All Tech Notes, Tech Alerts and KBCD documents and software are provided "as is" without warranty of any kind. See the Terms of Use for more information.

Topic#: 002590 Created: October 2011

Introduction

This Tech Note provides detailed instructions in three parts:

- Consuming Alarms Using Application Server QuickScript
- Acknowledging Alarms Using Application Server QuickScript
- Testing Application Server QuickScript for Consuming and Acknowledging Alarms

This is done by first creating an Instance of \$UserDefined Template Object, and then creating 2 different scripts – one for consuming Alarms and second for Acknowledging Alarm. Each Script is further documented with code snippets. By following along this TechNote, you will have a \$UserDefined Instance created and deployed in the Galaxy. Further, at the end of it, you will test both the Scripts to Consume and Acknowledge Alarm. The end result will be Alarm Records logged in the Logger, depending on the Alarm Query.

This Tech Note provides the following benefits:

- It provides a way to consume Galaxy Alarms, so that Alarms can be presented in a custom format if desired.
- It serves as an example how to use Methods of a COM dll in Application Server Quickscript.

Application Versions

- Wonderware Application Server 3.1 and later
- Alarm Toolkit 8.0

Note: For this Tech Note, WAS 3.1 SP3 PO1 and MS Windows XP SP3 were used.

About the Alarm Toolkit

The Wonderware Alarm Toolkit can be used to create custom Alarm Provider and custom Alarm Consumer. A custom Alarm Provider and Alarm Consumer typically uses the COM wrappers like **wnwrapserver.dll** and **wnwrapconsumer.dll** to provide and consume Alarms. These COM wrappers (dlls) are installed by InTouch® or Application Server Installation in ...**Program Files\Common Files\ArchestrA** folder.

The current version of Alarm Toolkit is 8.0 (November 2011). The Alarm Toolkit installation CD is available within the **Toolkits** CD of Advanced Development Studio. Alarm Toolkit 8.0 is supported with InTouch 8.0 and higher and with Application Server 2.0 and higher. This is because the wrappers for Alarm Toolkit are installed by the product installations of InTouch and Application Server.

For more details on the Alarm Toolkit, refer to the AlarmToolKit User Guide on WDN.

Prerequisites

Before diving into specifics of each script, there are couple of setup tasks required to be able to use the Alarm Toolkit COM Wrapper. In this example, wnwrapconsumer.dll will be used since we are interested in consuming Alarms. Secondly, you need to create a \$UserDefined Object.

Import wnwrapConsumer.dll

- 1. Open the ArchestrA IDE.
- 2. Click Import/Script Function Library and import from C:\Program Files\Common Files\ArchestrA\wnwrapConsumer.dll.



FIGURE 1: GALAXY -> IMPORT -> SCRIPT FUNCTION LIBRARY

Look in		- O & P				
	PFServerPublicProxyStub.dll	wnwrapServerEx.exe				
	RedundantDeviceCustomPackage.dll	🖄 wnwrapServerExps.dll				
My Recent	SRTCommonProxyStub.dll	👏 WWCabFile.dll				
Documents	🔄 simulate.exe	🔊 wwclintf.dll	Swwclintf.dll			
	SLS_perf.dll	🔊 wwcommon.dll				
	slsping.exe	WWDebug.dll				
Desktop	sissyc.exe	S wwdirapi.dll				
	Suitelink perf.dll	www.dlo32a.dll				
	Supro di	Numbers dl				
ly Documents	wrisp.di	www.ieap.cli				
	whitime.dll	wwinsqics.dll				
	wnwrapConsumer.dl	ww155Lic.dll				
	wnwrapConsumer.exe	wwnetdde.exe				
	wnwrapConsumerps.dll	mwntshar.exe				
	🔊 wnwrapServer.dll	🖄 wwnttcpchk.dll				
My Network	File name: wnwrapConsumer.dll	•)pen			
Places			ancel			

FIGURE 2: FILE LOCATION

Archestr	A IDE	×
į)	Import Script Function Library: Import of wnwrapConsumer.dll succeed	ied

FIGURE 3: IMPORT SUCCEEDED

After importing:

• The wnwrapConsumer.dll is copied to following folder: C:\Program Files\ArchestrA\Framework\FileRepository\ \Vendors\ArchestrA\wnwrapConsumer.dll.

Create an Instance of \$UserDefined

1. Create an Instance of the **\$UserDefined Object**.

For this *Tech Note* it is called **UD_AImConsumer**. Host the UD_AImConsumer under **Area_001**.

