OPC Alarms and Events Overview

This document provides an overview of the OPC Alarms and Events specification..

Table of Contents

Introduction

The OPC Foundation developed the OPC Data Access specification in 1996 to address the need for a common communication interface for transferring real-time process control data. The OPC Data Access specification defines a standard interface that allows applications to access real-time process control data from a variety of devices. The applications must implement one OPC Data Access compliant driver to access data from any OPC Data Access compliant server. The OPC Data Access specification has been widely adopted and is the standard communication mechanism for transferring real-time process control data.

The OPC Alarms and Events specification follows the same ideas set forth in the original OPC Data Access specification, only the Alarms and Events specification addresses real-time alarm and event data as opposed to real-time process control data. Released in 1999, the OPC Alarms and Events specification was developed to address a common communication interface for transferring real-time alarm and event data in the process control environment.

History of OPC Technology

OPC is based on the OLE (Object Linking and Embedding)/COM (Component Object Model) standard from Microsoft. OLE/COM was designed by Microsoft to be extensible by others. This allowed OPC to be developed on top of an existing technology, rather than inventing a completely new technology. It also provided a large number of clients for the OPC server data in the applications that are already OLE aware. Because the technology behind OPC is based on Microsoft OLE/COM, OPC must operate in a Microsoft-based environment.

OPC provides for a high degree of interoperability between client and server applications supplied by different suppliers. In the past, client application suppliers had to develop a different driver to interface with each control device. The OPC standard provides the client application suppliers the benefit of now only having to develop one driver to access data from a process control device. These situations are both shown in Figure 1.

If the control device supplier modified the interface to the device, the client supplier would also have to modify the client’s driver. OPC isolates the client software from the details of the various systems below the OPC interface. The device supplier is able to modify the functionality under the OPC server interface without affecting the client software. Client suppliers can now expend resources on true value added activities for their products in place of the effort required to maintain a library of device drivers.

Using the OPC specification, end users can chose the client application that best meets their needs. In the past, a user had to use specific client software that provided an interface for a particular control device. With OPC, any OPC compliant client application can interface to a control device with an OPC compliant server. In this way, the user gets the best solution for a particular task.

Another benefit comes from lower integration costs and risks. With plug and play OPC compliant components, available from a variety of suppliers, the system integrator can spend more time on the final integration goal and less time developing custom drivers. Since the solution is based on standard OPC components rather than custom drivers, the project risk is lower.

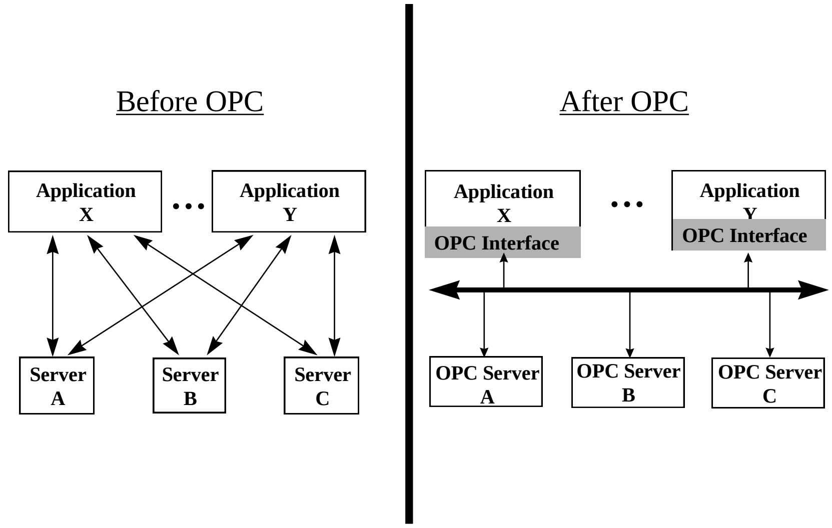

Figure 1: Before and after OPC

Architecture

As mentioned before, OPC is based on the client/server architecture. OPC servers are the interface to the control devices that supply information. An OPC server has two interfaces; a supplier specific interface to the supplier's process data and a common OPC interface which exposes the supplier specific data in an open format based on the OPC specification. OPC clients interface with the servers to read data from and write data to the control devices.

