soyMAIL - Son of yahMAIL

by Mark Daniel

For those looking for an alternative e-mail solution, away from the likes of MS Exchange, then soyMAIL may be what you're looking for. Mark Daniel, the Author, gives us an insight into its inner workings.

The working title of this David Lean -esque epic was for some time 'yahMAIL II'. Presumably any third version of web-mail arising from the fevered imagination of the author will have to be titled 'yahMAIL Redux – the Director's cut'. However, to inject some interest into an otherwise fairly dry activity, after starting and restarting a second cut of web-mail for VMS, I found myself needing one of several sanity breaks, and something to smile about during the wee, small hours. From this the handle 'soyMAIL' originated.

soyMAIL is not at all yahMAIL version 2. This is one reason for a completely fresh name. I'll explain why. For those few (in my hubris) who are not familiar with yahMAIL, it is a (now obsolescent) CGI-based, web-mail application for authenticated access to a user's VMS Mail.

The name yahMAIL was derived from 'yet another hypermail'. yahMAIL is supported on Apache (CSWS 1.n, 2.n), OSU and WASD web servers - and even Purveyor Bob, and running on any VMS platform (Alpha, Itanium and VAX), from VMS V6.0 up to the latest.

It is an embarrassingly large and single bowl of spaghetti, that began as a simple recipe for securely accessing (using SSL) my external VMS Mail through firewalls and proxy servers that would not support the more conventional protocols such as POP and IMAP. Of course once the basics were in place, additional functionality became necessary as increasingly I wanted to use it everyday. MIME was the main requirement. Then a few people who knew of it suggested it might be generally useful. It escaped as freeware and then demanded internationalisation. Oh boy! My bowl of spaghetti now had meatballs, a rich sauce and grated cheese - and still didn't *look* very appetising. A recipe for indigestion, at least for the cook!

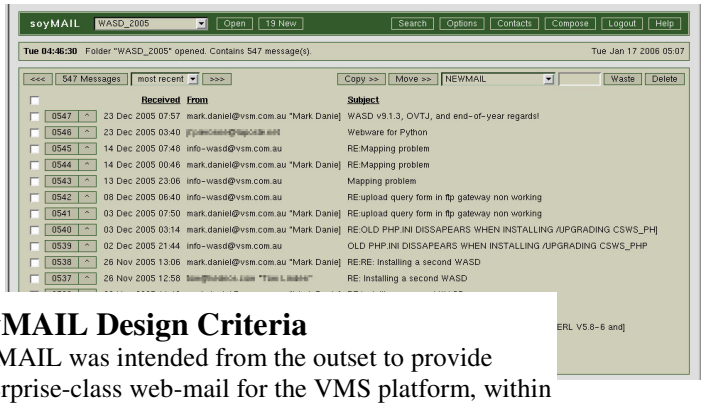
Hopefully I'm now done with the mixed metaphors.

soyMAIL has been designed completely from scratch, with no code and few concepts, reused from yahMAIL. That said, there has been significant experience gained with VMS Mail, web applications and MIME that has been employed in implementing soyMAIL.

yahMAIL was widely accepted and deployed as a useful VMS Mail utility. This was in spite of it's woeful look-and-feel (well it was before the days of general CSS support) and kludgy MIME support. yahMAIL had it's public BETA mid 1999 and initial release towards the end of 1999.

Over the next couple of years additional functionality was grafted in. So good was the feedback on having an application like this for VMS that even then I decided that it warranted a successor. Active development of yahMAIL ceased in late 2003.

Development of that successor started a couple of times in 2004. What eventually became soyMAIL commenced development in early 2005 and was BETA released at the start of 2006 (yes, it took a while!).



soyMAIL Design Criteria

soyMAIL was intended from the outset to provide enterprise-class web-mail for the VMS platform, within the practical constraints imposed by a single developer working in spare time. I aimed to cover a number of key criteria missing from yahMAIL :-

- Performance -- soyMAIL needed to provide low-latency access to a users Mail. Especially for MIME encoded messages with large attachments

- Look-and-feel -- the nineties browser buttons, chunky and clunky, must be avoided. Near universal adoption of Cascading Style Sheets (CSS) by browser platforms in the years immediately following yahMAIL meant the web application developer now had significant control of the display elements of those applications

- MIME -- could no longer be a kludge. It must be an integral component of the message processing. In fact native VMS Mail messages are very much the poor relation (oops, another metaphor). soyMAIL treats all messages as RFC[2]822 and RFC2045 (et al.) compliant, and makes native VMS Mail messages look that way during it's internal processing

- HTML awareness -- much as I'm not an advocate of HTML message content it is a fact of life and was one of the most frequently requested additions to yahMAIL. soyMAIL can display and create HTML message bodies. This uses an optional HTML editor, driven by JavaScript

- Features -- bells and whistles expected in useful applications. Things such as message search, message drafts, contact (email address) management, and context-sensitive help

- Internationalisation -- VMS is not exclusively English speaking or even ISO Latin-1 rendered. Neither should soyMAIL be so hard-wired

- Universal deployment -- in line with the (WASD) objective of everywhere for everyone VMS, soyMAIL should be available for any VMS web server on any VMS platform for all the major browser clients irrespective of platform

- JavaScript -- always a contentious issue. It indisputably adds useful dynamics to the client-end. Some hate it with a passion (anyone plagued by pop-up smiley advertisements can't but have an element of sympathy for this position). All fundamental email and messaging functions can be used without JavaScript, however soyMAIL is much more functional with JavaScript enabled!

