drivers” container.

Hierarchically, the component architecture is: AdrNetwork, AdrDevice, DR Programs and DR Events.

Figure 3-1 Adr network hierarchy in the nav tree

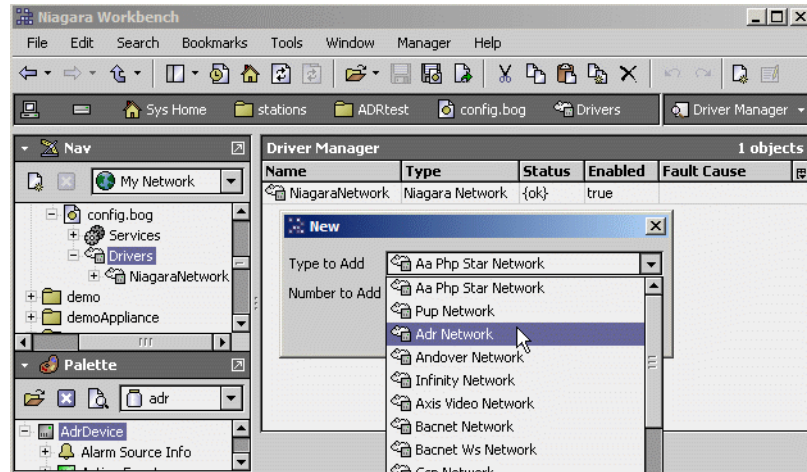


The following sections describe the major parts of the ADR architecture:

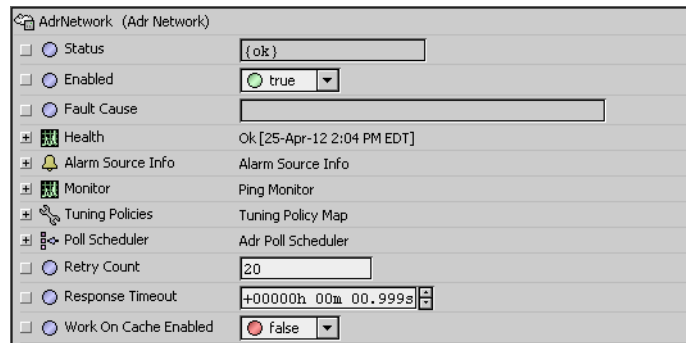
- [About the ADR Network](#)
- [About the ADR Device](#)
- [About DR Programs](#)
- [About DR Events](#)
- [About Log Setup](#)

About the ADR Network

The AdrNetwork is the top-level container component for AdrDevices in a NiagaraAX station. You can add the Adr Network from the “Driver Manager” view, using the **New** button, as shown in [Figure 3-2](#). You can also copy the AdrNetwork from the “adr” palette onto Drivers node in the nav tree.

Figure 3-2 Adding the ADR Network

The ADRNetwork 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 ADRNetwork property sheet view

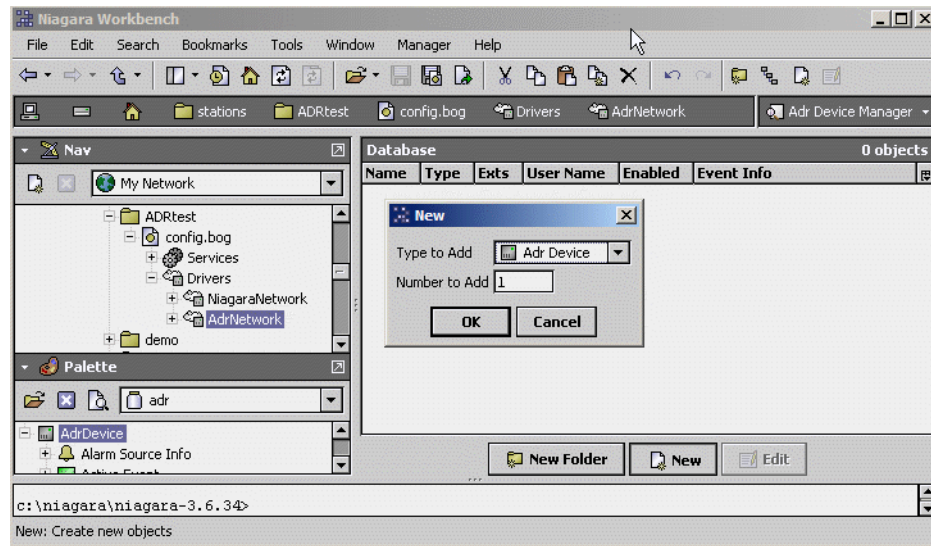
In addition to the common properties, note the following properties:

