Advanced Systems Monitoring

ProActive Systems with OPC DataHub

Software Toolbox
International Corporate
Headquarters, USA

148A East Charles Street
Matthews, NC 28105 USA
www.softwaretoolbox.com

TOLL FREE: 888-665-3678
GLOBAL: 704-849-2773
FAX: 704-849-6388

Table of Contents

Introduction

Today's interconnected systems are comprised of multiple software and hardware "parts" that combined make-up a system. Problems occur when one of those "parts" fail.

In this document we will look at how you can use the OPC Data Hub to identify such failures, send notifications, and attempt to fix certain types of problems.

Intended Audience

This document does involve some programming, albeit minimal, so some development experience would be helpful, although not required.

Systems integrators and network managers alike would benefit highly from this document.

Terminology Used

The terms "process" and "application" are interchangeably used throughout this document, but mean the same thing.

Leveraging Windows Performance Counters

Within Windows you will find the Performance Monitor tool. This tool is extremely powerful and is often overlooked. The information available is invaluable and if used smartly, could save you a lot of “trouble” down the road, whether that means avoiding expensive downtime, or avoiding the need to replace failing hardware etc.

What kind of information is available?

Virtually everything about your computer system is available, such as:

- CPU, memory and disk-space utilization
- Network consumption and available bandwidth
- Detailed information about each application
- and much, much more…

How can this information be of any use to me?

First, how did you manage without having access to this information in the first place?

Here are just a few examples of how this information could be extremely valuable in your system:

- Watch a key application, such as a custom application, if/when it crashes automatically restart it
- Send an email notification when CPU or Memory usage goes above a certain threshold
- If you have multiple Network Cards, observe the bandwidth of them both and adjust your systems configuration to balance-out their bandwidth requirements.
- Trend or log this information over time, to build a profile of the system so that you can reference it later.

How we will use this information in this document

We are going to focus on 2 key areas of a dynamic and pro-active system:

- Sending email alerts when resource utilization is above a certain threshold
- Restarting a process if/when it fails

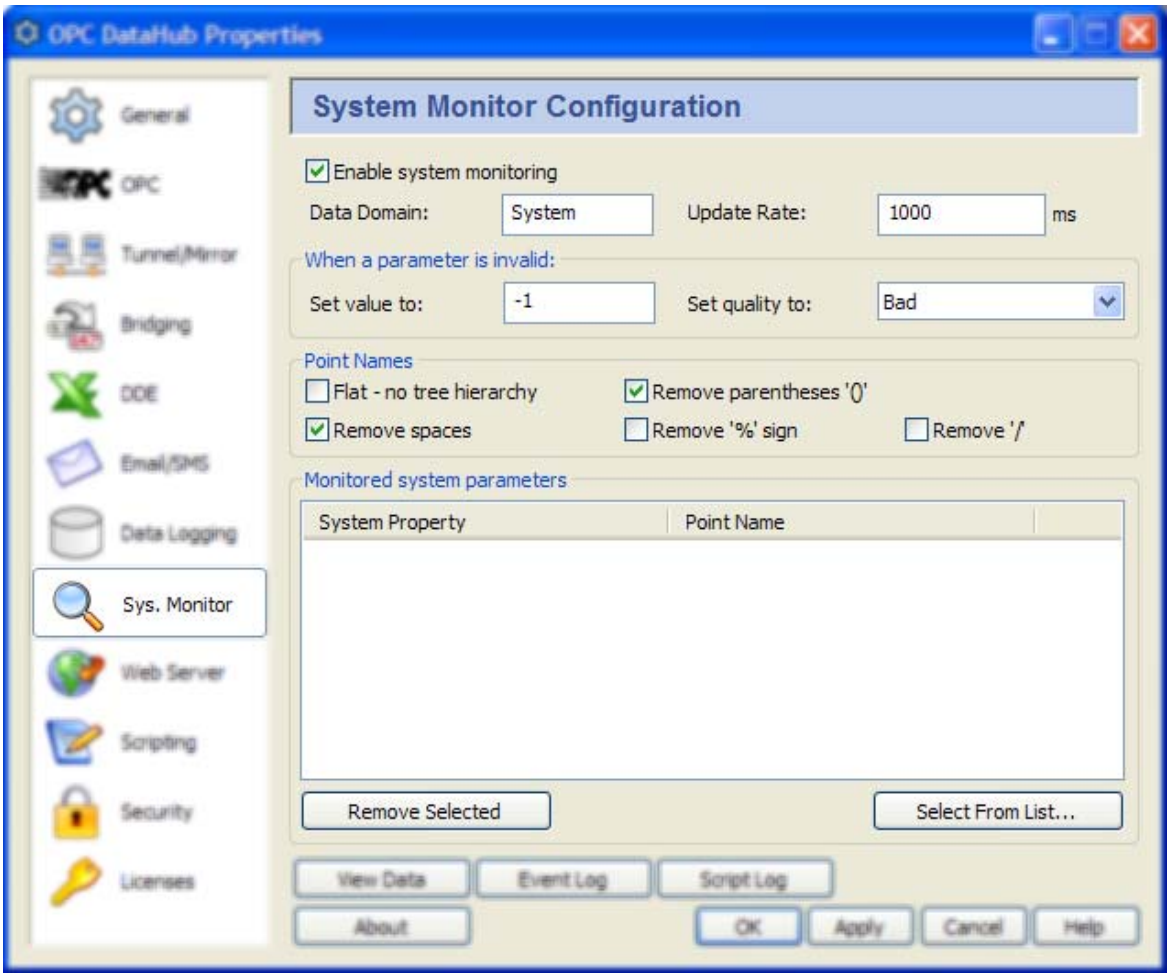
In our system we will have:

