

# newsletter



**PGCG**

Pretty Good Consulting Group

Got a problem? We create solutions.

**EOM News and Views**  
**Issue 19, Early Summer 2010**

## *Sentinel Alert Service external blinky thingy*

The Sentinel Alert Service exposes operational messages and problems occurring with the EOM service. Wouldn't it be nice to have an obnoxious blinking light to notify you that something is wrong?

We need to start this discussion with an impassioned plea: Please install the Sentinel Alert Service (SAS) when you install the EOM Service. Two customers in the last month have been caught with an error that was not immediately visible because they did not have the SAS service installed. While error information is in the EOM logfile, the visible, changing-color icon and Alert Explorer interface makes problem resolution much more efficient. The SAS service install is quick, painless, and needs no additional configuration to run.

As stated in the February 2009 Newsletter, the Alert Service does a nice job of exposing issues when you are looking at the EOM client and Alert Explorer, but what if your eyes are not glued to those interfaces? That Newsletter went on to show how to configure SAS to send an email via the Command Template when there was an alert. Great, but what if you aren't watching email?

An article in a recent Popular Science magazine got us thinking about self-contained interactive devices. The concept is that the device is sent information and based on the programming of that device it reacts to the information. This seemed like a good excuse to play with new toys and give some visibility to the usefulness of the Alert Service. So, this discussion will focus on a fairly simple first step: how can we use an interactive device to notify people that there is an EOM Alert?

### Hardware & Software

At the heart of the solution described in the Popular Science article is the "Arduino" microcontroller. Much more than just hardware, it is "an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board." It provides the ability to create stand-alone devices reacting to on-board sensor information, as well as communicate with other intelligent devices (like a PC). The Arduino

board has a USB-based serial interface that is used to upload the program code that runs the board, so let's start with that interface for our solution.

## *Contents*

SAS Device .....	1
EOM 8.1 feature .....	4
Quick Hits.....	5
Contact information .....	5

The Alert Service Policy Editor lets you create Templates for Command, Modem, and Serial device notification which are invoked when a template is configured for a particular alert ID. While we could have used the Command Template and custom programming to programmatically send information to the device, the existing Serial Template already provides a mechanism to send data via a serial connection.

The Arduino programming environment runs on Windows, Macintosh, and Linux, providing a C/C++ like language to react to incoming data and control on-board functions. You download the compressed file, decompress into a directory, and then execute the Arduino.exe program. You write the code in the programming environment, compile, and upload the compiled code to the board. As soon as the upload completes, the program code starts running automatically.

### Notification

One way to notify people of an alert is to send a 20,000 volt charge between their thumb and forefinger. While certainly possible, we will instead use 3 different colored LEDs, using color and blink rate to discern the importance of the alert. This is a very basic use of the capability of the Arduino board.

### Putting the pieces together

We outfitted the Arduino board with a "shield" so that the LEDs and resistors could be securely held in place (see pictures and screen shots on the next page).

The USB cable connects the Arduino board to the PC, which is used for both uploading the program as well as the live serial connection for the SAS service. When plugged in for the first time, you will be required to supply the "USB Serial Port" driver that comes with the Arduino programming environment (/Drivers subdirectory or Windows will eventually find the right drivers).

### Arduino programming

The Arduino programming environment does not have all the bells and whistles of Microsoft Visual Studio, but works well for the intended purpose. Every Arduino program has a setup and loop routine. The `setup()` procedure is used to initialize variables as well as start the serial connection. The `loop()` procedure is where you look for input on the serial connection and then act on that input (i.e. turn on lights in out case). For our example we have code that controls parsing for alert IDs and alert severity, controlling the blinking function of LEDs for current alerts, and cleanup code when alerts are cleared. As you might expect, the higher the alert severity the faster the LEDs blink and the more vibrant color of LED (blue, green, red).

### SAS configuration - serial device

The Alert Service needs to know which serial port to use to connect to the Arduino board. When the USB Serial Port driver is installed a port will be selected - so go to the Windows Device Manager, look in the "Ports (COM & LPT)" section for the port assignment. It will look something like "USB Serial Port (COM7)". Next, open the SERVICES file usually found in the c:\WINDOWS\SYSTEM32\DRIVERS\ETC directory and make an entry for the SAS service as follows:

```
SASCOM7 COM7/eai_serial 8BIT NONE 1STOP 9600 #COM7 port for Serial USB
```

The "SASCOM7" string is how the SAS service references the serial port information.

### SAS Configuration - Alert Policy

Configuring the SAS Alert Policy is pretty straightforward, set up the serial path and add actions that raise & clear alerts:

- Create the Serial Template, note that Service Name references the name used in the SERVICES file above ("SASCOM7").
- Add Actions for the Alert IDs you want to highlight with blinking lights. The Action(s) must send a text stream to the attached serial device that can be recognized by the code written for the Arduino board. For our example, when a DDA GetUserInput message is used, SAS raise an alert and will replace the Text value of "\\_ALERTID:\\_SEV\" with "DdaGetUserInput:major". When that alert id cleared, SAS will clear that alert and replace the Text value of "\\_ALERTID:clear" with "DdaGetUserInput:clear".