- Poll Scheduler**
 The ADR poll schedule defines different poll rates for “normal, slow, fast” buckets. This is different from the typical Niagara poll scheduler.
 - Fast Poll Rate: 20 seconds
 - Normal Poll Rate: 40 seconds
 - Slow Poll Rate: 1 minute
 The poll scheduler property is set to “true” by default. It can be completely disabled by setting the poll enabled flag to “false”.
- Retry Count**
 This property is not used.
- Response Timeout**
 This property is not used.
- 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. The property is set to “false” by default. 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.

About the ADR Device Manager view

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

Figure 3-4 Adr Network and Adr Device Manager view

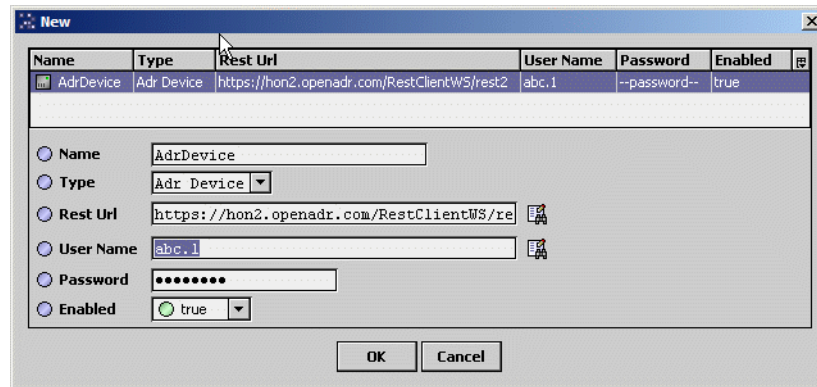


Note: Unlike many drivers, the device manager doesn't support device discovery. This is because each AdrDevice is modeled as a single client connection to the DRAS server and a discovery process is not necessary.

About the ADR Device

The ADR Device represents the connection to the target DRAS server through a unique participant account. Unlike most other drivers, the AdrDevice maps to a client configured in DRAS server. All devices under an AdrNetwork connect to the DRAS server with the REST URL configured in the device's property sheet, as shown in Figure 3-5.

Figure 3-5 New AdrDevice configuration



Note: Rest Url specifies the DRAS server connection to be used by this client.

As the "device-level" component in the ADR driver architecture, it is similar to most driver's device components— see "Common Adr network components" in the *NiagaraAX Drivers Guide* for general information.

The following ADR properties are available in the AdrDevice property sheet view:

- **Status**
Status of connection to the target DRAS server with the configured client account. Possible status flags include:
 - **ok**
Normal communications, no other status flags
 - **disabled**
Enabled property is set to false, either directly or in the ADR Network. While disabled, polling stops.
 - **down**

Error communicating to the target server. If status was previously ok (without changing account configuration), this may mean the target 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 ADR Network. See Status disabled description above.
- **Fault Cause**
This will be empty if the device is healthy. Otherwise, if status has fault, describes the cause.
- **Health**
Contains properties describing the health of the device.
 - **Down**
True if device is down, false otherwise.
 - **Alarm**
True if device is in alarm state, false otherwise
 - **Last Ok Time**
A display-only value indicating when the device was last healthy.
 - **Last Fail Time**
A display-only value indicating when the device last failed
 - **Last Fail Cause**
A string property describing the cause for the last failure.
- **Alarm Source Info**
Alarm source info under the ADR device has no significance in this driver.
- **Rest Url**
HTTPs URL of the target DRAS server.
***Note:** If the target server uses other than the standard HTTPs port number (port 443), then the port number must be included in the URL. For example, if the target server uses port 500, Rest Url should be: https:500//TargetServer.com/etc.)*
- **User Name**
User name for a unique client account configured in DRAS server.
- **Password**
Password for the client account configured in DRAS Server.
- **Poll Frequency**
Rate at which device sends a POLL request to the DRAS server to receive the event information. Configure the Normal, Slow, Fast poll frequencies in the ADR network's poll scheduler as required.
- **Active Event**
This value indicates the event in process. At any point in time, there is only one active event. The type of event information is determined by the DR program. Possible values are listed below:
 - **PRICE_ABSOLUTE**
Indicates price number, for example \$0.25.
 - **PRICE_RELATIVE**
Indicates change in price, for example \$-0.05.
 - **PRICE_MULTIPLE**
Indicates multiple of current price, for example 1.5
 - **LOAD_LEVEL**
Indicates amount of load based on an enumeration, for example moderate, high, etc.
 - **LOAD_AMOUNT**
Indicates fixed amount of load shed, for example 5 MW.
 - **LOAD_PERCENT**
Indicates percentage of load shed, for example 10%.
 - **LOAD_RELIABILITY**
A number signifying the reliability of the grid.