- OPC DataHub, monitoring and maintaining our “system”
- TOPServer OPC Server
- Windows Notepad

Identifying & Configuring your Performance Counters

Lets open the OPC DataHub and begin by configuring the Performance Counters to expose as OPC Items.

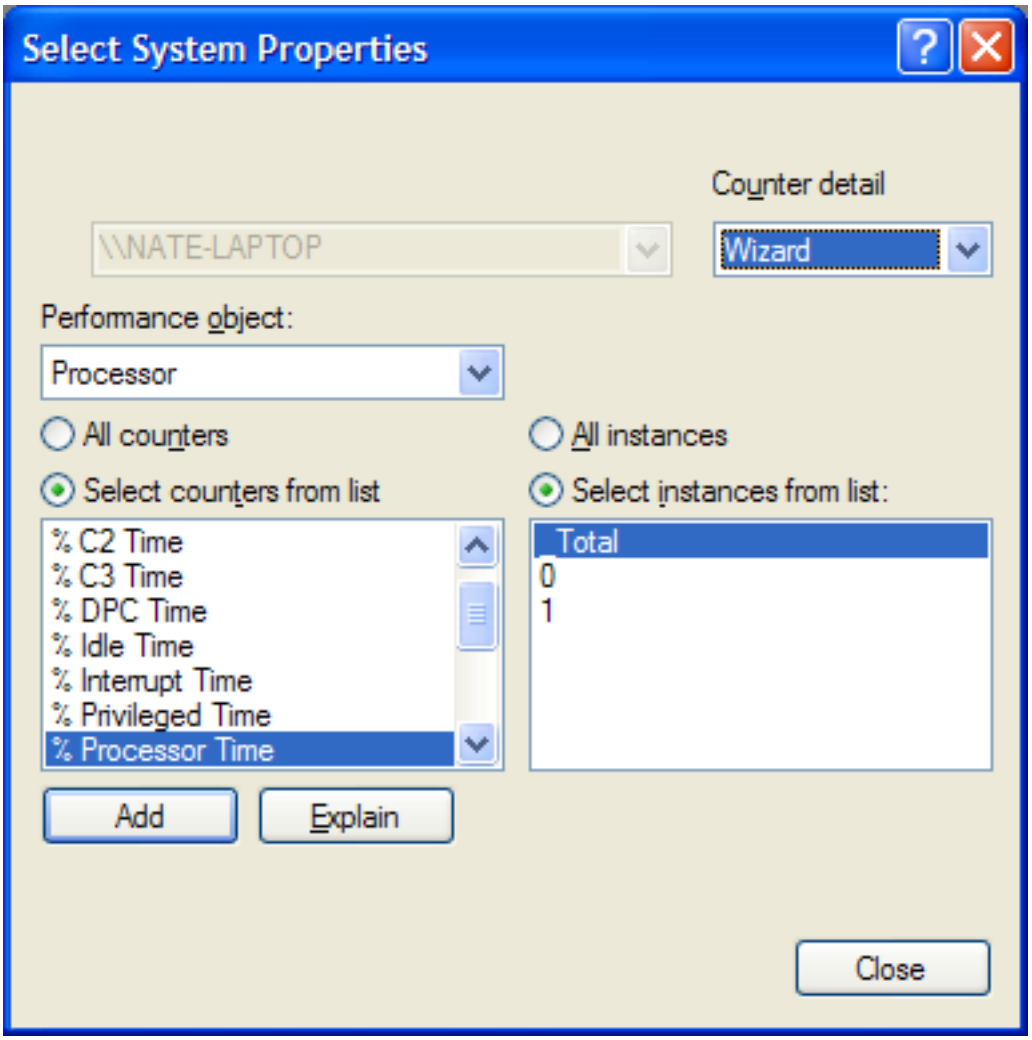
1. Click on the **Sys. Monitor** tab on the left-side of the properties window:



2. Next, we will set the options: Leave **Flat – no tree hierarchy**, **Remove '%' sign**, and **Remove '/'** unchecked. Put a check in the box of the other options.

