

# newsletter



**PGCG**

Pretty Good Consulting Group

Got a problem? We create solutions.

**EOM News and Views**  
**Issue 11, April 2009**

## Secure fonts

Want to impress the boss with some fancy printing? Or how about make your checks more secure?

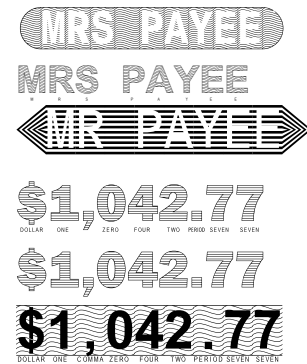
Printing checks usually makes people nervous. The composing and printing process is pretty simple, especially with EOM, but there are a few things to worry about (Are the check amounts correct? Is the MICR data line in the right place? Did we print the check properly? Can we account for all the check stock? And so on). Then, once the check is printed, we have to worry about possible ways the check could be corrupted.

Good check stock is made of specially treated paper that has multiple safeguards right in the paper: finely printed patterns that are too small for copiers to reproduce, watermarks that do not copy via a copy machine or are visible when copied, paper that turns color or smears or displays the word VOID if someone tries to change the check with chemicals, strands of special thread that can be seen with ultraviolet light, chemicals that make it hard to modify the printed MICR text, color schemes that are hard to copy, and other

safeguards.

Security built into the printing process includes the MICR line at the bottom of the check (bank routing code and account), a control number to uniquely identify that check, and playing with the text to minimize the ability of bad guys to change the data on the check (i.e. adding asterisks around the number so numbers are not added after the check is printer). There is another way to safeguard the printed data on a check: use special fonts for the amounts and even some of the text printed on the check.

For example, [Elfring Fonts](#) has a collection of fonts called "Secure Fonts" that provide multiple styles that make it harder to corrupt the text. A few examples:



Note that the fonts allow for text below each character if desired, as well as a few special characters that change the prefix or suffix:



## Contents

Secure Fonts.....	1
EOM 8.0 Feature .....	1
Tiff printing.....	2
Quick Hits.....	8
Contact information .....	8

## EOM 8.0 Feature

What the heck is the Transform Job used for? The new Transform Job manipulates the input file into a differently formatted output file AND allows actions to be taken with that new file. The input to the Transform Job can be XML or text, the output file can be a variety of formats depending on what you want the Transform Job to do. We will investigate just one use of the Transform Job here, specifically how to take an input XML file, marry it with a stylesheet, and produce an HTML email. For this example, pretend that the real problem you are trying to solve requires that an HTML email be sent to a user to confirm their recently submitted phone numbers on the National Do Not Call list ([www.donotcall.gov](http://www.donotcall.gov)).

The first thing to get straight is the data flow:

Data -> File Mask -> Transform Job -> XML Processing Directive -> Email Job -> HTML email sent

### Data

Lets start with the data. The formatted HTML email is a combination of the input data with the XML stylesheet (.XSL or .XSLT file). There can be formatting commands in either the input data or the stylesheet file or both. The example Demo1-Email.XML file released with EOM 8.0 has formatting data in the XML file:

(continued on page 3)

# How do I ...?

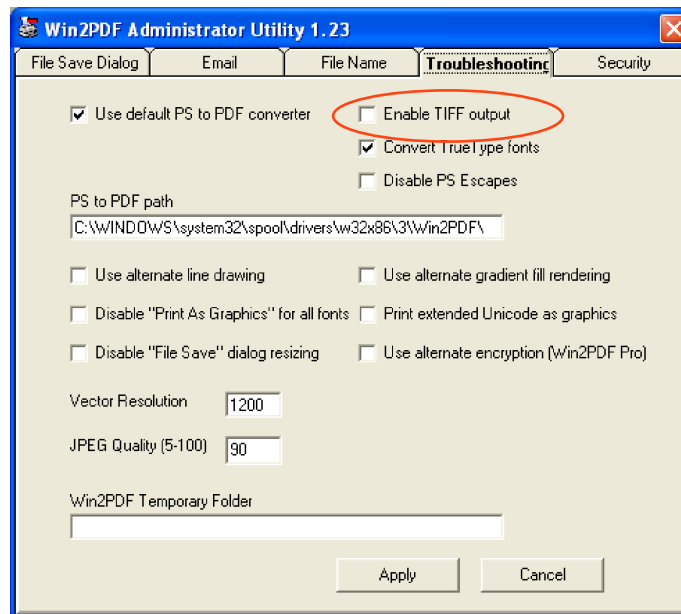
This section of the newsletter will discuss solutions to questions that come from real customers trying to solve real problems. Is there any way for EOM to produce TIFF files?