The event information shown above along with the operation Mode value [Op Mode Value] are modeled as control points under the active event. An event information point with the value "null" indicates that the active event does not carry information for that point.
- **Programs**
This value indicates DR programs associated with the client account configured on the DRAS server
- **Last Poll Time**
This display-only value indicates the time the last poll was made.
- **Event Info**
A comma separated string representing the event information (Load/Pricing) belonging to active event. If there are no active events under the device, this slot will be empty.

- **Acknowledgement URL**
Each DR event received by the client must be acknowledged. The “restConfirm” interface is used as the Acknowledgement Url although communication to the DRAS server is primarily through the “rest2” interface. The Acknowledgement Url is set to a default value that is in accordance with the Rest Url slot. If required, you can edit the Acknowledgement Url. If the event is correctly acknowledged, the “Event confirmation” slot under Adr Event will read “SUCCESS”.
- **Enable Cache**
A Boolean slot, this must be enabled if the event op mode schedules/event info instances are to be saved under the event component. When set to true, the schedule informatoin will be stored under the corresponding DR event component.
***Note:** This cached information is used to update the “Active Event” slot when the device is down provided the “Work On Cache Enabled” property in the AdrNetwork is set to true.*

About the Connection Status

If the crypto service is not present under the Services node (**station > config > services**), the AdrDevice status is {down} and Fault Cause is stated as “javax.baha.naming.UnresolvedException: Service not found: crypto:CryptoService”, as shown in Figure 3-6.

Figure 3-6 Service Not Found Error

The screenshot shows the AdrDevice configuration window. The Status is set to {down}. The Enabled checkbox is checked. The Health slot shows a failure message: "Fail [12-Oct-11 2:29 PM IST] javax.baha.naming.L...". The Last Fail Cause is "Service not found: crypto:CryptoService".

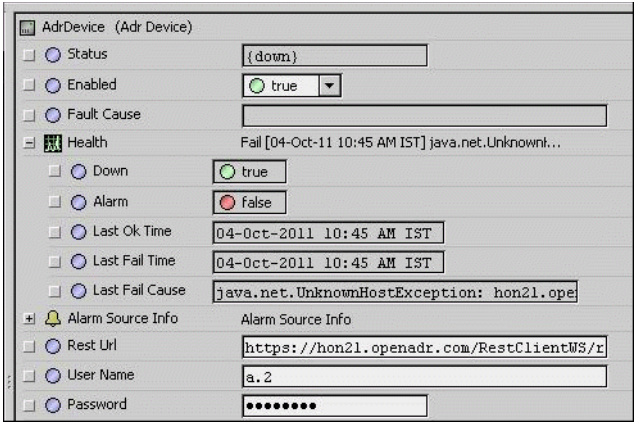
Configuring an AdrDevice with an invalid username/password results in an authentication error. The **Health** slot shows the details of the connection error. In this case, an invalid username/password generates “HTTPError code: 401 ;access denied” and the AdrDevice status is “{down, alarm}”, as shown in Figure 3-7.

Figure 3-7 Device Connection Error - Authentication Error

The screenshot shows the AdrDevice configuration window. The Status is set to {down, alarm, unacked}. The Enabled checkbox is checked. The Health slot shows a failure message: "Fail [04-Oct-11 10:26 AM IST] HttpError code:401...". The Last Fail Cause is "HttpError code:401 ;Access denied". The Rest Url is "https://hon2.openadr.com/RestClientWS/re", User Name is "a.2", and Password is masked with dots. The Poll Frequency is set to Normal.

