3 Tips to Distribute With COM+ Events

Use these workarounds to make distributed deployments possible.

by Juval Lowy

WHAT YOU NEED

Windows 2000

The COM+ Event service is an exciting service that has evolved to address some classic problems of notifying and receiving events. Also known as Loosely Coupled Events (LCE), COM+ Events provide an effective way to decouple components by putting the logic for publishing and subscribing for events outside your components' scope. COM+ decouples the event publisher from its subscribers by introducing a middleman called the event class. You provide the sink interfaces signature, and COM+ provides the event class implementation and the publication mechanism itself.

As long as you install the publisher, the event class, and the subscribers all on the same machine, you can have any topology of interaction. On the same machine, publishers can publish to any event class, event classes can deliver events to any subscriber, and subscribers can subscribe to as many event classes as they like. Unfortunately, the COM+ Events service has a limitation—the event class and its subscribers must all be on the same machine. This limitation means distributed deployments aren't possible.

In this article you'll learn three workaround solutions that let you distribute your events across the network. All the solutions adhere to the limitation that the event class and the subscribers must reside on the same machine, and they solve the problem by designing around it. Like most things in life, each solution has pros and cons, and it's ultimately up to you, the system designer, to select the most appropriate solution for your domain problem.

This article assumes you're familiar with the concepts of COM+ Events, such as event classes, publishing events, persistent subscribers, and transient subscribers; if you're unfamiliar with these terms, see Resources for articles that cover COM+ Events basics, which you should read first to take full advantage of this article.

All three solutions for distributing COM+ Events require you to create a proxy application export of the

EXPORT A PROXY APPLICATION

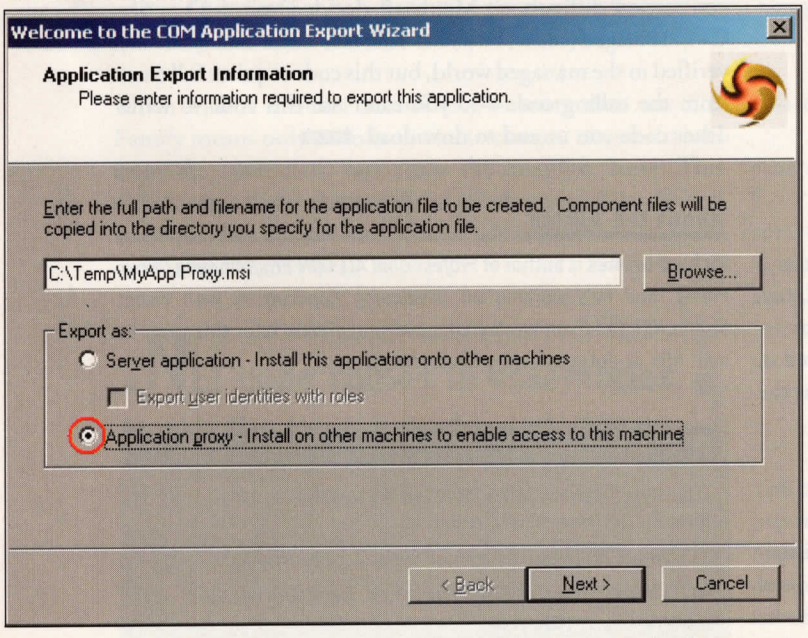

Figure 1 | To distribute your event classes, you must generate an application proxy to the server application where the event class resides. To do this, select the Application proxy option in the Application Export Wizard.

COM+ server application that contains the event class. To do this, right-click on the application icon in the COM+ Component Services Explorer and select Export from the pop-up context menu. This should bring up the Application Export Wizard. Click on Next to go to the second wizard screen, where you enter a name and location for the application export file to be created (see Figure 1). Next, export the application as an Application proxy.

An Application proxy export includes the type information, as well as the proxy and stub DLLs if required—but not the event classes or any other component in the application. You use a proxy installation when you want to enable remote access from a client machine to the machine where the application actually resides. A proxy export is only available for a COM+ server application, not for a library application. Once installed, you can configure the proxy application to access any remote machine on the network where the server application is installed, not just the machine that generated the proxy export. You specify the "real" application location on the proxy application properties page, which is located on the Activation tab in the Remote server name edit box.

As long as you install the publisher, the event class, and the subscribers on the same machine, you can have any topology of interaction.