We have had the request to print TIFF files pop up every once in a while over the years. The typical answer is that yes you can have EOM create TIFF files as long as you find a Windows Print driver that creates TIFF files and that driver supports the programmatic interface to name the file.

Another solution for TIFF output was right under our collective noses, found by the EOM Development team. It turns out that the Dane Prairie Win2PDF solution for printing PDF files from EOM also has the capability to produce TIFF files. The catch is that you have to download a couple of additional files and set ALL Win2PDF output to produce TIFF files.

Installation and configuration are straightforward:

- Go to the [Dane Prairie download page](#), to download both the Win2PDF Font Helper application and the Win2PDF Admin Utility.
- Install both of these downloads, Font Helper first, on the same PC/server that the Win2PDF software resides.
- Run the Win2PDF Admin Utility
  - Select the "Troubleshooting" tab
  - Check the option labeled "Enable TIFF output".



There are a few downsides to this solution:

- ALL files run through the Win2PDF printer will be turned in to TIFF files.
- It will produce only one page in the TIFF output.
- The output is not as crisp as PDF, for example characters are a little fuzzy.
- Color graphics are turned to black & white.

So, if you already have Win2PDF, need single-page TIFF files, and can switch between PDF/TIFF manually then Win2PDF is a workable solution.

# EOM 8.0 Feature *(continued from page 1)*

```
Demo1-Email.xml
<?xml version="1.0" encoding="utf-16"?>
<Demo1-Email>
  <To>SteveD@PrettyGoodConsultingGroup.com</To>
  <Subject>Demo1 from XML, converted to PDF, and sent via e-mail</Subject>
  <Body>
    <p>Dear Enterprise Output Manager enthusiast,</p>
    <p>
      This is an example of what can be done with XML in Enterprise Output Manager
    </p>
    <p>
      The body is simply HTML text extracted from the original XML file.
      You can provide HTML markup in your data to have <b>
        bold and/or <u>underlined</u> text</b>
        and even <span style='color:red'>color</span>.
    </p>
    ...
  </Body>
  ...

```

```
SelectEmailBodyText.xsl
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:fo="http://www.w3.org/1999/XSL/Format"
  xmlns:svg="http://www.w3.org/2000/svg"
  version="2.0">
  <xsl:output method="html"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>Generating HTML from XML</title>
      <body>
        <xsl:copy-of select="//Body"/>
      </body>
    </head>
  </html>
</xsl:template>
</xsl:stylesheet>

```

The corresponding XSL file (SelectEmailBodyText.XSL) plucks that data from the input data file, specifically the text within the "body" node:

In our example, the input file will contain only the variable data and we will rely on the stylesheet to handle the formatting. The input file XML file looks like:

```
<?xml version="1.0" encoding="utf-16"?>
<Register-Email>
  <To>SteveD@PrettyGoodConsultingGroup.com</To>
  <Subject>National Do Not Call Registry - OPEN AND CLICK on Link to Complete Your Registration</Subject>
  <Body>
    <ReplyLink>https://www.donotcall/register/RegConf.aspx?BBC3C4BC-F8BD-491B-8B61-1EE180858B8C</ReplyLink>
    <LastFourDigits>1234</LastFourDigits>
  </Body>
</Register-Email>