An AdrDevice configured with an invalid Rest Url results in a connection failure, as shown in Figure 3-8 below. The AdrDevice status is {down} and the **Health** slot for Last Fail Cause shows the details of the connection error. In this case, “UnknownHostException” error.

Figure 3-8 Device Connection Error - Invalid Url



About DR Programs

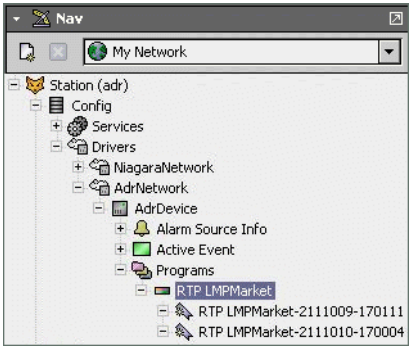
The DRAS server supports many types of DR programs. Two of the available DR programs are described here:

- **CPP**
Critical Peak Pricing (CPP) sends accurate information regarding the cost of energy. This helps you make informed decisions about how and when to use electricity. While the price of electricity is higher during periods of high energy use, called CPP events, the CPP rate offers lower prices during all other times. This gives you the opportunity to better assess and potentially reduce overall energy costs.
- **DBP**
Demand Bidding Program (DBP) pays you an incentive to reduce your electric load when notified of a demand response event.

Each DR program sends DR signals containing a particular type of information to the customer. For instance, the CPP program sends you the relative price of the electricity at the peak hour of the day whereas, DBP sends the expected load you are to shut down at a given time of the day.

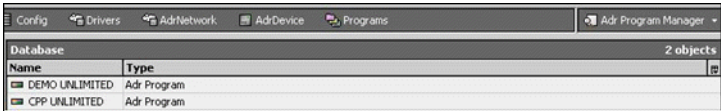
Once the ADR device attains {ok} state, it starts polling the DRAS server at the configured poll frequency. Program and event information received is then stored in the station under the device node. As shown in Figure 3-9, the DR program associated with a specific client account on the DRAS server is modeled under the Programs component of the ADR device.

Figure 3-9 DR Program associated with client account



Double clicking the Programs node opens the ADR Program Manager view, as shown in Figure 3-10.

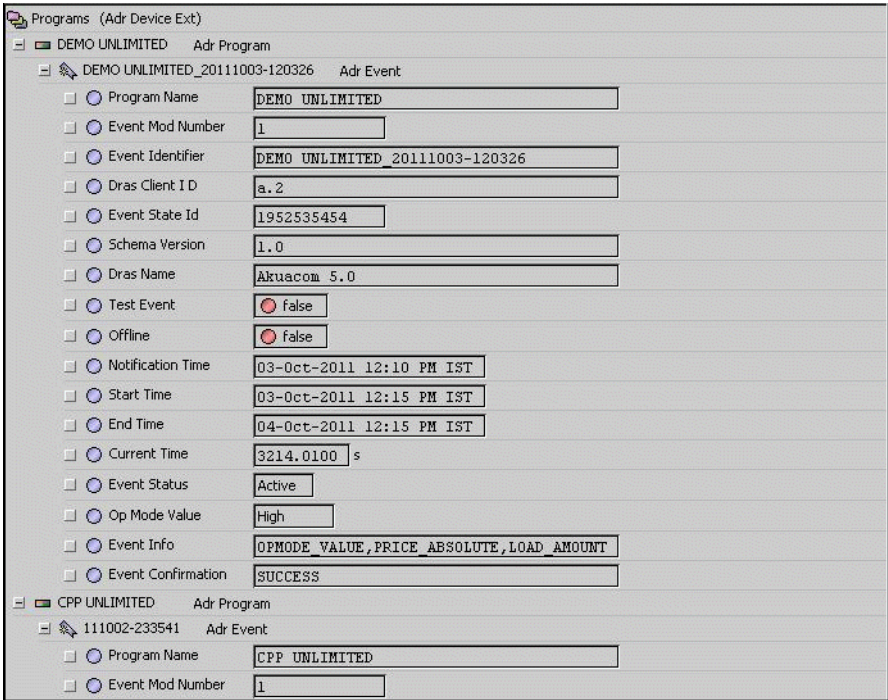
Figure 3-10 ADR Program Manager view



One or more DR events can be configured (on the DRAS server) for each program visible in the ADR Program Manager view.