3. Now we will add our Performance Counters…

4. Click the **Select From List…** button, which will open the following dialog:



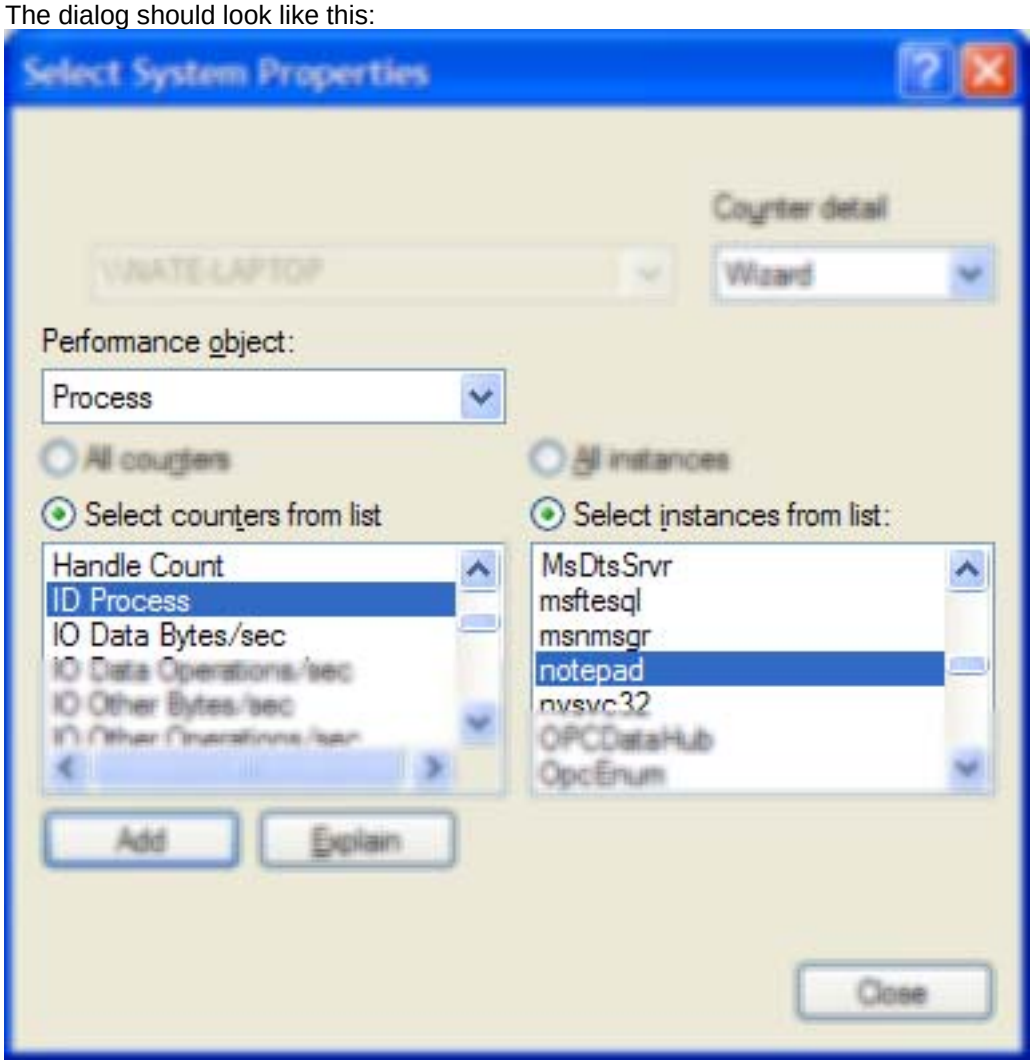
5. The first counter we will add, will be for monitoring the Notepad application to ensure that it is still running. Later, we will create a script to automatically re-launch it if and when it fails.

    a. Click on the **Performance object** drop down and pick **Process**.

    b. Now select **ID Process** from the list under *Select counters from list*.

    c. Now select **Notepad** from the list under *Select instances from list*.

    d. The dialog should look like this:



    e. Click the **Add** button.

6. The next counter to add will be for our CPU utilization measurement:

    a. Select the **Processor** option from the *Performance object* drop down.

    b. Select **% Processor Time** from the list under *Select counters from list*.

    c. Select _**Total** from the list under *Select instances from list*.

    d. Click the **Add** button.

7. The last counter will be to monitor the available memory:

    a. Click the **Memory** option from the *Performance object* drop down.

    b. Select **Available Mbytes** from the list under *Select counters from list*.

    c. Click the **Add** button.