1. One Machine for All Subscribers and Event Classes

This first solution is the simplest to implement. You install every event class on one machine, along with all subscribers. You install the event classes in a COM+ server application and generate a proxy installation for the event classes' application. Then you deploy the event class proxy application on all machines that host publishers, ensuring the proxy applications point to the event classes and subscriber's machine.

When a publisher on a remote machine A wants to fire an event of type E_1, the publisher creates a proxy for that event class and calls the event method on the proxy. COM+ marshals the event call to where the event class resides—on the subscriber's machine—and publishes COM+ to all the subscribers that subscribed to it (see Figure 2). Subscribers can subscribe easily to more than one event class because the event classes are installed locally on the subscriber's machine.

This solution, however, has some disadvantages. If you distance the event class from the publishers, you introduce extra—and expensive—round trips across the network. In addition, the single machine hosting all the event classes and subscribers becomes a hot spot for performance, so the machine's CPU and operating system must handle all the traffic. Your product also

INSTALL EVENT CLASSES AND SUBSCRIBERS ON ONE MACHINE

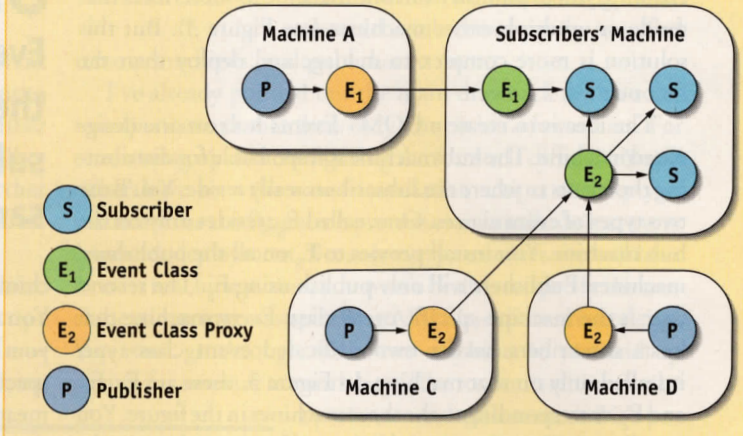

Figure 2 | You can install every subscriber and event class on one machine and have the publisher access the event classes through proxy application. This solution is easy to set up, but it lacks design flexibility.

MACHINE-SPECIFIC EVENT CLASSES

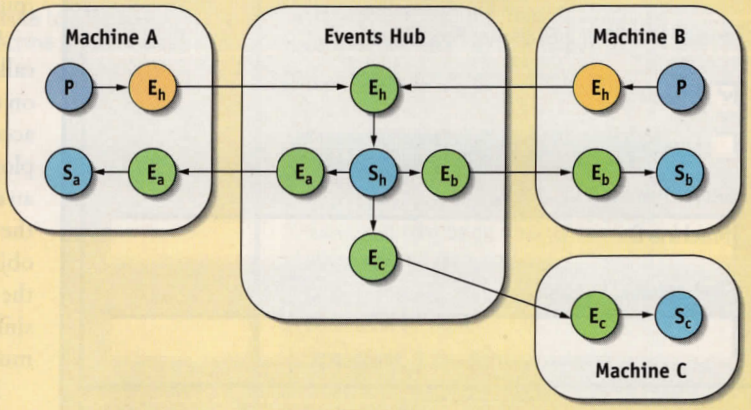

Figure 3 | A dedicated hub machine distributes the events from a publisher on one machine to subscribers on many machines using machine-specific event classes.

RESOURCES

- MSDN: http://msdn.microsoft.com
- "A Hands-On Look at COM+ Events" by Jeff Prosise, *VCDJ* January/February 2000

won't have load balancing, which is a major reason for distributing your components in the first place. Another problem: The subscribers might not be deployed ideally. Without the constraint of having to reside where the event classes are, you might have put the subscribers somewhere else—maybe on the same machine where the database is, if the subscribers must access it frequently—so performance might suffer. Finally, the subscriber-machine solution becomes a single point of failure in your system.

2. Machine-Specific Event Classes

This solution allows you to distribute your subscribers anywhere, according to whatever design preference you have, enabling you to publish from one machine to subscribers that reside on multiple other machines (see Figure 3). But this solution is more complex to manage and deploy than the first one.

The idea is to create a COM+ Events hub on one designated machine. The hub machine is responsible for distributing the events to where the subscribers really reside. You'll use two types of event classes. One, called E_h, resides only on the hub machine. You install proxies to E_h on all the publishers' machines. Publishers will only publish using E_h. The second type is the machine-specific event class. Every machine that hosts subscribers has its own dedicated event class type, installed only on that machine. In Figure 3, these are E_a, E_b, and E_c, corresponding to the three machines in the figure. You install a proxy to every machine-specific event class on the hub machine.