The device property sheet view shown in [Figure 3-11](#) below shows the two programs that are associated with this client account, DEMO UNLIMITED and CPP UNLIMITED.

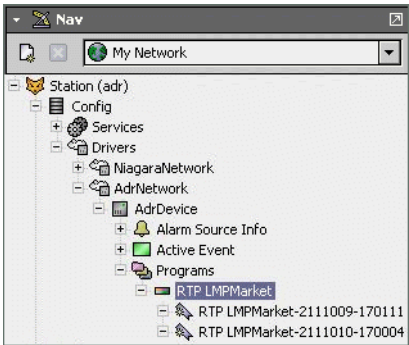
Figure 3-11 *AdrDevice property sheet view shows programs associated with client account*



About DR Events

There can be more than one event under a DR program, as shown in [Figure 3-12](#).

Figure 3-12 *Multiple events under DR program*



Double-clicking a DR program under the Programs node opens the Adr Event Manager view., shown in [Figure 3-13](#)

Figure 3-13 *Adr Event Manager view*

Database							2 objects	
Name	Type	Program Name	Start Time	End Time	Event Status	Event Info		
RTP LMPMarket-2111009-170111	Adr Event	RTP LMPMarket	10-Oct-11 12:30 PM IST	11-Oct-11 12:29 PM IST	Active	OPMODE_VALUE		
RTP LMPMarket-2111010-170004	Adr Event	RTP LMPMarket	11-Oct-11 12:30 PM IST	12-Oct-11 12:29 PM IST	Near	OPMODE_VALUE,PRICE_ABSOLUTE		

Each DR event carries information about event start time, end time and event info values. [Figure 3-14](#) below shows the property sheet view of a DR event.

Figure 3-14 Adr Event Property Sheet view

The screenshot shows the 'Property Sheet' for an event with ID '111002-233541'. The properties are listed in a table-like format with expandable/collapsible icons on the left. The values are as follows:

Program Name	CPP UNLIMITED
Event Mod Number	1
Event Identifier	111002-233541
Dras Client ID	a.2
Event State Id	1952539718
Schema Version	1.0
Dras Name	Akuacom 5.0
Test Event	<input type="radio"/> false
Offline	<input type="radio"/> false
Notification Time	03-Oct-2011 12:06 PM IST
Start Time	03-Oct-2011 05:00 PM IST
End Time	03-Oct-2011 10:30 PM IST
Current Time	-9941.1970 s
Event Status	Near
Op Mode Value	Normal
Event Info	OPMODE_VALUE, PRICE_RELATIVE
Event Confirmation	SUCCESS

The following Event properties are available in the AdrEvent property sheet view:

- Program Name
Identifier of the program that this DR event was issued for.
- 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.
- Event Identifier
Identifier for the DR event that was created when the DR event was first issued.
- Dras Client ID
Identifier of the DRAS client that this event is being sent to.
- Event State Id
This is a transaction ID that is guaranteed to be unique for each instance of the event that is generated and sent to the DRAS Client. It is used by the DRAS Client to confirm reception of the event from the DRAS.
- Schema Version
This is the version number for the event state XML schema sent from DRAS server.
- Dras Name
Optional field that is the name and/or version of the DRAS server that is sending the event 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.
Note: This is not tested.
- 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.
Note: This is not tested.
- Notification Time
Date and time that the DRAS Client should first be notified of the upcoming DR event.
- Start Time
Date and time for when the DR event begins.
- End time
Date time for when the DR event ends.
- Current Time
This is the relative time in seconds from the beginning of the DR event window that this particular instance of the event is valid for.
 - When this value is negative, it indicates that event has still not reached active state.
 - When this value reaches zero, it indicates the event start time.
 - When this value is positive, it indicates that the event is active.

Note: One exception is that Event Status might display as "active" even if the Current Time has a negative value when there is a manual configuration in the server. For additional details on "Current Time," see section *Manual vs. Auto-DR Mode*.

- Event Status
This is the DR event status that is valid at the time specified by the current Time slot. DR event can be in four different states:
 - None
There are no events scheduled in server.
 - Far
Event is notified, but not yet started.
 - Near
Event is notified, and it is nearing the active window.
 - Active
Event is active.
- Op Mode Value
This is the operation mode value that is valid at the time specified by the Current Time slot. This can take one of the following values: Normal, Moderate, High and Special
- Event Info
A comma separated string representing the event information that's available for this event.
- Event Confirmation
Represents whether event acknowledgement is sent to server successfully.
If success, this slot will have a value "SUCCESS", in case of failure, appropriate error string is displayed.