```

The stylesheet (XSLT) file is a little more complicated because we include the formatting commands as well as references of where the variable data (ReplyLink, LastFourDigits) is to be inserted. The stylesheet is also more complicated because we used a GUI program to design the stylesheet (more on that below). The really important syntax in the stylesheet is:

```
<xsl:value-of select="Register-Email/Body/ReplyLink"/>
<xsl:value-of select="Register-Email/Body/LastFourDigits"/>

```

This is where the input data from the input XML file, node ReplyLink and node LastFourDigits is merged into the output stream. Note that the value of the To and Subject nodes are not referenced in the XSL file because that data is used by the EOM Transform Job directly when emailing the file. As you might guess, there can be many nodes of data in the input XML file can then be referenced in the resultant HTML email. Once processed via the Transform Job, the result is an email that looks like the graphic on the bottom of page 6.

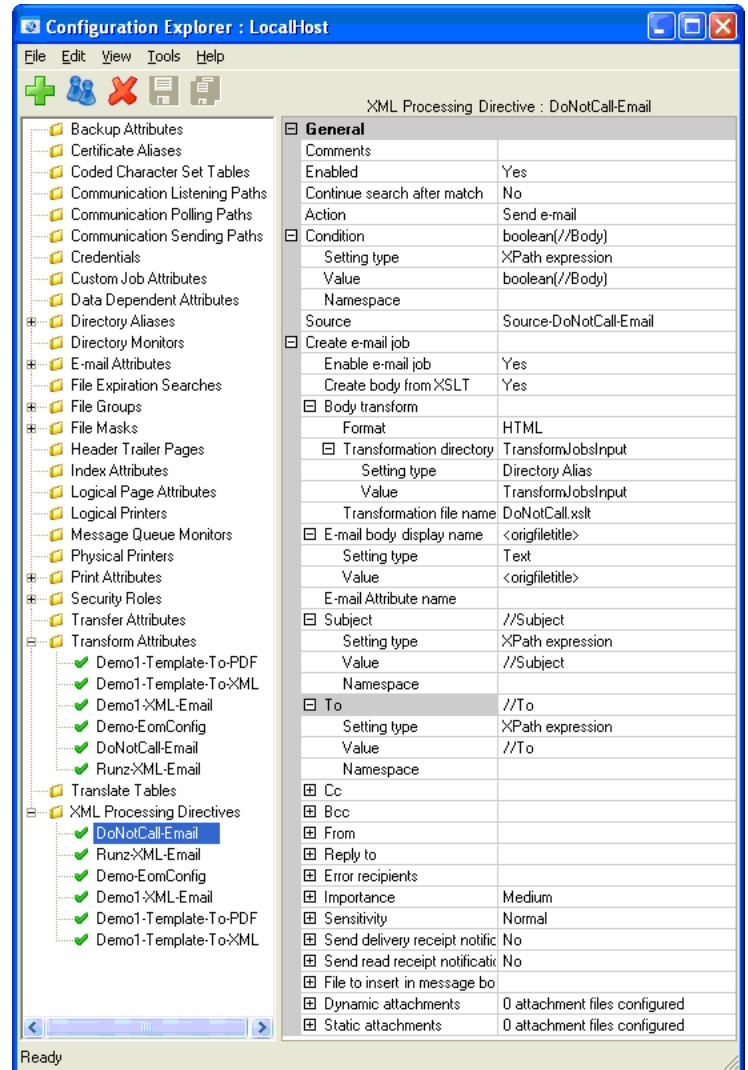
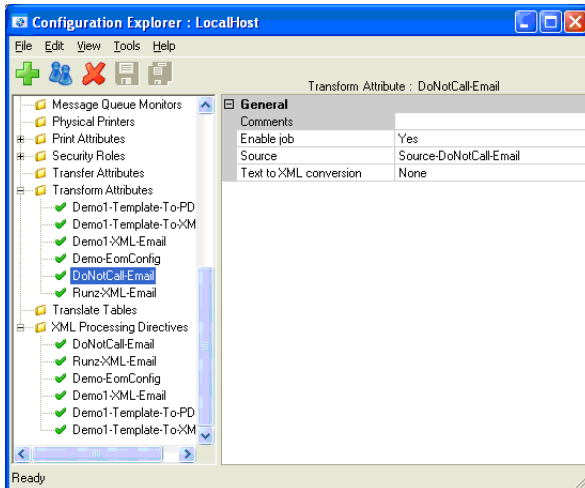
(continued on page 4)