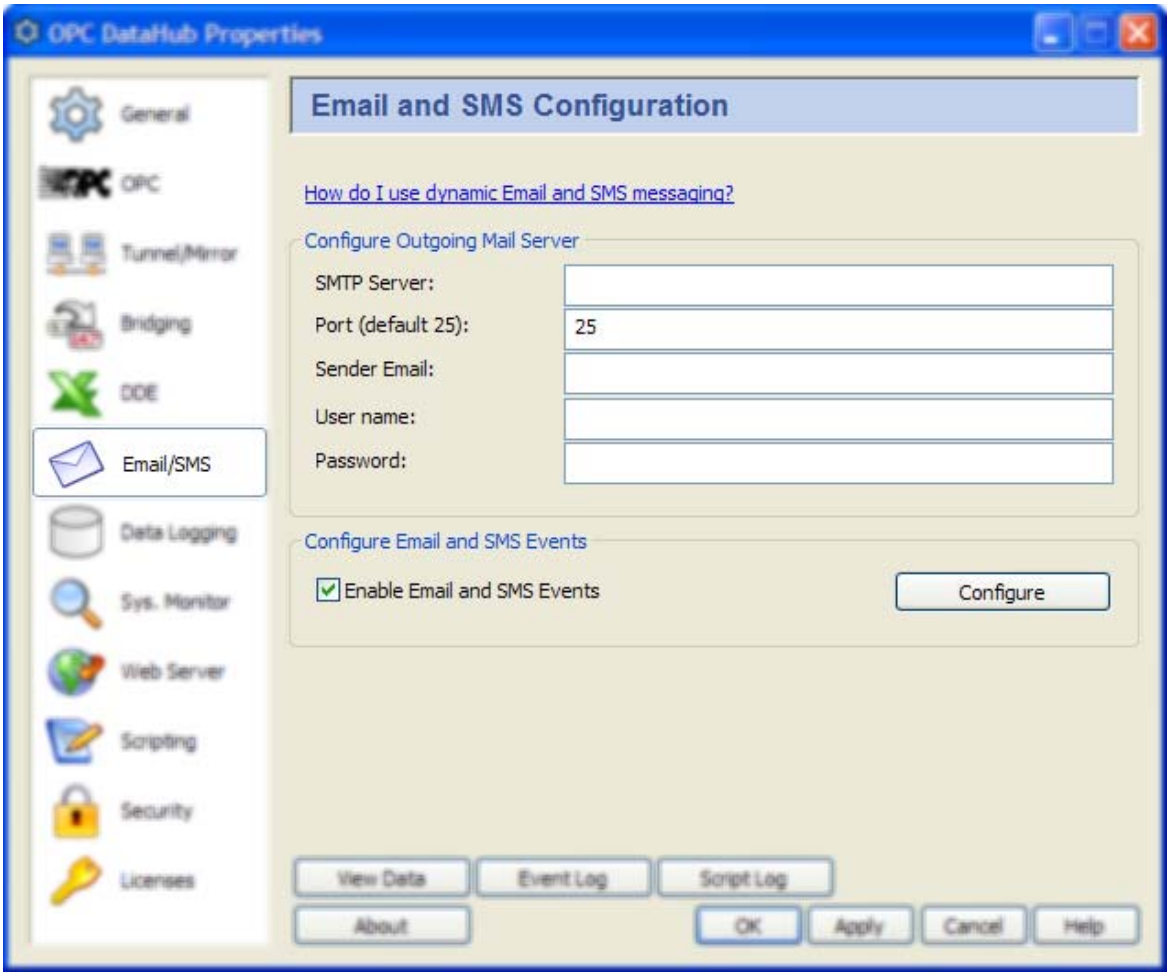
8. Close the dialog and then click the **Apply** button at the bottom of the DataHub properties screen.

Email Notifications on High Resource Utilization

Now that we have added our performance counters, and there are many, many more that you could add, we will take a look at how to send an email notifications based on our criteria.

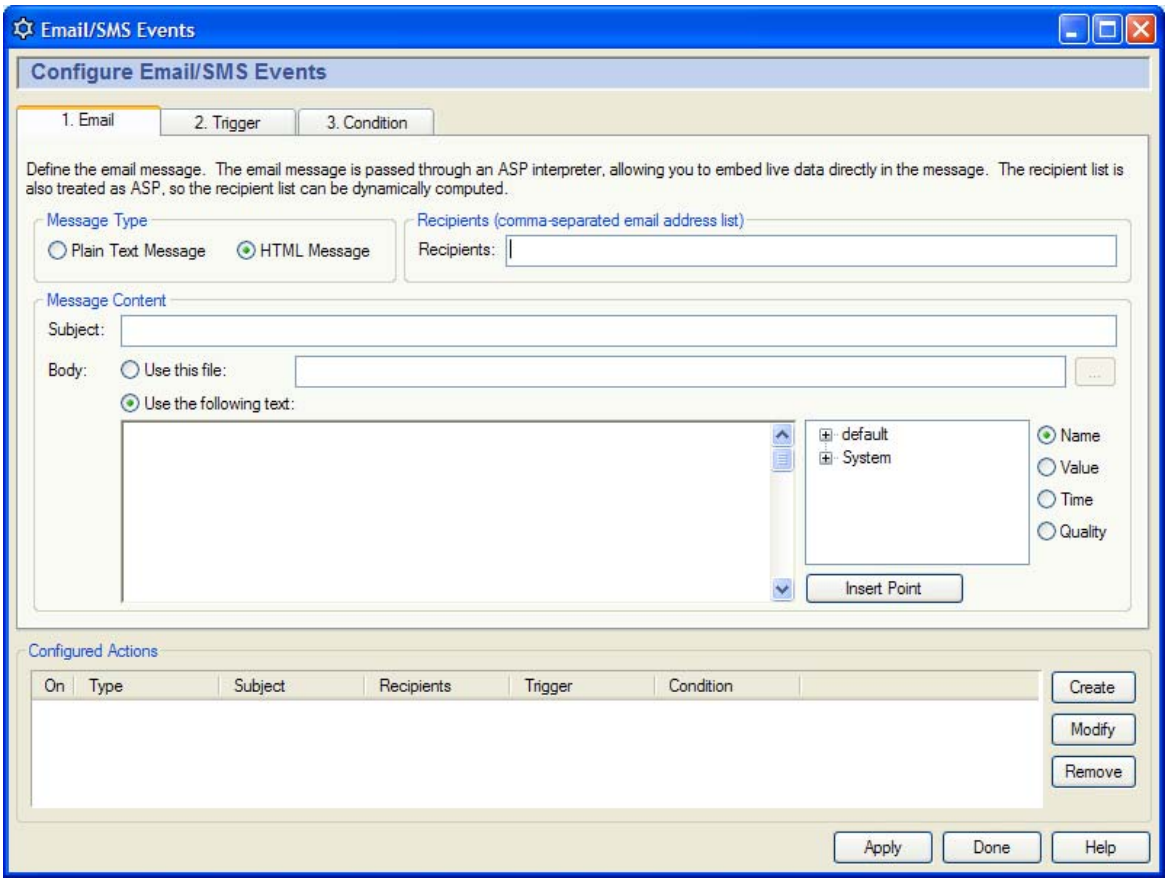
No coding or scripting is necessary here.

