

Case Study: Oxinet Mailshot

1 Project Aims

In September 2005 it was decided that a new mailshot programme was required that would take over the tasks performed by the various mailshot utilities currently used by Oxford Internet Consultants, and would be commercially viable product as well.

The Oxinet Mailshot project needed to have the following characteristics:

- Send large quantities of e-mail reliably without overwhelming the hosting server
- Flexibility in job setup, and job sending times and frequencies
- Advanced emailing features such as:
 - Content embedding (attachments and images)
 - Multipart content (send plain text and HTML in parallel)
 - Catching bounced emails
- Easy setup for the application and the jobs that will run
- Based on Win32 Architecture to run on existing servers

To achieve these aims it was decided that the application would comprise two parts. Namely a windows service and a windows form application. The windows service would send the emails. The windows form application would act as a configuration editor and as an interface for controlling, and viewing the activity of, the windows service.

The development would be in Visual Basic.NET (DotNet v1.1)

2 The Development Process

The design is centred around the `.config` file. This is an XML file that specifies all the information that a job requires to run. It is loaded by the Job Configuration component which is shared by the service and the form application, hence allowing significant code reuse and a cut down development time. One `.config` file is made for each type of job created in the configuration editor and can be copied between servers running the Oxinet Mailshot.

There is a long list of features provided by the Oxinet Mailshot which is available at <http://www.oxi.net/oxinetmailshot/features>

Multi-threaded programming was used to allow the service to simultaneously run threads to check for new jobs, run existing jobs and send emails. In this way if any part of the system encounters a hold up, it has a smaller effect on any other work happening at that time. Examples of this could be a message

being sent to an SMTP server that has to wait for a response across a network. In a single-threaded programme this would cause the whole programme to wait until a response was received. In the Oxinet Mailshot the programme will continue to check for new jobs, and send up to nine other emails while this is happening.

Testing has been performed on servers running both XP™ and 2003 Server™ versions of Microsoft® Windows™.

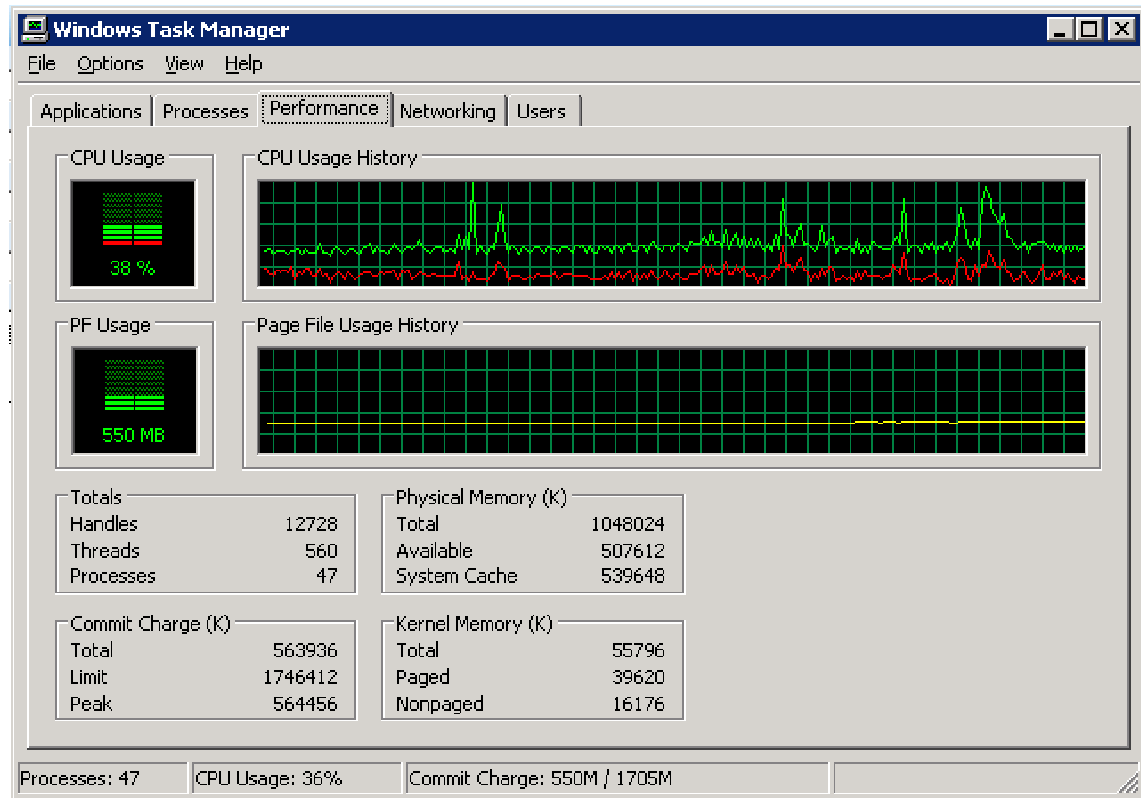


Figure 2-1: Oxinet Mailshot Job running on a Celeron 1.4GHz with Windows® 2003 Server™

Several issues were encountered during the development of the service:

2.1 TcpClient

It was noted during a test run that although the TcpClient DotNet component connected in a very short time when connecting to a local port, the time taken to connect to a remote port was far longer. Although it is expected that it would be a marginally longer time, the difference was large enough that a third of a second was being used each time a connection was made.

Searching the internet provided the answer. With thanks to Richard (see footnote¹). The DotNet implementation of the TcpClient object has two

¹ Stronglytyped.com message boards - <http://www.stronglytyped.com/v1/2003/07/google-search-tcpclient-slow.html>



constructors. One uses two strings as the IP and host, the other uses an IPEndPoint. Swapping the Oxinet Mailshot service to the second option from the first caused a significant speed increase from 0.3 emails per second to 1.8 emails per second and a drop in processor load on the testing server of thirty to forty percent².

2.2 Threading issues

The mailshot service ran on Windows XP™ without problem, but would exit unexpectedly on Windows 2003 Server™. This was seems to be due to Windows 2003 Server™ not recognising a timer as a valid reason to keep a thread running; where XP would allow the thread to continue, 2003 Server™ would cause the thread to abort. This was shutting down the job before the mails had completed sending and causing a race condition that made the symptoms hard to diagnose. A minor modification to the code has solved this issue and provided a slight performance increase.

2.3 Outlook Express Content Embedding

Although not the recommended method for marking embedded images in emails, Oxinet recognise the need to make applications work in the real world, as well as making them RFC compliant. For this reason, the Oxinet Mailshot service supports image embedding that will work both in Microsoft® Outlook Express™, and in open source mail clients such as Thunderbird. A side-effect of this is that images linked in emails and sent with embedded images by the mailshot service will use the embedded images if available and if not will still work as links to the outside world. This provides better robustness and means that wherever possible the end-user will see the images.

3 Conclusions

The Oxinet Mailshot is a robust, flexible application capable of sending large quantities of emails quickly and efficiently. It provides excellent feedback to let you know what has happened, and now also supports sending from multiple locations or multiple IP addresses in a single location.

Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

² On a Pentium 4(HT) 3.2GHz processor running windows XP sending a batch of 27,000 emails at 40KB each.