Each event class in this solution supports the same set of sink interfaces. When a publisher on machine A wants to publish an event to subscribers on machines A, B, and C, the publisher on machine A creates an instance of the E_h event class, which only creates a proxy, and fires on the proxy. The E_h proxy forwards the call to the location where E_h executes—on the hub machine, which has a hub subscriber (S_h) that subscribes to the E_h event. S_h handling the event creates all the machine-specific event classes (E_a, E_b, and E_c) and fires that particular event on them.

Because the hub machine has only proxy installations of the machine-specific event classes, the event is distributed to multiple machines where local subscribers—the "real" subscribers—handle the event. This solution gives you complete freedom in locating your subscribers—a big advantage.

However, this flexibility comes with a hefty price. When you publish, you encounter many expensive trips across the network. Even if every subscriber is on the publisher's machine, the publisher still must go through the hub machine. You must duplicate this solution for every kind of event class you have, and you end up with separate sets of machine-specific and hub-event classes. This solution's complexity means you'll probably end up with a deployment, administration, and maintenance nightmare on your hands.

Unfortunately, the COM+ Events service has a limitation—the event class and all its subscribers must all be on the same machine.

3. COM+ (Partial) Routing

This last solution takes advantage of a feature provided by COM+—but it only works with persistent subscribers, so you haven't solved the whole problem. If your application uses transient subscribers, which is likely, you must use the first or second solution. This solution resembles the hub machine solution, so to distinguish between them I'll call this the routing solution.

COM+ provides a field for every persistent subscription called "Server name" on its properties page, which is located on the Options tab in the Server name field (see Figure 4). To access any properties page on any item in the COM+ Explorer, right-click on the item. Whenever COM+ publishes an event to a persistent subscriber, COM+ checks the value of the Server name property before creating the subscriber object. If the property isn't an empty string, COM+ creates the subscriber on the specified machine, fires the event on the sink interface, and releases the subscriber. Routing events to multiple machines takes advantage of this feature.

Instead of using machine-specific event classes, as described in the second solution, you use machine-specific persistent subscriptions in the routing solution . For example, suppose you have a publisher on machine A and a subscribing component, called MySubscriber, that you want to deploy on

ROUTE A SUBSCRIBER ACTIVATION

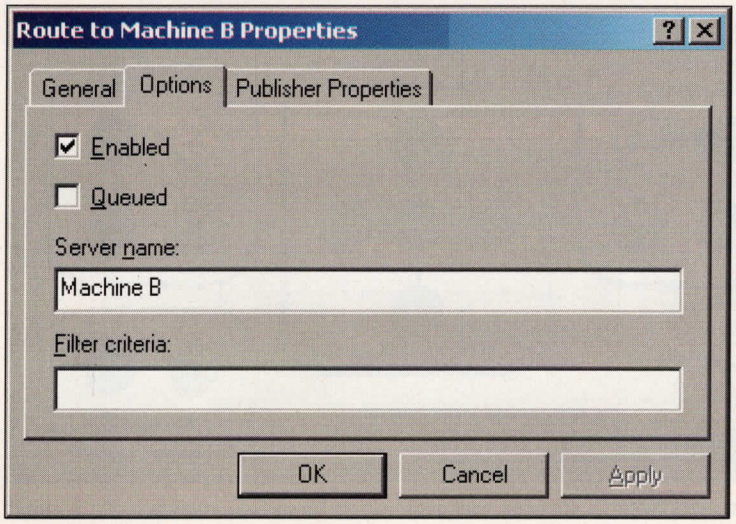

Figure 4 | You can instruct COM+ to create the subscriber object on the machine specified in the Server name field on the persistent subscription properties page.

MACHINE-SPECIFIC SUBSCRIPTIONS

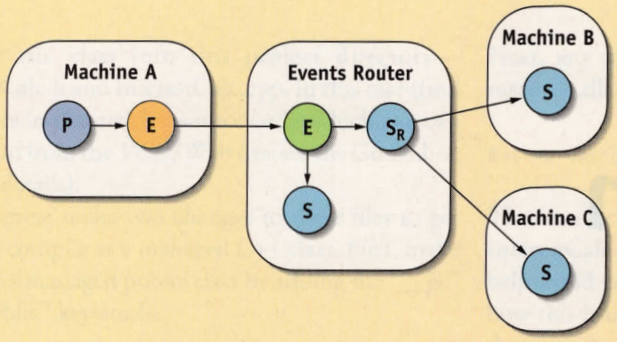

Figure 5 | Using machine-specific subscriptions and a designated router machine, you can distribute events for persistent subscribers. This solution is available only for persistent subscribers.

machines B and C. The publisher will publish using an event class called E. On machines B and C you add subscriptions to the event class to the locally installed copies of MySubscriber (shown as "S" in Figure 5). You then install the MySubscriber component on another designated routing machine, together with the event class E, and install only the proxy to E on machine A (see Figure 5).