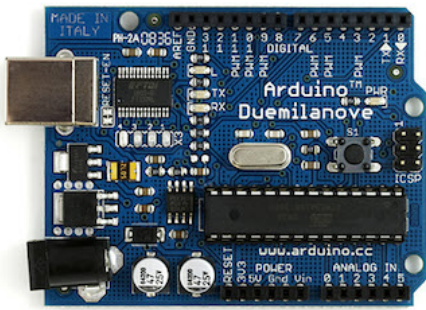
After everything is installed and configured it is very likely that you will have to restart the SAS service (so it grabs the serial port) and then activate the Alert Policy that references the Serial Template. From then on, when one of the Alert Ids configured with an Action occurs, the Text string is sent to the Arduino board via the serial connection, parsed, and acted upon (blink lights). When an operations person clears that alert (or the alert clears by itself) the Text string for the clear action is sent to the Arduino board.

### Next Steps

As you can see it really is not that hard to expose EOM alerts with an external device. Keep in mind that the blinking LEDs could be replaced by any number of things: buzzers, motors, air horns, other circuitry, and so on. The really cool part of this solution is that it can be built upon using Arduino-ready components: for example, instead of using a USB-based serial connection you could equip the Arduino board with a wireless attachment and place the blinky light apparatus anywhere that has wireless access (note that the Arduino code would change to include networking commands).

Feel free to contact us if you have a need for this device or if you have other interesting applications.

Arduino board



Picture by SPARKFUN

Board, shield, components



Device in simple diffuser



### Configure the Serial Template

The screenshot shows a tree view on the left with 'Alert Policies' expanded to 'OutputManager7.1' and then 'Templates'. The main pane shows a table of templates and a properties section for the selected 'serial1' template.

Template Name	Type
EmailMessage	Command
serial1	Serial

Properties for serial1:

Host	
Initialization	
Service	SASCOM7
Template Name	serial1
Type	Serial

### Configure the Actions for Alert IDs

The screenshot shows a tree view on the left with 'CustomAlert9' selected. The main pane shows a table of actions and a properties section for the selected 'Raise' action.

Action Event	Template	Text	Delay	Enabled
Clear	serial1	\\_ALERTID\\clear	0	True
Raise	serial1	\\_ALERTID\\_SEV\\	0	True

Properties for Raise action:

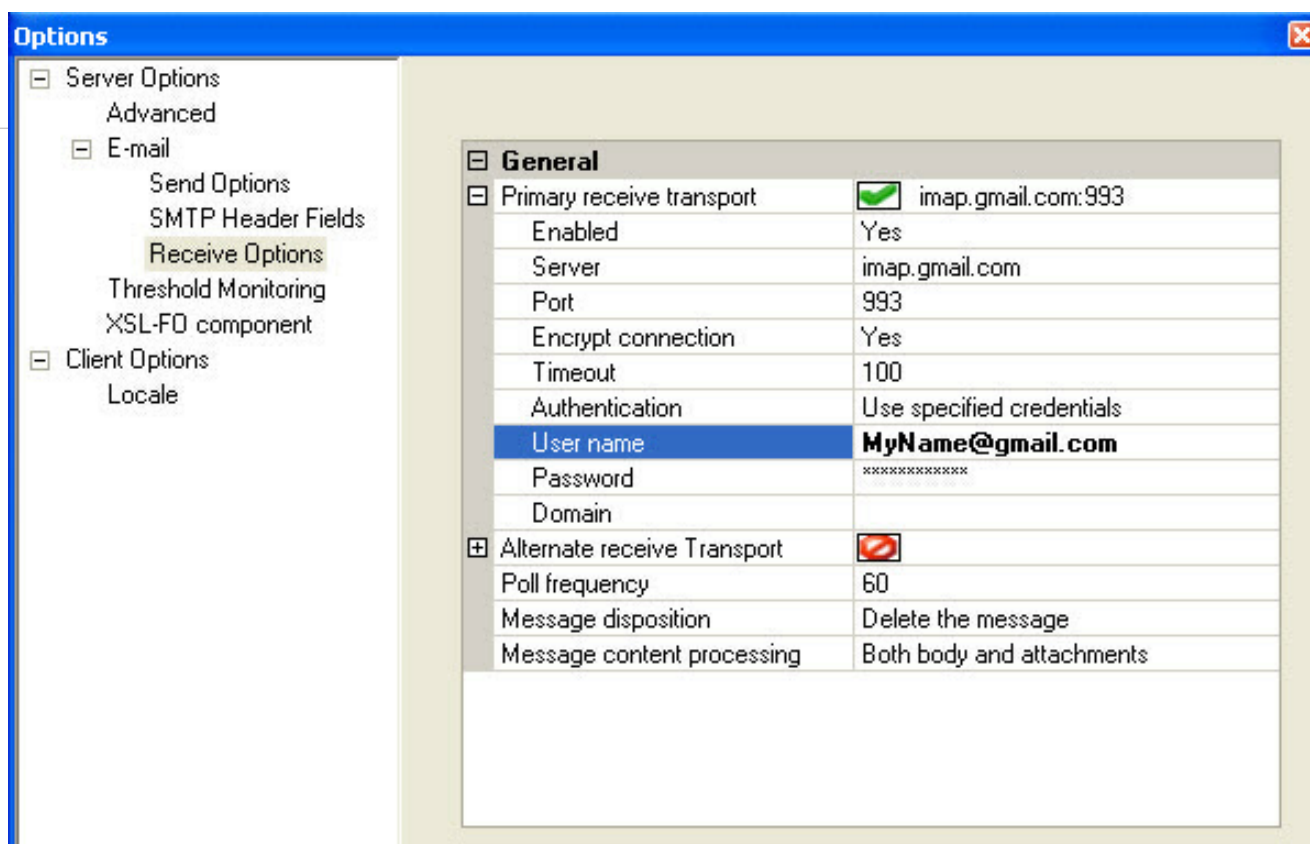
Action Event	Raise
Delay	0
Delay Unit	seconds
Enabled	True
Template	serial1
Text	\\_ALERTID\\_SEV\\