About Event Cache

When a DR event xml is sent from the DRAS server, in addition to the current state of the event, a schedule of event values are also sent, as shown in the example below:

- Event Start Time: 08:00 P.M
- Event End Time: 09:00 P.M

A list of various op mode value changes/ pricing and load shed changes over the event's active window can also be configured in the DRAS server. The DR event xml received from the DRAS server contains the complete information. The schedule of event information that is valid at different time offsets will help when there is a temporary network connection issue between DRAS client and server.

The ADR Device configuration determines whether or not schedule information is stored in the station bog file. This determination is made using the "Enable Cache" slot under the device. The default value for this slot is "false" (schedule is not cached). If a schedule needs to be stored, the Boolean slot must be set to "true".

The "Work On Cache Enabled" slot under ADRNetwork determines whether cached data will be used to update the "Active Event" component in the case of a network failure. This slot is defaulted to "false". When set to "true", a network level monitoring thread updates the "Active Event" component for devices that are currently "down" if the "Enable Cache" slot for those devices is set to "true".

If both the "Work On Cache Enabled" component under the ADRNetwork and the "Enable Cache" component under the ADRDevice are set to "true", then the "Active Event" component under the ADRDevice will be updated with cached event schedules as shown below. However, the status of the points is still marked down to indicate that the client connection to the DRAS server is unhealthy, as shown in [Figure 3-15](#).

Figure 3-15 Active Event Updated From Cached Data

Active Event		Adr Active Event
Price Absolute	0.04116	{down}
Price Relative	-	{down,null}
Price Multiple	-	{down,null}
Load Level	-	{down,null}
Load Amount	-	{down,null}
Load Percentage	-	{down,null}
Grid Reliability	-	{down,null}
Op Mode Value	Normal	{down}
Event Id	RTP LMPMarket-2111010-170004	
Event Start Time	11-Oct-11 12:30 PM IST	
Event End Time	12-Oct-11 12:29 PM IST	
Program Name	RTP LMPMarket	

About Active Event

At any point in time, there can be only one active event. This active event is displayed as a slot under Adr Device as shown in [Figure 3-16](#)

Note: *Akuacom DRAS Server would not restrict creating more than one active event for a specific timeslot. But Akuacom team confirms that -creating more than one active event for the same timeslot is a very rare use case scenario and if that happens, there is an internal priority maintained in the server that decides the high priority active event out of all configured events. This is reflected in DR event xml as the *topmost node*.*

Event Info can be of various types depending on the requirements of the DR program. This event info may take on any of the following values:

- PRICE_ABSOLUTE - Price number, i.e. \$0.25
- PRICE_RELATIVE - Change in price, i.e. -\$0.05
- PRICE_MULTIPLE - Multiple of current price, i.e. 1.5
- LOAD_LEVEL - Amount of load based on an enumeration, i.e. moderate, high, etc.
- LOAD_AMOUNT - Fixed amount of load to shed, i.e. 5 MW
- LOAD_PERCENTAGE - Percentage of load to shed, i.e. 10%
- GRID_RELIABILITY - Number signifying the reliability of the grid

The event information along with the “Op Mode Value” are modeled as control points under the Active Event.

[Figure 3-16](#) shows an active event with Event Id "DEMO UNLIMITED_20111003-120326" under the "DEMO UNLIMITED" DR program. Event Start Time and Event End Time - represent the active period of the event. This event provides two types of event information - Price Absolute and Load Amount with the OpMode Value, “high”. The rest of the event information slots have a "null" value, indicating that this particular DR event does not carry any information for those points.

Figure 3-16 Active Event

Active Event (Adr Active Event)	
Price Absolute	11.00000 {ok}
Price Relative	- {null}
Price Multiple	- {null}
Load Level	- {null}
Load Amount	11.00000 {ok}
Load Percentage	- {null}
Grid Reliability	- {null}
Op Mode Value	High {ok}
Event Id	DEMO UNLIMITED_20111003-120326
Event Start Time	03-Oct-11 12:15 PM IST
Event End Time	04-Oct-11 12:15 PM IST
Program Name	DEMO UNLIMITED

The event info value read from DRAS server is reflected in Adr Proxy Ext's read value as shown in [Figure 3-17](#) below.