2. Proceed to Consuming Alarms Using Application Server QuickScript.

Consuming Alarms Using Application Server QuickScript

Create UDAs

- 1. Open the UD_AImConsumer object instance in ArchestrA IDE and select the UDAs tab.
- 2. Create following UDAs that will be used in the script:
- Booleans: bConsumeAlms: Used to trigger the script for consuming and logging Alarm Records.

Create a Script

• Click the Scripts tab, then Add Script. Call the script AlmCons. This script is used to Consume Alarms and then Log those Alarm Records in Logger.

Verify that the Alarm Consumer Class Methods are Available for Use in the Script

• Use the Display Script Function Browser button at the far right of the window (Figure 4 below).



FIGURE 4: DISPLAY SCRIPT FUNCTION BROWSER BUTTON

	t Function Browser		Þ
ė.	WNWRAPCONSUMERLib.wwAlarmConsur	merClass	~
1.000	AlarmAck		
	S AlarmAck		
	AlarmAckByGUID		
	AlarmAckByGUID		
	AlarmAckByName		
	AlarmAckByName		
	- O Deselect		
	GetAlarm		
	SetAlarm		
	GetHighPriority		
	GetStatistics		
1	Geotodistics	3	×
			>

FIGURE 5: SCRIPT FUNCTION BROWSER

Creating the Scripts

Declare Script Variables

As per good programming practices, variable declarations are done in the Script editor's **Declarations** section.

```
Dim MyConsumer as WNWRAPCONSUMERLib.wwAlarmConsumerClass;
Dim currentXMLAlarms As Object;
Dim Result as Integer;
Dim almStr as String;
Dim xDoc as System.Xml.XmlDocument;
Dim node as System.Xml.XmlNodeList;
Dim leafnode as System.Xml.XmlNode;
```

FIGURE 6: DIMENSION DECLARATIONS

Script Example

```
Dim MyConsumer as WNWRAPCONSUMERLib.wwAlarmConsumerClass.
Dim currentXMLAlarms As Object;
Dim Result as Integer;
Dim almStr as String;
Dim xDoc as System.Xml.XmlDocument;
Dim node as System.Xml.XmlNodeList;
Dim leafnode as System.Xml.XmlNode;
```

WHERE

- MyConsumer in the Quickscript is used for Alarm Consumer Methods.
- currentXMLAlarms Alarm Records are returned in XML format.
- Result An integer to hold the result of Initialize and Register Alarm Consumer Methods.
- almStr Temporary storage for Alarm Records in XML format
- **xDoc**, **node** and **leafnode** variables are used to parse the XML Alarm records string to break it into a single alarm record and then further break into details within each alarm record like GUID, DATE, TIME, TAGNAME, TYPE, VALUE and STATE.

Create Your Startup Script

• Execution Type: Startup: Startup script is called when an object containing the script is loaded into memory, such as during deployment, platform, or engine start. Startup instantiates COM objects and .NET objects.

Depending on load and other factors, assignments to object attributes from the Startup method can fail.

• Attributes that reside off-object are not available to the Startup method.

```
MyConsumer = new WNWRAPCONSUMERLib.wwAlarmConsumerClass;
xDoc = new System.Xml.XmlDocument;
Result = MyConsumer.InitializeConsumer("ConsumerApplication");
LogMessage(StringFromIntg(Result, 10));
Result = MyConsumer.RegisterConsumer(0, "testConsumer",
 "ConsumerApplication", "1.1.1");
LogMessage(StringFromIntg(Result, 10));
LogMessage("---Instantiate, Initalize, Register AlarmConsumer,
SetXMLAlarmQuery---");
MyConsumer.SetXmlAlarmQuery("<QUERIES FROM_PRIORITY=""1""
TO_PRIORITY=""999"" ALARM_STATE=""Al1""
DISPLAY_MODE=""Summary"'><QUERY><NODE>localhost</NODE><PROVIDER>Galaxy</PR
OVIDER><GROUP>Area_001</GROUP></QUERIES>");
System.AppDomain.CurrentDomain.SetData("AlarmConsumerApp", MyConsumer);
```