OPC clients can reside on the same PC as the OPC server (local) or they can reside on a separate PC (remote). In the local client application, COM is used as the communication mechanism between client and server. In the remote client application, Distributed COM (DCOM) is used as the communication mechanism between client and server. DCOM distributes the functionality of COM across PCs.

Alarms and Events

The OPC Alarms and Events specification is intended for use with process automation systems that generate alarms and events. An OPC Alarms and Events server is written to expose the alarms and events present in the system. The OPC Alarms and Events server does not create the alarms and events, it only reports the alarms and events previously defined in the system using a common communication interface. Once defined, the alarms and events in the system are automatically generated, based on the operating conditions and actions performed in the process plant. The level in a process vessel may reach a high state and generate a process alarm or an operator may make a set point change on a level control loop and generate an operator event. The OPC Alarms and Events server captures these alarms and events automatically, and makes them available to any client application that is interested in this information.

A client application that is interested in an alarm subscription to the server must first link to the server. A local client makes this connection through COM; a remote client makes this connection through DCOM. When the client requests a link to the server, the server loads into memory and starts executing on the PC. Once the link to the server is established, the client can establish one or more subscriptions to that server. In doing so, the client is telling the server to monitor the alarms and events in the system and report these alarms and events back to the client via a callback. The client can create more than one subscription to the server if needed. The client can also set filter criteria for each subscription. The OPC Alarms and Events specification defines a set of common filter criteria that can be applied. The OPC Alarms and Events specification also defines that this filtering be done by the server. By insisting that the server filter the alarms and events, traffic over the DCOM network link is decreased, which in turn increases the speed and efficiency of the link.

Definitions

In the process control industry, the terms alarms and events are used to describe occurrences in a process plant which have a certain meaning. In informal conversation, the terms alarm and event are often used interchangeably and their meanings are not distinct. However, the OPC Alarms and Events specification identifies these terms with unique meanings.

An alarm is an abnormal condition that requires special attention. Conditions may be single state, or multi-state. A multi-state condition is one whose state encompasses multiple ranges or sub-states which are of interest. For example, a level alarm condition may have multiple sub-states or sub-conditions including high alarm and high-high alarm. A single state condition has only one sub-state of interest, and thus has only one sub-condition associated with it. An example of a single state condition is a hardware failure condition, where a hardware device is either in the failed condition or not.

An event may or may not be associated with a condition. For example, the transitions into the level alarm condition and the return to normal are events which are associated with conditions. However, operator actions, system configuration changes, and system errors are examples of events which are not related to specific conditions.

Event Types

When a client application subscribes to a server for a specific set of events, the server provides an event notification to the client. There are three types of events: simple, tracking, and condition.

The three types of events as defined by the OPC Alarms and Events specification are shown in Figure 2. The condition-related events and tracking events share the same attributes as the simple events, plus they have their own unique attributes as shown.