Figure 3-17 DRAS Event Info as Niagara Points

The screenshot shows a configuration window for an 'Active Event (Adr Active Event)'. It contains several sections with expandable/collapsible icons. The 'Price' section includes 'Price Absolute' (16.00000 {ok}), 'Price Relative' (- {null}), and 'Price Multiple' (- {null}). The 'Load' section includes 'Load Level' (- {null}) and 'Load Amount' (16.00000 {ok}). The 'Facets' section shows 'precision=5'. The 'Proxy Ext' section includes 'Status' ({ok}), 'Fault Cause' (empty), 'Enabled' (true), 'Device Facets' (empty), 'Conversion' (Default), 'Tuning Policy Name' (Default Policy), 'Read Value' (16.00 {ok}), 'Write Value' (0.00 {ok}), and 'Out' (16.00000 {ok}).

About Manual vs. Auto-DR Mode

In Akuacom DRAS server a client account can be configured to operate under two modes, Manual and Auto-DR.

- Manual Mode without scheduled Auto-DR events
Even though currently there are no active events configured for this client, Op Mode Value is displaying "moderate", as shown in [Figure 3-18](#). Event Id and other fields are empty and that indicates active event has a manually configured value.

Figure 3-18 Active Event - Manually Configured Value

The screenshot shows a configuration window for an 'Active Event (Adr Event)' with the title 'DEMO UNLIMITED_20111011-145910'. It contains various fields: 'Program Name' (DEMO UNLIMITED), 'Event Mod Number' (1), 'Event Identifier' (DEMO UNLIMITED_20111011-145910), 'Dras Client ID' (8.2), 'Event State Id' (70564016), 'Schema Version' (1.0), 'Dras Name' (Akuacom 5.0), 'Test Event' (false), 'Offline' (false), 'Notification Time' (11-Oct-2011 03:05 PM IST), 'Start Time' (11-Oct-2011 03:10 PM IST), 'End Time' (11-Oct-2011 03:20 PM IST), 'Current Time' (-193.9790 s), 'Event Status' (Active), 'Op Mode Value' (Special), 'Event Info' (OPMODE_VALUE,PRICE_ABSOLUTE,LOAD_AMOUNT), and 'Event Confirmation' (SUCCESS).

- Manual Mode along with scheduled Auto-DR event
Manual mode can be set when there is an associated DR event for the same client. As you see in the below figure, since the manual mode is set, even though the event DEMO_UNLIMITED_20111011-145910 reaches active state only after 193 seconds [indicated by the current time field], Event Status is marked as "active" and Op Mode Value has a value "Special", as shown in [Figure 3-19](#).

Figure 3-19 Active Event Showing Manually configured Value

Active Event (Adr Active Event)	
Price Absolute	- {null}
Price Relative	- {null}
Price Multiple	- {null}
Load Level	- {null}
Load Amount	- {null}
Load Percentage	- {null}
Grid Reliability	- {null}
Op Mode Value	Special {ok}
Event Id	
Event Start Time	null
Event End Time	null
Program Name	

When the event reaches active state, price absolute and load amount displays the value coming from Auto-DR event. Op Mode Value should display "Normal" as per Auto-DR configuration. Instead, this displays "Special" due to manual configuration, as shown in [Figure 3-20](#).

Figure 3-20 Active Event Displaying Both Manual And Auto-DR value

Active Event (Adr Active Event)	
Price Absolute	1.00000 {ok}
Price Relative	- {null}
Price Multiple	- {null}
Load Level	- {null}
Load Amount	1.00000 {ok}
Load Percentage	- {null}
Grid Reliability	- {null}
Op Mode Value	Special {ok}
Event Id	DEMO UNLIMITED_20111011-145910
Event Start Time	11-Oct-11 3:10 PM IST
Event End Time	11-Oct-11 3:20 PM IST
Program Name	DEMO UNLIMITED

Note: Auto-DR/Manual configuration is done at the client level. This means for the same example stated above, client "a.2" operating under "manual" mode and client "a.1" operating under "Auto-DR" mode, "Active Event" component under Device "a.1" will always display Auto-DR value.