FIGURE 7: STARTUP SCRIPT

Script Example

```
MyConsumer = new WNWRAPCONSUMERLib.wwAlarmConsumerClass;
xDoc = new System.Xml.XmlDocument;
Result = MyConsumer.InitializeConsumer("ConsumerApplication");
LogMessage(StringFromIntg(Result, 10));
Result = MyConsumer.RegisterConsumer(0, "testConsumer", "ConsumerApplication", "1.1.1");
LogMessage(StringFromIntg(Result, 10));
LogMessage("---Instantiate, Initalize, Register AlarmConsumer, SetXMLAlarmQuery---");
MyConsumer.SetXmlAlarmQuery("<QUERIES FROM_PRIORITY=""1" TO_PRIORITY=""999" ALARM_STATE=""All""
DISPLAY_MODE=""Summary" ><QUERY><NODE>localhost</NODE><PROVIDER>Galaxy</PROVIDER><GROUP>Area_001</GROUP></QUERY></QUERIES>");
System.AppDomain.CurrentDomain.SetData("AlarmConsumerApp", MyConsumer);
```

This script does the following:

- Creates an instance of the Alarm consumer class WNWRAPCONSUMERLib.wwAlarmConsumerClass and XmIDocument class.
- Calls methods of the Alarm Consumer class to Initialize and Register the consumer: InitializeConsumer and RegisterConsumer. The InitializeConsumer method ensures that Alarm Manager has been started and Alarm system has been initialized. The RegisterConsumer registers with the Distributed Alarm System, with a Product Name of testConsumer and Application Name ofConsumerApplication.
- Calls **SetXmlAlarmquery** method Alarm Consumer class to set the Alarm Query. In this case Alarm Query is set to consume all alarms from the Galaxy under the **Area_001** group.
- The System.AppDomain.CurrentDomain.SetData method is used to share the MyConsumer Object connection with another script within the same or another Application Server object instance. In this case the Myconsumer Object is shared with script AckAlarmByName within the same object instance: UD_AlmConsumer.

Note: For more details on Alarm Toolkit Class Methods, refer to the AlarmToolkit Users Guide on WDN.

Create Execution Script Trigger

Create an Execution Script Trigger with the following:

Execution Type: Execute: This script is configured to trigger when AppEngine performs a scan, the Object is OnScan and when the boolean UDA **bConsumeAlms** is set to **TRUE**.

icripts:			Execution type: Execute 🖌 🖌
Basics 👜 Expression:	me.bConsumeAlms		
Trigger type:	OnTrue		Quality changes
Trigger period:	00:00:00.0000000	6	Runs asynchronously
Deadband:	0.0	6	Timeout limit: 0 ms 🗗
Historize scri	ipt state	6	Report alarm on execution error
			Priority:

FIGURE 8: EXECUTION SCRIPT TRIGGER



FIGURE 9: SCRIPT

Script Example



me.bConsumeAlms = false;

- The boolean UDA bConsumeAlms is set to TRUE using the Object Viewer. This triggers the execution of the script. The GetXmlCurrentAlarms2 method from the Alarm Consumer class is used to get all the alarm records in accordance with the Alarm Query set in an earlier section.
- The alarms in XML form are then copied in a temporary string called **almStr**.

• The classes System.Xml.XmlDocument, System.Xml.XmlNodeList and System.Xml.XmlNode are then used to separate out each individual element from the Alarm Records as shown above in the code snippet.

After this script executes, the boolean UDA **bConsumeAIms** is set back to **FALSE** to ensure that Alarms can be requested on a demand basis and not continuously at every scan cycle. Also the LogMessage is used not only for logging alarm records but also used as debugging tool to monitor the progress of the script execution.

Create Execution Script Type: Shutdown

A Shutdown script is called when the object is about to removed from memory, usually as a result of the AppEngine stopping. Shutdown scripts are primarily used to destroy COM objects and .NET objects and to free memory.

```
LogMessage("---DeRegisterConsumer---");
MyConsumer.DeregisterConsumer();
```

FIGURE 10: SHUTDOWN SCRIPT

Script Example

```
LogMessage("---DeRegisterConsumer---");
MyConsumer.DeregisterConsumer();
```

Acknowledging Alarms Using Application Server QuickScript

Create UDAs

- 1. In the UD_AImConsumer object instance in ArchestrA IDE and select the UDAs tab.
- 2. Create following UDAs that will be used in the script:
- Booleans: **bAckAlarm** used to enable the script for Acknowledging Alarm
- String: strAlarmName Alarm Name to be acknowledged.

Create Script

Create a second script for acknowledging alarm by name, and call it AckAlarmByName.