a. Open the OPC DataHub properties and go to the **Email/SMS** tab:



b. Configure your Post Office setting by specifying the **SMTP Server** and **Sender Email**. That's usually all that's needed.

c. Now click on the **Configure** button:



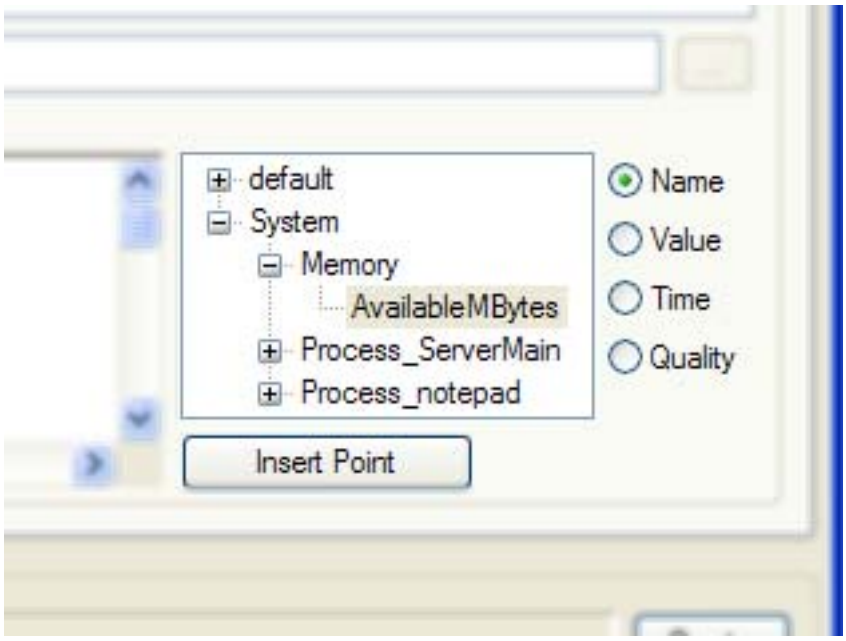
d. We will now specify a simple email message to indicate that resource usage is too high:

   a. Enter the email address(s) in the **Recipients** field of those people who should receive these notifications.

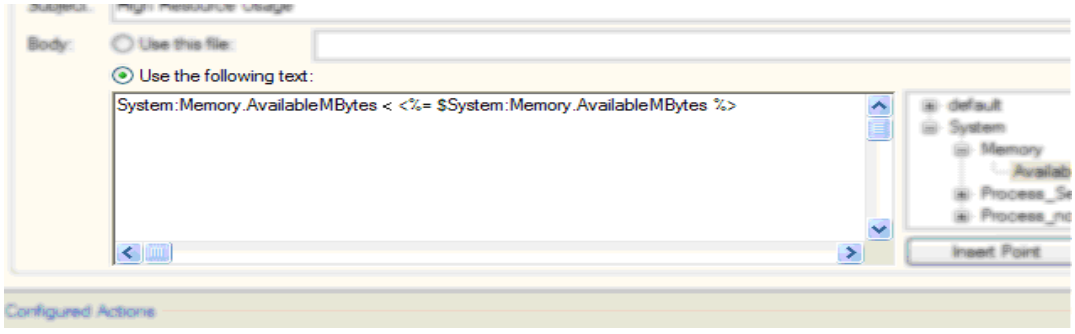
   b. In the **Subject** enter “High Resource Usage” (without the quotes)

   c. In the **Body** section select the **Use the following text** option.