## File Mask

This simple example can be made even simpler if we assume that an application creates individual XML files for each email to be sent. Maybe it is a web application that fires off as soon as the customer submits the phone number on a web page, or maybe the application looks into a database every 15 minutes for new entries to be handled. No matter how the XML file is generated, if it is written into a directory that the EOM Directory Monitor is "watching" then getting the file into the EOM environment is done. The File Mask is configured to submit a Transform Job to process the file. Note - you could use the new HTTP interface to feed the data to EOM from the web application, but that is discussion for another newsletter.

## Transform Attribute and XML Directive(s)

The Transform Attribute configuration is really simple. You give the attribute a name and then define a "Source", which is really a string that is used for matching one or more of the new XML Processing Directives. Much like the behavior of the File Mask, the Source string is used as a match conditional starting at the top of the list of XML Processing Directives and continues through the list until there is a match. You can configure XML Processing Directives to Continue Search After Match. Once the Source string matches the XML Processing Directives Source string, the Condition property is evaluated and, if true, the XML Directive is executed.



The XML Processing Directive is not quite as simple. Note that the Condition of the XML Directive is simply to verify that there is a //Body section within the input file. We need to create an email and want to use our XSLT file to create the body of the email message. Notice that the To and Subject fields are automatically defined from the input file data via XPath statements. If this email was always sent to the person or distribution list then the Directive could have that email address defined directly so it would not be required in the XML input file. The remaining email properties are identical to the Email Attribute properties.

## Using a GUI to design the stylesheet

Why design the XSL stylesheet with a GUI program? Very simply, creating an XSL file from scratch requires significant intestinal fortitude and a deep knowledge of the XSL syntax ("not for the faint of heart" was one description we heard). You can certainly compose the stylesheet from scratch using Notepad or other text-based editor, but using a GUI to "draw"

the expected result should be much easier. You also may find Notepad less than satisfactory if the XML and/or XSL file do not have records ending in carriage-return linefeed - your first indication of trouble will be that the records wrap throughout the displayed area. There are a handful of editors that present the XML/XSL better, one is the Microsoft Visual Studio Tools for Applications, another is the Altova XMLSpy. One product that provides a GUI interface to create XSL stylesheets is [Altova's Stylevision](#) product.

(continued on page 5)

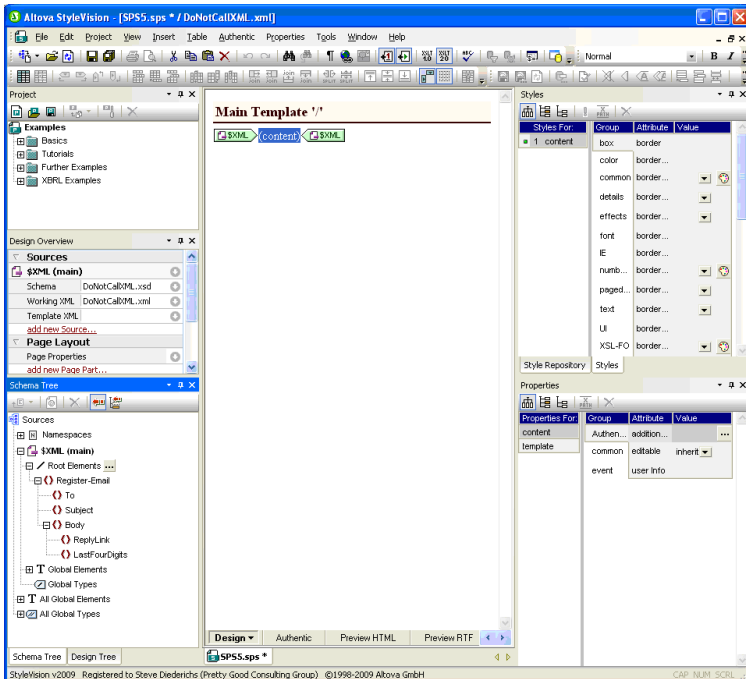
The Altova Stylevision program provides a drag & drop design environment where data references from an XML file (or database) can be inserted into the stylesheet. Stylevision has numerous other features, we will stick to the simple example here, where it is used to create the format of the HTML email.