Declare Script Variables

Declarations

```
Dim MyConsumer as WNWRAPCONSUMERLib.wwAlarmConsumerClass;
```

```
Dim Result as Integer;
```

FIGURE 11: DECLARATIONS

- Execution Type: Execute This script is configured to trigger when AppEngine performs a scan, the Object is OnScan and when the boolean UDA bAckAlarm is set to TRUE.
- Execute Script:



FIGURE 12: EXECUTE SCRIPT

Script Example

MyConsumer = System.AppDomain.CurrentDomain.GetData("AlarmConsumerApp");

Result = MyConsumer.AlarmAckByName(me.strAlarmName, "\\localhost\Galaxy", "Area_001", "ack from
ppk", "oprNameisPPK", System.Environment.MachineName, System.Environment.UserDomainName, System.Environment.UserName);
me.bAckAlarm = false;

The **System.AppDomain.CurrentDomain.GetData** method gets the value stored in the current application domain for MyConsumer. The **strAlarmName** UDA is of type string and can be configured for a default value. For example: **UD_AImConsumer.Analog_001.LoLo**. The string can be changed at runtime to acknowledge other alarms by name.

Testing Application Server QuickScript for Consuming and Acknowledging Alarms

Note: The InTouch Alarm Provider must be enabled to consume Alarms (Figure 13 below).

Seneral	Engine	Alarms	Platform History	Scheduler History	Engine History	Object Informatio
Netwo	ork addre:	ss:		PRERANAKN	B20	
Histor	y store fo	orward di	rectory:			6
Minim	um RAM:			1024	MB	6
Statis	tics avera	ige perio	d:	10000	ms	6 W
-InTo	uch alarm]Enable II) provider nTouch a	larm provider			6
	Regis	ter using	"Galaxy_ <galaxy< td=""><td>name>" instead of "</td><td>Galaxy"</td><td>6</td></galaxy<>	name>" instead of "	Galaxy"	6
	Alarm ar	eas (blar	ik for all):		1	6

FIGURE 13: INTOUCH ALARM PROVIDER CONFIGURATION

1. Deploy **UD_AlmConsumer**.

- 2. Set up a Field Attribute for generating Alarms within the Area_001 in the Galaxy. For example Generate Lo, Hi, HiHi alarms.
- 3. Start Object Viewer and set up a Watch Window with the following AttributeReferences (Figure 14 below):

🥔 Object Viewer											
<u>File Edit View Options Help</u>											
🛛 🍇 🌃 🏘 🛒 🛃 🗍 Attribute Reference:				ŀ	Go						
🖃 🚀 OPSManage2011	Attribute Name	Value	Ti	Quality	Status	SecurityC.	Category	Locked	Τ , 🔨		
🖃 😼 WinPlatform_001[PRERANAKNB20]	Tagname	UD_AlmConsumer		C0:Good	Ok	ReadOnly	System	UnLocked	St		
🖻 😓 AppEngine_001	ShortDesc	The UserDefine		C0:Good	Ok	ReadOnly	Writea	UnLocked	In		
🖮 🎒 Area_001 [Area_001]	ScanStateCmd	true		C0:Good	Ok	Operate	Writea	UnLocked	Bc		
UD AlmConsumer [UD AlmCons	ScanState	true		C0:Good	Ok	ReadOnly	Calcula	UnLocked	Bc		
	SecurityGroup	Default		C0:Good	Ok	ReadOnly	Writea	UnLocked	St		
	Area	Area_001		C0:Good	Ok	ReadOnly	System	UnLocked	Rŧ		
	Container			C0:Good	Ok	ReadOnly	System	UnLocked	Rŧ		
	Host	Area_001		C0:Good	Ok	ReadOnly	System	UnLocked	Re		
	AlarmMode	Enable		CU:Good	OK	ReadOnly	Calcula	UnLocked	ά		
	AlarmModeCmd	Enable		CU:Good	OK	Operate	Writea	UnLocked	a		
	Alarminnibit	false		CU:Good	OK	Operate	Writea	UnLocked	BC		
	InAlarm	raise		CO:Good	OK	ReadOnly	Calcula	UnLocked	BC		
	Configuersion	40		CO:Good	OK	ReadOnly	writea	UnLocked			
	<								>		
AttributeReference Value		Timesta	mp			Quality	Status				
UD_AlmConsumer.bConsumeAlms false		10/20/2011 4:4			4:47:04.223 PM		Ok				
UD_AlmConsumer.bAckAlarm false		10/20/2	10/20/2011 4:47:04.223 PM			C0:Good	Ok				
UD_AlmConsumer.strAlarmName UD_AlmConsu	mer.Analog_001.LoLo	0 10/20/2	10/20/2011 4:47:04.223 PM		C0:Good	Ok					
Watch List 1											
Ready	F	ILE: C:\Documents	and Sett	tings\preranal	ku\My Doc	uments\OPSM/	ANAGE 2011\w	atchWindow	s\Wa //		

FIGURE 14: OBJECT VIEWER FOR UD_ALMCONSUMER OBJECT

- 4. Set **bConsumeAlms** to true. The **AlmCons** script will execute.
- 5. Look at the Log Viewer to see the logged Alarm Records (Figure 15 below).