About Log Setup

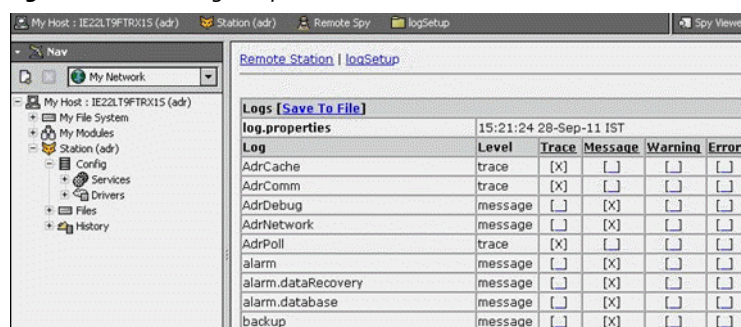
Niagara supports logging messages in the station console/platform application director to debug any functionality issues. To set this up, go to: **Station > Spy page > Log Setup page**.

Functionality based tracing is achieved using the classification of logs, as shown in [Figure 3-21](#).

- **AdrCache**
Captures messages for storing/monitoring event cache
- **AdrComm**
Captures messages related to communication to DRAS server
- **AdrDebug**
Captures messages related to adding/deleting program/event component
- **AdrNetwork**
Captures messages related to default network
- **AdrPoll**
Captures messages related to polling

Set these logs to the required level for debugging issues. By default, these will be set to "message" level.

Figure 3-21 ADR Log Setup



CHAPTER 4

ADR Smart Client Driver Components and Plugins Guide


The following sections provide summary descriptions of the components and component views (plugins) that are related to the ADR Smart Client Driver.

Components in the adr module

The following components are contained in the adr module:

- [AdrNetwork](#)
- [AdrDevice](#)
- [AdrPollScheduler](#)
- [AdrActiveEvent](#)
- [AdrDeviceExt](#)


adr-AdrNetwork

 AdrNetwork models a network of Adr objects. It is available in the adr palette. For more details see, [“About the ADR Network”](#) on page 3-1.

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

- [ADR Smart Client Driver Architecture.](#)


adr-AdrDevice

 AdrDevice represents the client connection to the DRAS server through a participant account. It is available in the AdrNetwork. For more details see, [“About the ADR Device”](#) on page 3-3.

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

- [ADR Smart Client Driver Architecture.](#)


adr-AdrPollScheduler

 The ADR poll schedule defines different poll rates for “normal, slow, fast” buckets. The poll rates are different from the typical Niagara poll scheduler. It is available in the AdrNetwork property sheet. For details about the Poll Scheduler, see the property sheet discussion in the [“About the ADR Network”](#) on page 3-1.

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

- [ADR Smart Client Driver Architecture.](#)


adr-AdrActiveEvent

 AdrActiveEvent models the event currently in process. It is available on the AdrDevice property sheet and in the adr palette under AdrDevice. For more details see, [“About Active Event”](#) on page 3-10.

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

- [ADR Smart Client Driver Architecture.](#)

adr-AdrDeviceExt

 AdrDeviceExt models programs associated with the client account. It is available in the AdrDevice property sheet under Programs. For more details see, [“About DR Programs”](#) on page 3-6.

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


- [ADR Smart Client Driver Architecture.](#)

Plugins in the adr module

The following plugins (views) are contained in the adr module:

- [Adr Device Manager](#)
- [Adr Program Manager](#)
- [Adr Event Manager](#)


adr-AdrDeviceManager

 The Adr Device Manager view is the default view of the AdrNetwork component. This view displays a table view of various connection details including status, health, alarm, monitoring, tuning, and poll scheduling, as well as any added AdrDevice, showing the device Name, user name, password, active event and DR programs associated with the client. For more details see, [“About the ADR Device Manager view”](#) on page 3-2.

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

- [ADR Smart Client Driver Architecture.](#)


adr-AdrProgramManager

 The Adr Program Manager view is the default view on Programs under the AdrDevice component. This view displays a table view of DR programs associated with the client account on the DRAS server. For more details see, [“About DR Programs”](#) on page 3-6.

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

- [ADR Smart Client Driver Architecture.](#)

adr-AdrEventManager

 The Adr Event Manager view is the default view on Events under a Program under the AdrDevice component. This view displays a table view of DR events under a Program. For more details see, [“About DR Events”](#) on page 3-7.

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

- [ADR Smart Client Driver Architecture.](#)