The steps to create a stylesheet using Altova's Stylevision are:

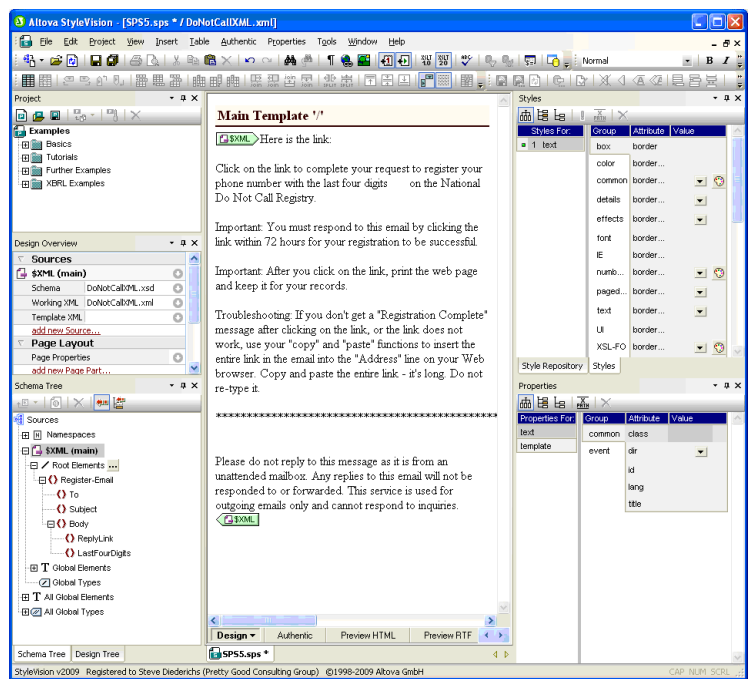
- 1) Select New / New from XML Schema/DTD/XML...
- 2) Browse to your example XML file
- 3) If you do not already have an XML schema, you will be required to select "Generate new XML Schema" and provide a file name
- 4) Create your stylesheet
  - A) Add the static text, graphics, hyperlinks, and whatever to the page
  - B) Add dynamic Bookmarks for the dynamic data (from the XML file). The "Insert", "Insert Bookmark" menu selection allows you to select the Dynamic node right out of the example XML file
  - C) Add dynamic Hyperlinks for the dynamic hyperlink data (from the XML file). The "Insert", "Insert Hyperlink" menu selection allows you to select the Dynamic node right out of the example XML file
  - D) Save the design in the Altova native format (.sps)
  - E) Go to "File" / "Save Generated Files" and select "Save Generated XSLT-HTML File..."
  - F) Edit the resultant .XSLT file. This step is messy. Editing is much easier if you use an XML/XSL editor like Microsoft Visual Studio Tools for Applications.
    - 1) Replace the string "<title>" with "<title>a</title>" (found after first <html> <head> sequence)
    - 2) Remove the <xsl:attribute name="name"> and <xsl:attribute> around each of the dynamic Bookmarks you added.

One of the major advantages of using the GUI design software is that you can quickly modify the look and feel of the stylesheet, including fonts, colors, images, and so on.