There have been some compromises required. Not least of all those related to development time available. Some of these compromises will undoubtedly be addressed during future development.

soyMAIL Under-The-Hood

As 'Ping' is largely a technical journal this section concentrates on some of the more interesting (at least to me) aspects of the implementation, rather than be a blog of the dry slog of coding for page margins and HTML form processing.

soyMAIL provides access to a VMS users VMS Mail repository via the VMS callable Mail interface. It does not require or use POP or IMAP servers. There are obvious efficiencies in avoiding these, particularly some of the IMAP implementations (by reputation).

Standard CGI was the obvious implementation environment being the lowest-common-denominator for widespread deployment. With the CGI paradigm soyMAIL is not persistent and so incurs some overhead with it's one full Mail access cycle - open-process-close - per request. Persistent implementations using WASD CGIplus or as an Apache module are possible and might ameliorate this a little but introduce variants impossible to manage in a time and enthusiasm constrained shop (mine). In practice soyMAIL seems more than responsive enough and of moderate impact on system performance.

Many VMS Mail messages are stored in files external to the primary MAIL.MAI ISAM file. Due to the size of most Internet mail format and MIME encoded messages, these inevitably end up in such external files. VMS callable mail transparently provides the content of both internal and externally stored messages. Unfortunately, the overhead when using the per-record interface of callable Mail (which in turn uses RMS) makes accessing many larger MIME-encoded messages quite expensive and highly latent.

soyMAIL avoids this by directly reading these external files as large opaque objects and then explicitly parsing the record content itself. This improves access time by some twenty fold, reducing the latency when accessing a 1MB attachment, from twenty seconds or so, down to less than one second, based on a DS20 test system.

To be as comprehensive as possible soyMAIL supports multiple mail repository files (those ending in .MAI but not beginning with MAIL$). To do this without unnecessarily multiplying soyMAIL-specific, external-to-Mail data files, the directory containing the primary MAIL.MAI needs to be searched. The search is for *.MAI and must then exclude MAIL$*.MAI.

In a directory with hundreds or even thousands of external message files this becomes non-trivial. Using $SEARCH on a directory with twenty thousand files, it would take some three minutes and put system EXEC mode through the roof.

Some relief could be gained by caching the results for a time but even a once-when-starting three minute latency was obviously unacceptable. The solution was to read the entire MAIL.MAI parent directory file into memory and explicitly parse the entries. This reduced the latency several hundred fold, down to a couple of seconds, and barely registers on a system utilization monitor.

Accessing Internet mail messages via VMS callable Mail is fine. Just ignore the VMS Mail header and treat all messages as RFC even if it requires a little creative programming.

As there is no standard MIME library available for VMS soyMAIL would need to implement all the MIME basics itself. Sending Internet mail messages, including MIME encoded ones, required more effort.

As with MIME decoding there is no library to allow encoding of content - soyMAIL must provide it's own. In addition, there are a number of Message Transport Agents (MTA) for VMS. TCP/IP Services own SMTP server, Process Software's PMDF, the freeware and commercial versions of MX, MultiNet, TCPware just for starters. Some of these provide their own API, others do not.

It was decided that to make soyMAIL as universal as possible, and without varying builds and interface maintenance, it would need to be able to format Internet style messages itself, including MIME-encoded content, and use the SMTP protocol to convey these messages directly to a site's SMTP messaging agent. *soyMAIL therefore requires a local SMTP relay* to be able to generate Internet style messages.

Though not a refinement expressly for VMS, soyMAIL undertakes an interesting (albeit small) experiment in Ajax (http://ajaxpatterns.org/). Ajax is just a term coined to describe the idea of web applications having a fine processing and display granularity, avoiding the entire page refresh paradigm we are all so familiar with. It is based on CSS, JavaScript and the W3C Document Object Model (DOM) technologies.

When residing in the NEWMAIL folder, soyMAIL can be instructed to automatically check the server for new messages. yahMAIL had a similar feature and merely used a JavaScript timer event to periodically refresh the page. soyMAIL however uses a very lightweight 'poll' of the server. This occurs in it's own thread of processing within the browser. Depending on the response from this poll the soyMAIL client either restarts the timer (no new messages) or refreshes the page (displaying the new messages). This reduces the usual request overhead from something like 2.5 Kbytes to more like 500 bytes and the response from 25 Kbytes (an 'average' soyMAIL message listing page - 5 Kbytes when GZIP transfer-encoded) to less than 300 bytes! Over the Internet this can reduce the latency for the new message check from several seconds to a single second, and to sub-seconds on an intranet.

From the desktop perspective there are only those annoying browser flickers in the background when there is actually something new to display. Further refinements using the Ajax approach may be possible but the design criterion requiring all major functionality be available without using JavaScript may limit it's applicability.

So There You Have It

A bit of history, design criteria and some (hopefully) interesting implementation anecdotes. soyMAIL is an elegant and functional web-mail application for VMS. All you need is a web server, the configuration instructions, and all that VMS-based Mail is instantly and securely accessible from anywhere on the globe. By the time you are reading this article all of the obvious bugs of the early BETA phase should be addressed and soyMAIL should be available as a stable application you can be proud to show off running on your VMS system.

soyMAIL is in BETA release and available from
http://wasd.vsm.com.au/wasd/

A soyMAIL overview, including screenshots can be found at
http://wasd.vsm.com.au/soymail/-/doc/soymail_overview.html