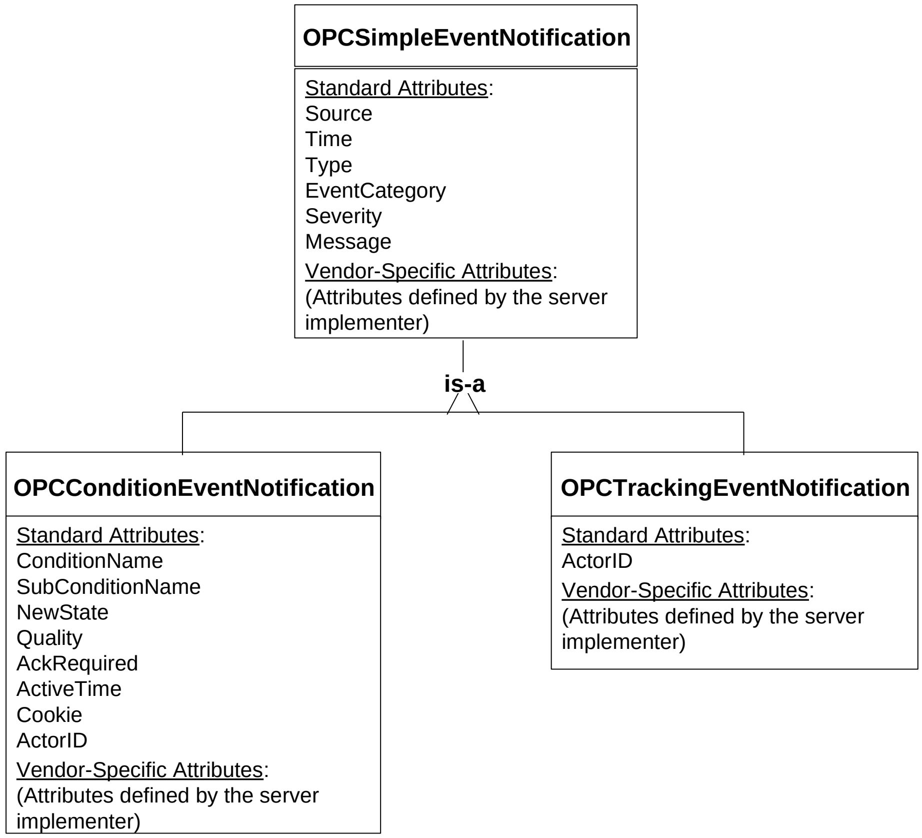

Figure 2: OPC Event Notifications

Simple Events

Simple events are typically informational messages that do not require any particular action be taken. Examples of simple events are systems messages such as startup or shutdown, operator logon or logoff; batch status messages such as startup or completion; and device failure messages.

Simple events have the following standard attributes.

Source	A reference to the object which generated the event notification. For example, this would be a device tag name (e.g. FIC101) if the event pertains to a tag entering the level alarm condition (condition-related event). It could also be a tag name for a tracking event such as the operator changing the set point value for FIC101. For a simple event such as a system error, the Source value might be “system”.
Time	The time that the event occurred.
Type	The type of the event, i.e. condition-related, tracking-related, or simple.
Event Category	The category to which this event belongs. Event Categories define groupings of events supported by an OPC Event server. Examples of event categories might include “Process Events”, “System Events”, or “Batch Events”. Event subscriptions may be filtered based on event category.
Severity	The urgency of the event. This may be a value in the range of 1 – 1000, with 1 being the lowest severity and 1000 being the highest. Typically, a severity of 1 would indicate in event which is informational in nature, while a value of 1000 would indicate an event of catastrophic nature which could potentially result in severe financial loss or loss of life. It is expected that few server implementations will support 1000 distinct severity levels. Therefore, server developers are

responsible for distributing their severity levels across the 1 – 1000 range in such a manner that clients can assume a linear distribution. For example, a client wishing to present five severity levels to a user could implement the severity level as shown in Figure 3.

Message Message text which describes the event. For condition-related events, this will generally include the description property of the active sub-condition.

Client Severity	OPC Severity
HIGH	801 – 1000
MEDIUM HIGH	601 – 800
MEDIUM	401 – 600
MEDIUM LOW	201 – 400
LOW	1 – 200

Figure 3: OPC Event Severity Level Example

Tracking Events

Tracking events are similar to simple events. However, tracking events typically indicate that some specific action was performed on the source by some specific actor. Therefore, in addition to information provided with simple events, tracking events also contain an 'actor ID' to indicated who or what performed this action. The actor ID is in the form of a supplier specific string. Like simple events, tracking events are often informational and require no specific action. Examples of tracking events are operator process actions such as changes to control loop set points or tuning parameters; system configuration changes, batch related information, access violations, etc.

Tracking events have the same standard attributes as simple events, in addition to the following.

Actor ID The identifier of the OPC Client which initiated the action resulting in the tracking-related event. For example, if the tracking-related event is a change in the set point of FIC101, the Actor ID might be a reference to the client application which initiated the change or might be the User ID of the operator who specified the change. This value is server specific, and its definition is the responsibility of the supplier implementing the server.