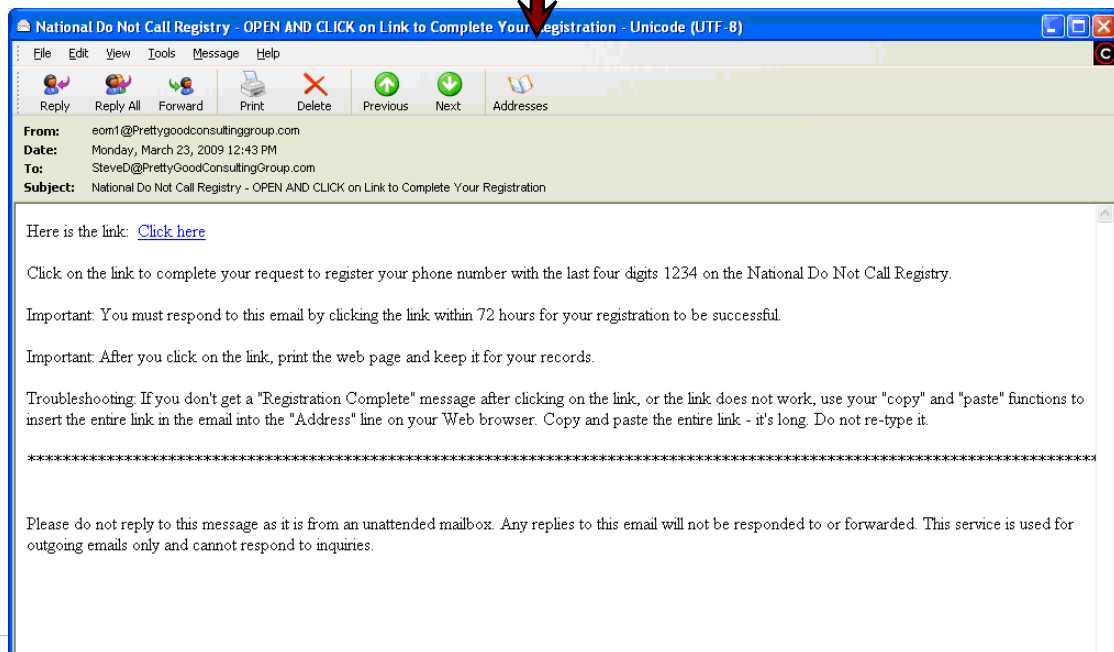
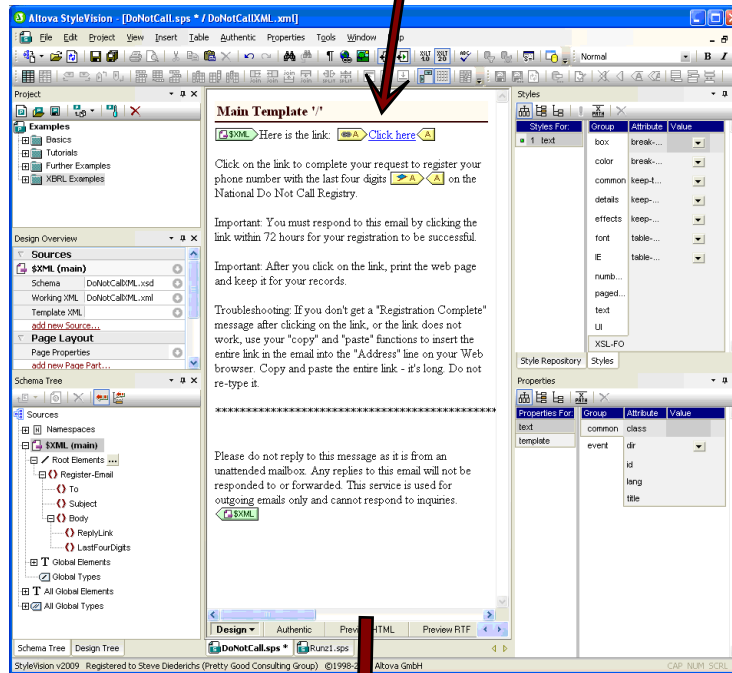
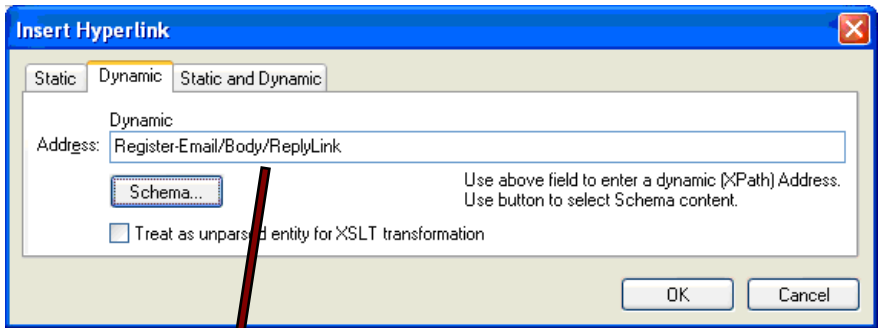
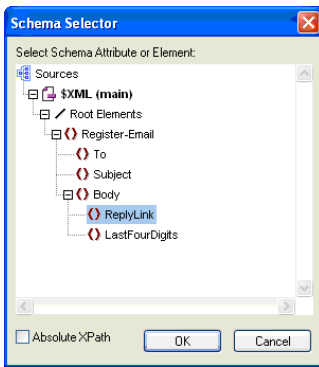
#### Step 1