To the installation of MySubscriber on the router machine (called S_R in Figure 5), you add machine-specific subscriptions. For every MySubscriber deployment on another machine (such as B or C), you add a subscription, and you redirect the invocation to that machine using the Server name field (see Figure 6).

Now, when the publisher on machine A creates a proxy to the event class and fires an event on it, the call goes to the router machine. COM+ inspects the subscriptions on the router machine for the event class, detects the Server name in the subscriptions, creates the subscribers on the specified remote machines, and publishes to them.

This solution uses machine-specific persistent subscriptions.

I've already pointed out the main drawback of this solution (persistent subscribers only), but it has a few others. For one, setting up and configuring the system is difficult. You'll either have to write some installation scripts to help you automate the configuration process or manually do it at every customer deployment. Because every customer site will have its own machine names, you won't be able to specify the machine names in your application MSI file, exported for release. You must go through the router machine, so you end up paying for an extra network hop. The router machine can be a performance bottleneck, and it's potentially a single point of failure. VCDJ

THE ROUTING SUBSCRIPTIONS

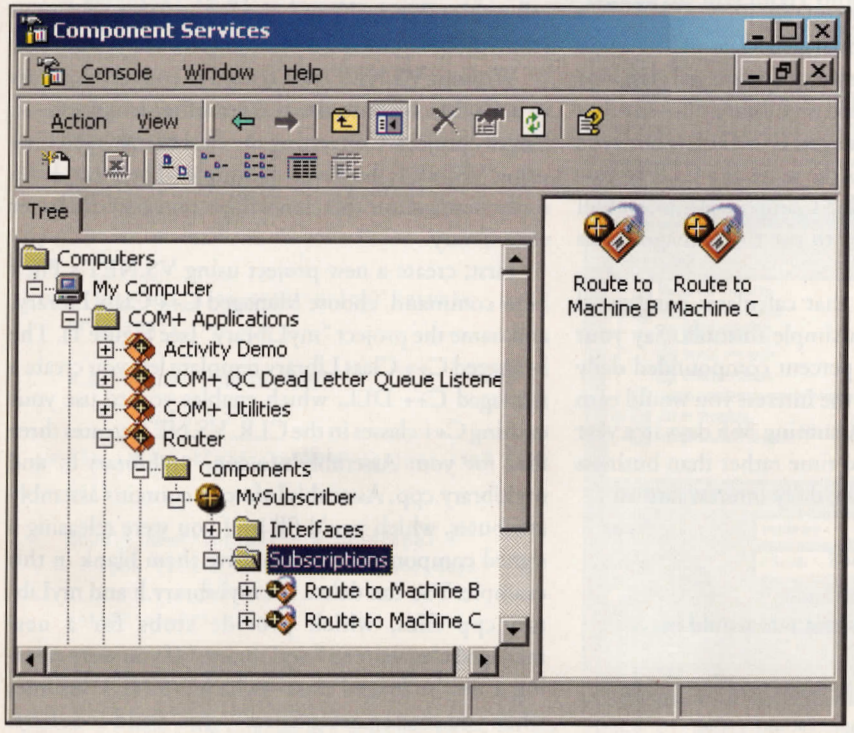

Figure 6 | On the router machine, you install another copy of the subscriber and add a subscription for every machine you have the subscriber deployed on.

About the Author

Juval Lowy is a seasoned software architect who spends his time publishing and conducting classes and conference talks on component-oriented design and COM/COM+. He was an early adopter of COM and has unique experience in COM design. This article is based on excerpts from his upcoming book on COM+ and .NET (O'Reilly). E-mail him at idesign@componentware.net.

GO ONLINE

Use these DevX Locator+ codes at www.vcdj.com to go directly to these related resources.

VC0104 Download all the code for this issue of *VCDJ*.

VC0104PC_T Read this article online. DevX Premier Club membership is required.

Want to subscribe to the Premier Club? Go to www.devx.com.