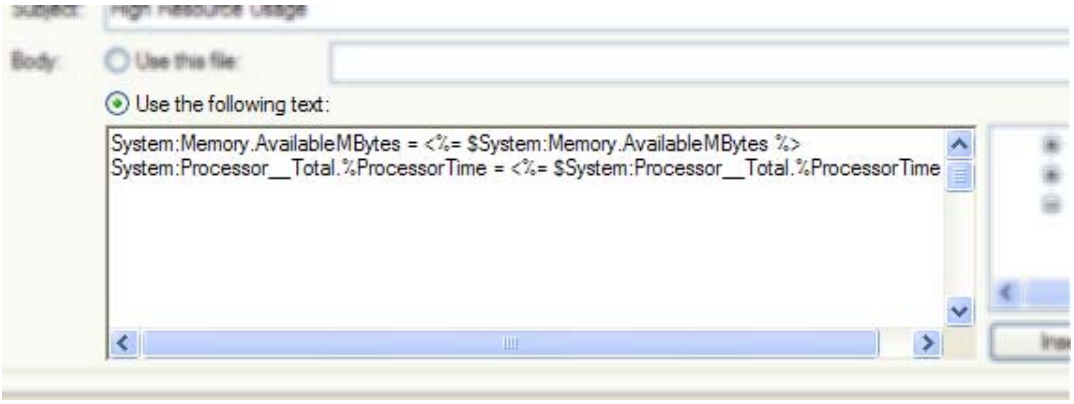
d. To the right of the body text, expand the **System** option and then highlight **AvailableMBytes**. Click the **Name** option and then press the **Insert Point** button:



e. You will see the name added to the Email Body. Edit the email body by placing the cursor at the end of the line, pressing space, and then pressing an '=' sign and space again. Now press the **Value** option and then press **Insert Point**. The email body should now look like this:



f. Now add another line to the email body and specify the values of the other resource items… your message should look like this:

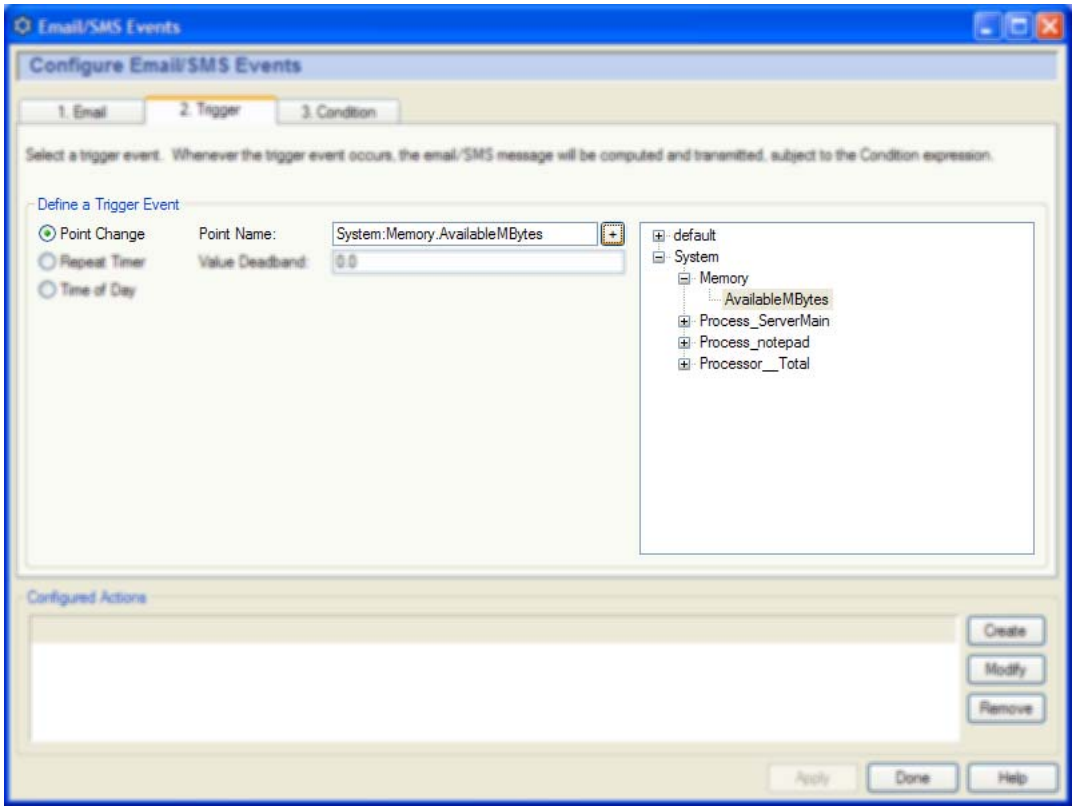


At this point we have specified the email message. We now need to specify when to send it.

a. Click on the **Trigger** tab at the top of the screen.

b. In the treeview (right-side of the dialog) you will expand the **System** node, then expand the **Memory** node, finally you will select the **AvailableMBytes** option.

c. Click on the **+** button beside the *Point Name* option, so bring the option into the field:

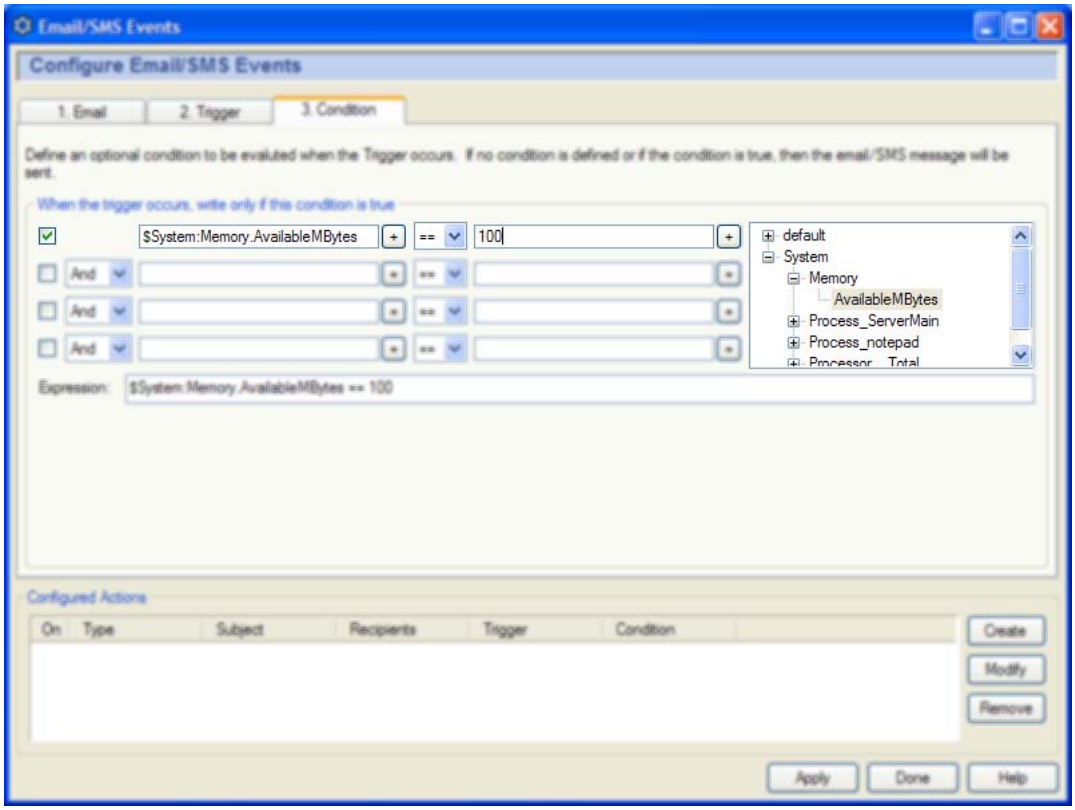


Please review the **Important Considerations** on oage **19**,

d. Now click on the **Condition** tab.

e. We will now specify the condition of our triggers. Using the treeview to the right, expand **System**, then expand **Memory** and then highlight the **AvailableMBytes** option. Click the left-most **+** button on the first condition line. Then leave the **==** symbol in the condition dropdown. Specify a value of **100** in the condition field. We only care when there is less

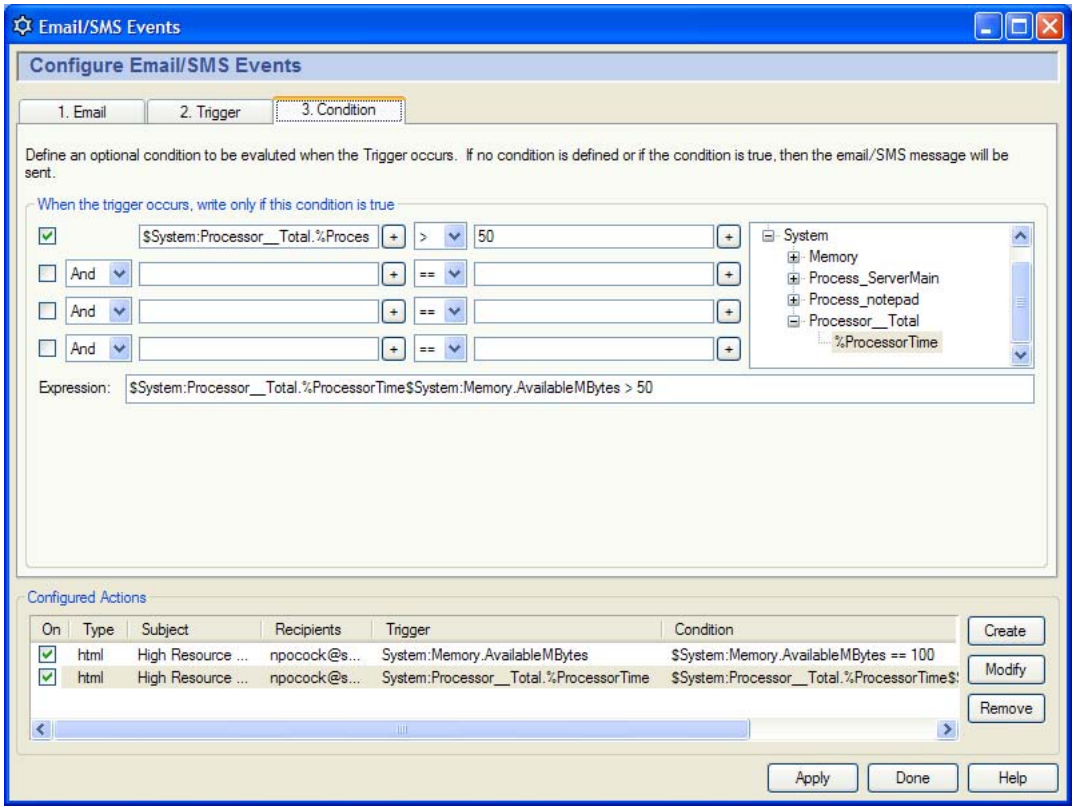
than 100mb of physical memory remaining:



f. Click the **Create** button at the lower-right of the dialog to create this trigger and condition.

g. Now add another trigger and condition for the other item, the **CPU Utilization**.

h. Your dialog should look like this when finished:



i. Click the **Apply** button, followed by **Done**. This will close the dialog.

Automatic Restarting of Failed Processes

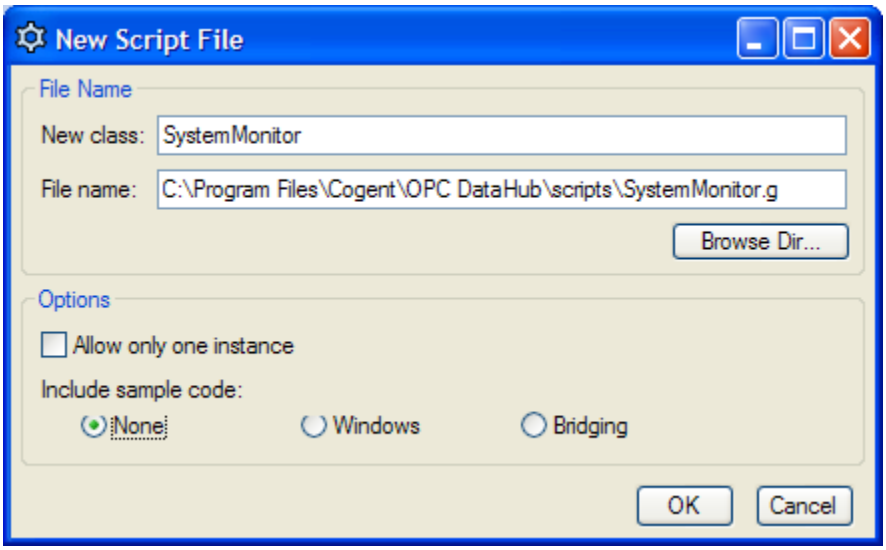
In this example we will monitor a process (NOTEPAD.EXE) and will automatically re-launch it if we detect that it is no longer running.

Whenever a process is launched it is assigned a numeric Id, which we can monitor. Once that process is shutdown the value goes to -1. This means, we can monitor a process id and whenever the value goes to -1 we can take action to restart it.

This will require just a little scripting.

a. While in the OPC DataHub Properties, click on the **Scripting** tab.

b. Click the **New** button and then give your script a name, such as *SystemMonitor* for example. You do not want sample code included:



Click the **OK** button to continue.

c. When our script is first loaded we want to setup an event handler to respond to changes to our performance monitor tags. To do this:

   a. Locate the `method <filename>.Constructor ()` method.

   b. Within this empty method, enter the following line:

```
.OnChange(#$System:Process_Notepad.IDProcess,
`(@self).CheckProcessRunning($System:Process_ServerMain.IDProcess));
```

      i. Where :-

      `System:Process_Notepad.IDProcess` = name of the tag created to monitor notepad

      `CheckProcessRunning` = name of a method we are about to create.

   c. Scroll up to an empty line and then create the following method:

```
/* Use methods to create functions outside the 'main line'. */
method MyApp.CheckProcessRunning ( value )
{
	if( value == -1 )
	{
		ShellExecute(0, "open", "notepad.exe", "", "c:\\", SW_SHOW);
	}
}
```

   d. Save and close your script.

While this script is very minimal, it simply checks the value of the specified item and blindly launches Notepad, you could enhance it to so that the line *.OnChange* calls different methods for different processes to start.

Conclusion

This tutorial barely scratched the surface of the power and potential for what you can create with this important information.

In this document we saw how to:

a. Send email notifications when resources are too high or too low

b. Automatically restart lost processes

In this area, your imagination is your only limitation.

Important Considerations

When using notifications such as those mentioned in section **Email Notifications on High Resource Utilization**, you could find your inbox full of notifications. Therefore it is important to prevent this by implementing some method of reducing the notifications to a single notification, or to notify only when a certain threshold has been surpassed. Many methods exist, but perhaps the best would be to utilize the **Deadband** feature within the Email properties window, specifically the **Trigger** tab. This would then cause the trigger to occur only when the value-change is within a particular “range” or “fluctuation”.