Condition Events

Condition events are associated with the detection of some condition in the supplier system that generally requires some sort of response or acknowledgement by the user or operator. Therefore, these conditions have some 'state' information associated with them. Additional condition events are sent any time there is a change in the state information associated with the condition. Examples of condition events are all categories of alarms, including single state alarms such as a deviation alarm and multi-state alarms such as a level alarm with high and low states.

Condition events have the same standard attributes as simple events, in addition to the following.

Condition Name The name of the associated OPC Condition.

Sub-Condition Name The name of the currently active OPC Sub-Condition.

Change Mask Indicates to the client which properties of the condition have changed.

New State Indicates the new state of the condition.

Condition Quality Indicates the quality of the underlying data items upon which this condition is based.

Ack Required	An indicator as to whether or not an acknowledgement is required.
Active Time	The time of the transition into the condition or sub-condition which is associated with this event notification.
Cookie	Server defined cookie associated with the event notification. This value is used by the client when acknowledging the condition. This value is opaque to the client.
Actor ID	The identifier of the OPC Client which acknowledged the condition, which is maintained as the Acknowledger ID property of the condition. This is included in event notifications generated by condition acknowledgments.

Areas

The expectation is that events and conditions available in the server are organized within one or more process areas. An area is a grouping of plant equipment configured by the user, typically according to areas of operator responsibility. According to the OPC Alarms and Events specification, implementation of the area concept is optional, but if areas are available, an OPC client can filter event subscriptions by specifying the process areas to limit the event notifications sent by the server.

Filtering

The OPC Alarms and Events client application can set filter conditions for each event subscription. Events may be selected using the following criteria:

- Type of event, i.e. simple, condition, or tracking – this filter allows the client to choose what classes of events to deliver on this subscription.
- Event categories – this filter allows the client to choose what combination of server specific event categories to deliver on this subscription. The client can ask the server for a list of available categories.
- Lowest severity – this filter allows the client to subscribe to all events with a severity greater than or equal to the specified severity.
- Highest severity – this filter allows the client to subscribe to all events with a severity less than or equal to the specified severity.
- Process areas – this filter allows the client to choose what combination of server specific area to deliver on this subscription. The client can ask the server for a list of available areas.
- Event sources – this filter allows the client to specify a set of event sources to deliver on this subscription. Wild cards can be supported.

A list of values for a single criterion are logically ORed together (i.e. if two event categories are specified, event notifications for both categories will be received). If multiple criteria are specified, they will be logically ANDed together, i.e. only those events satisfying all criteria will be selected. An example is specifying both lowest priority and highest priority will result in the selection of events with priorities lying between the two values.

For example, a client application sets the following filter criteria for each event subscription on an OPC Event server.

Type = CONDITION

Category = PROCESS

Low Severity = 600

Area = AREA1, AREA2

The specified criteria would result in the selection of condition-related events within the Process category in both AREA1 and AREA2 which are of high urgency (greater than or equal to 600).

Frequently Asked Questions

What is OPC?

OPC (originally OLE for Process Control; OLE has since been restructured and renamed ActiveX) is an industry standard created with the collaboration of a number of leading worldwide automation software and hardware suppliers working in cooperation with Microsoft. The OPC standard defines methods for exchanging real-time automation data among PC-based clients using Microsoft operating systems. The organization that manages this standard is the OPC Foundation.

What is the OPC Foundation?

The OPC Foundation is made up of over 230 members and 450 products from around the world (as of March 2000), including nearly all of the world's major providers of process control systems and instrumentation. For a complete list of OPC Foundation Members, go to the OPC Foundation web site at www.opcfoundation.org.

How will OPC benefit the user/supplier community in the process control and manufacturing world?

For software application users, OPC has made "plug and play" software a reality. OPC allows different applications written in different languages running on different Windows platforms to integrate.