🖳 logv	iewer - [Log	g Viewer\Def	ault Group\	_ocal]			
File A	ction ⊻iew	Help					
4 4		a 🔊 🖬	🥩 🍲 🗛	8 2 6	ь <i>Б</i>		
			*1 1 1 0 * 0	Ma / e	¥ 🗐	Ly,	
🛛 🔽 Star	t Time: 10/20/2	2011 4:26:47	PM J	End Time	e: 12/31	/2100 11:59:59	9 PM
No:	Date	Time	Process ID	Thread	Log	Component	Message
10617	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons:GetAlarmsGetAlarms
10618	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: xml version="1.0"? <alarm_records count="4"><4</alarm_records>
10619	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons:
10620	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: Alarm Record
10621	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 58F04EC106D34010A1666273DC41BD98
10622	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 58F04EC106D34010A1666273DC41BD98
10623	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 2011/10/20
10624	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 23:48:2.506
10625	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UD_AlmConsumer.Analog_001.LoLo
10626	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: LoLo
10627	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 9
10628	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UNACK_ALM
10629	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: Alarm Record
10630	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 617707ACE5C140DCAC474D9C0FF932DC
10631	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 617707ACE5C140DCAC474D9C0FF932DC
10632	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 2011/10/20
10633	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 23:48:2.506
10634	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UD_AlmConsumer.Analog_001.HiHi
10635	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: HiHi
10636	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 9
10637	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UNACK_RTN
10638	10/20/2011	4:49:56 PM	3372	4668	Info	ScriptRuntime	UD AlmConsumer.AlmCons: Alarm Record
<u> </u>							
Connecte	đ						Filtered

FIGURE 15: LOG VIEWER ALARM RECORDS

- 6. Set **bAckAlarm** to true. The **AckAlarmByName** script executes.
- 7. To monitor that the falarm for UD_AImConsumer.Analog_001.LoLo was acknowledged, repeat steps 4 and 5.

Notice in the logger that Alarm record for UD_AlmConsumer.Analog_001.LoLo alarm has status of ACK_ALM.

🖳 logvi	iewer - [Log	Viewer\Def	ault Group\	Local]			
File A	ction <u>V</u> iew	Help					
			🤝 🗇 🗛		h /7		
		2 📑 🖿			2	LQ.	
🛛 🔽 Star	t Time: 10/20/2	2011 4:26:47	PM]]	End Time	: 12/31	/2100 11:59:59	9 PM
No:	Date	Time	Process ID	Thread	Log	Component	Message 🔺
10657	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons:GetAlarmsGetAlarms
10658	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: xml version="1.0"? <alarm_records count="4"><4</alarm_records>
10659	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons:
10660	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: Alarm Record
10661	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 58F04EC106D34010A1666273DC41BD98
10662	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 58F04EC106D34010A1666273DC41BD98
10663	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 2011/10/20
10664	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 23:55:11.845
10665	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UD_AlmConsumer.Analog_001.LoLo
10666	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: LoLo
10667	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 9
10668	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: ACK_ALM
10669	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: Alarm Record
10670	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 617707ACE5C140DCAC474D9C0FF932DC
10671	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 617707ACE5C140DCAC474D9C0FF932DC
10672	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 2011/10/20
10673	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 23:48:2.506
10674	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UD_AlmConsumer.Analog_001.HiHi
10675	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: HiHi
10676	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: 9
10677	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD_AlmConsumer.AlmCons: UNACK_RTN
10678	10/20/2011	4:57:16 PM	3372	4668	Info	ScriptRuntime	UD AlmConsumer.AlmCons: Alarm Record
<u> </u>							• • • • • • • • • • • • • • • • • • •
Connecte	ł						Filtered

FIGURE 16: ACK_ALM STATUS

P. Kulkarni

Tech Notes are published occasionally by Wonderware Technical Support. Publisher: Invensys Systems, Inc., 26561 Rancho Parkway South, Lake Forest, CA 92630. There is also technical information on our software products at Wonderware Technical Support.

For technical support questions, send an e-mail to wwsupport@invensys.com.

Back to top

©2011 Invensys Systems, Inc. All rights reserved. No part of the material protected by this copyright may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording, broadcasting, or by anyinformation storage and retrieval system, without permission in writing from Invensys Systems, Inc. Terms of Use.