#### Step 4a



Step 4 B & C



Then, from here you can spruce up the formatting without changing the input data.

The screenshot shows the Altova StyleVision interface. The main window displays a template titled "Main Template '1'". At the top, there is a "No Cell Phone" icon (a yellow smiley face with a red circle and slash over it). Below the icon, the text reads: "Here is the link: [Click here](#)". This is followed by instructions: "Click on the link to complete your request to register your phone number with the last four digits on the National Do Not Call Registry." There are two "Important" sections: one stating a 72-hour response time and another about printing the page. A "Troubleshooting" section explains what to do if the link doesn't work. At the bottom, there is a disclaimer: "Please do not reply to this message as it is from an unattended mailbox. Any replies to this email will not be responded to or forwarded. This service is used for outgoing emails only and cannot respond to inquiries."



The screenshot shows an email client window titled "National Do Not Call Registry - OPEN AND CLICK on Link to Complete Your Registration - Unicode (UTF-8)". The email content is the rendered version of the template. It features the "No Cell Phone" icon, the link "Click here", and the same instructions and troubleshooting information as the template. The email header shows it was sent from eom1@Prettygoodconsultinggroup.com on Monday, March 23, 2009 at 1:00 PM.

# Time to Upgrade?

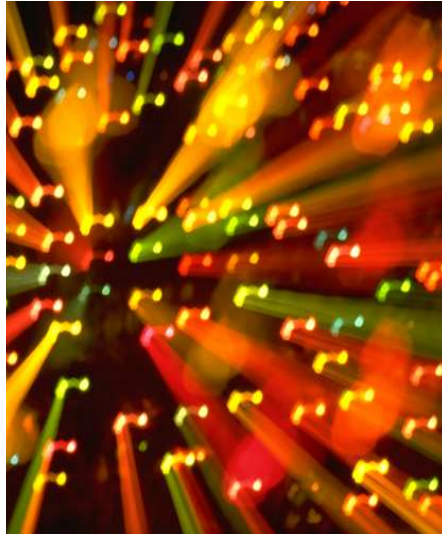
We continue to run into a lot of EOM 6.1.2, 6.1.3, and even a 6.1 sites. Helping customers with these versions is not nearly as easy as with EOM 8.0, nor is it as easy for the customer to maintain. Keep in mind that EOM versions 6.1 and below are no longer supported. EOM 8.0 is a new release with enterprise-class features to automate, re-engineer, and control your output environment.

## 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.



## Quick Hits

In the February newsletter we discussed the Alert Service configuration. In an example we stated that we wanted to know when the Legacy Assistant stopped and then proceeded to explain how to configure the Alert Policy to do that. Only one problem - Astute EOM user Dave McCann of Lumbermans Merchandising points out that the alert pops only after a Custom Job attempts to use the Legacy Assistant. He also reports that EOM Development recommends that you may also want to configure to run the email script if the EmailJobConfiguration alert occurs so either type of job will notify you when the Legacy Assistant is not running.

---

Unisys Customer Education has scheduled a web-based EOM Basic Workshop for April 21st -> April 24th. The class is held from 11am to 3pm (EST). Contact Unisys for details.

---

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.

---

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

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