Hardware users will also benefit. Process device hardware suppliers will develop OPC compliant interfaces, much like PC printer suppliers provide printer drivers that are able to work with multiple Windows applications. Users will be able to choose best-in-class products for a given application and easily integrate information from these devices with software applications across the enterprise.

Suppliers will benefit by reducing development costs associated with developing multiple drivers to automating systems. Instead, suppliers will be able to focus on the value-added product functionality.

What is the difference between OPC Data Access and OPC Alarms and Events?

The OPC Data Access specification addresses the need for exchanging real-time process control data using a common communication standard. Real-time process control data includes the analog and discrete information that comes from field devices, systems, and operators. The OPC Alarms and Events specification addresses the need for exchanging real-time alarm and event data using a common communication standard. Real-time alarm and event data includes the upset conditions that arise out of field devices and systems and the actions that systems and operators make in the course of controlling the process plant.

Is the OPC Foundation working on other specifications?

In addition to the Data Access specification, released in 1996, and the Alarms and Events specification, released in 1999, the OPC Foundation has developed a Historical Data Access specification (released in 1998) and a Batch specification (released in 2000).

How do I write an OPC server or an OPC client application?

The first place to start is with the OPC specification. Each OPC specification provides details on creating the COM objects necessary for implementing an OPC server. For example, there are three COM objects associated with the implementation of an OPC Alarms and Events server: OPCEventServer, OPCEventSubscription, and OPCEventAreaBrowser. For creating the client application, there are two separate OPC specifications associated with each server that define the interface to the server. One is the custom interface that defines the requirements for a C++ client and the other an automation interface that defines the requirements for a Visual Basic client.

How "fast" is the OPC standard (i.e. what "speed" of data transfer does it provide)?

The answer is, IT DEPENDS! OPC is a protocol standard, not a specific implementation scheme. OPC allows a wide variety of implementation schemes including "in-process" / "out-of-process" servers and "remote" / "local" servers. The type of server implemented, as well as the hardware platform used, network environment, etc. can ALL impact the "speed" or performance of the resulting implementation. It is technically possible to implement OPC with a wide range of speed and throughput capacities.

Will OPC run on ANY operating system or computer?

No, OPC is based on Microsoft's OLE (now ActiveX) environment. Consequently, OPC requires the OLE COM technology to be present in the operating system. The OLE COM technology is currently available on Microsoft's Windows 95, Windows NT, and Windows 2000 operating systems. OLE COM technology is now available on the UNIX operating systems.

Can any other companies join the OPC efforts?

The OPC Foundation is a not-for-profit corporation chartered to extend and evolve the OPC Specification. Suppliers and end-users are encouraged to join. Find out more information on OPC membership on the OPC Foundation web site at www.opcfoundation.org.

What is Microsoft's involvement in the OPC effort?

Microsoft was instrumental in forming the OPC Task Force. Microsoft technical resources worked closely with the OPC Task Force on the OPC standard. Microsoft will also act as an on-going advisor to the OPC Foundation.

Who owns the standard?

The standard is in the public domain and available to anyone who wishes to use it. A copy of the standard is posted on the OPC Foundation web site so that it is readily accessible to the user/supplier community. Download a copy of the OPC specifications from the OPC Foundation web site at www.opcfoundation.org.

References

OPC Alarms and Events Custom Interface Standard, Version 1.02, OPC Foundation, November 2, 1999

OPC Data Access Custom Interface Standard, Version 2.03, OPC Foundation, July 27, 1999

OLE For Process Control (OPC) Overview White Paper, Emerson Process Management Systems Division, February, 2000

To locate a sales office near you, visit our website at:
www.EmersonProcess.com/DeltaV
Or call us at:
Asia Pacific: 65.6777.8211
Europe, Middle East: 41.41.768.6111
North America, Latin America: +1 800.833.8314 or +1 512.832.3774

For large power, water, and wastewater applications contact Power and Water Solutions at:
www.EmersonProcess-powerwater.com
Or call us at:
Asia Pacific: 65.6777.8211
Europe, Middle East, Africa: 48.22.630.2443
North America, Latin America: +1 412.963.4000