## EOM 8.1 preview: Receive email

We noted two new EOM 8.1 features in the last Newsletter: ability to create TIF files directly and much better ability to manage Custom Jobs. This time we bring forth the new capability to receive email messages.

"Huh?" you are thinking. "Can't I already receive email messages with EOM?" Yes, but you are forced to use the GUI, logon-to-Windows Legacy Assistant program. The new feature uses the industry standard IMAP protocol and does not require the Legacy Assistant. Configuration is very easy and, like the SMTP interface for outbound messages, is rock solid.

The configuration for the IMAP interface is set from the Configuration Explorer, Tools / Options / Receive Options.



When a file is received, the Host System / IP Address / File Server field is set to "IMAP" for your File Masking pleasure. As you would expect, the Originator, Subject, and User Tag fields are appropriately set. Very simple.

On another note... In case you were wondering, the feature that allows outbound SMTP e-mail with digital signature and/or encryption is done, but will likely be released later this year as a separately orderable component. The delay is due to legal issues that need to be worked out regarding export of encryption capabilities.

# *EOM 8.1 has been released!*

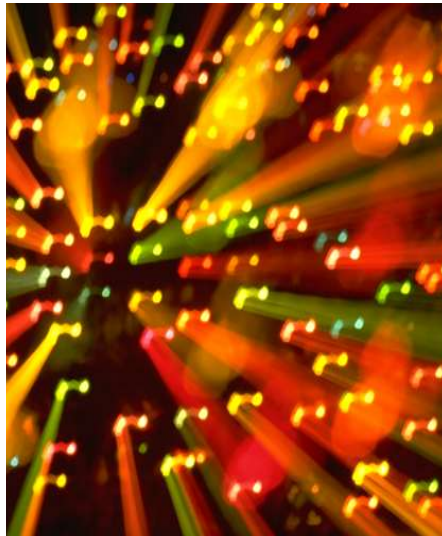
New features and an accumulation of (very few) bugs. This feature should be orderable by the time you read this note. A few of the features were highlighted in last month's Newsletter, too. EOM 7.0 and EOM 7.1 are officially decommitted October 31st, 2010.

## *Who are we?*

Highly skilled, creative, solution provider focused on the Unisys Enterprise Output Manager product (formerly known as DEPCON) with a sense of urgency sums up who we are and what we do. We provide general Enterprise Output Manager consulting, migrations, upgrades, configuration, training, and custom programming.

On-site services, remote services, and general consulting are available now.

Why use PGCG? Deep knowledge of the EOM product integrated into a variety of customer environments sets us apart. Our customers production environment depends on solid, working solutions that we provide.



Pretty Good Consulting Group, LLC  
4235 Ivy Court North  
Lake Elmo, Minnesota 55042

[www.PrettyGoodConsultingGroup.com](http://www.PrettyGoodConsultingGroup.com)

## *Quick Hits*

EOM guru Bill Harberts of EMC Insurance brought a nifty feature to our attention that was not highlighted in the EOM 8.0 release notes. Keyword substitution now includes <DateTime> and <DateTimeUtc>. While there are fancy date/time options using the "%?" syntax, these new keywords are recommended and have significantly more capability. Even better, these new keywords can be used ANYWHERE keyword substitution is allowed and allow for the partial keyword selection.

Bill also noted how he used the EOM FTP capability to place files on an FTP server by simply printing to EOM. The User Tag values hold filename and path information, a little keyword substitution in the Transfer Attribute and ta-dah, he has a robust, stable mechanism to FTP files.

We have run into a major problem when trying to use Adobe 9.0 to create PDF files with EOM. A new, hidden window is created by Adobe 9.0 that EOM does not ignore. No workaround as yet.

Have a suggestion for "How do I ...?" Write a brief description and send it to [SteveD@PrettyGoodConsultingGroup.com](mailto:SteveD@PrettyGoodConsultingGroup.com) for future newsletter discussion.

Interested in EOM training? We can either do custom training on-site, via WebEx or arrange for a formal class through Unisys. Please contact us for details.