1 mod_perl Advocacy

1.1 Description

Having a hard time getting mod_perl into your organization? We have collected some arguments you can use to convince your boss why the organization wants mod_perl.

You can contact the mod_perl advocacy list if you have any more questions, or good arguments you have used (any success-stories are also welcome to the docs-dev list).

Also see Popular Perl Complaints and Myths.

1.2 Thoughts about scalability and flexibility

Your need for scalability and flexibility depends on what you need from your web site. If you only want a simple guest book or database gateway with no feature headroom, you can get away with any EASY_AND_FAST_TO_DEVELOP_TOOL (Exchange, MS IIS, Lotus Notes, etc).

Experience shows that you will soon want more functionality, at which point you'll discover the limitations of these "easy" tools. Gradually, your boss will ask for increasing functionality and at some point you'll realize that the tool lacks flexibility and/or scalability. Then your boss will either buy another EASY_AND_FAST_TO_DEVELOP_WITH_TOOLS and repeat the process (with different unforseen problems), or you'll start investing time in learning how to use a powerful, flexible tool to make the long-term development cycle easier.

If you and your company are serious about delivering flexible Internet functionality, do your homework. Then urge your boss to invest a little extra time and resources in choosing the right tool for the job. The extra quality and manageability of your site along with your ability to deliver new and improved functionality of high quality and in good time will prove the superiority of using solid flexible tools.

1.3 The boss, the developer and advocacy

Each developer has a boss who participates in the decision-making process. Remember that the boss considers input from sales people, developers, the media and associates before handing down large decisions. Of course, results count! A sales brochure makes very little impact compared to a working demonstration, and demonstrations of company-specific and developer-specific results count for a lot!

Personally, when I discovered mod_perl I did a lot of testing and coding at home and at work. Once I had a working heavy application, I came to my boss with two URLs - one for the plain CGI server and the other for the mod_perl-enabled server. It took about 30 secs for my boss to say: ‘Go with it’. Of course since then I have had to provide all the support for other developers, which is why I took time to learn it in first place (and why this guide was created!).

Chances are that if you've done your homework, learnt the tools and can deliver results, you'll have a successful project. If you convince your boss to try a tool that you don't know very well, your results may suffer. If your boss follows your development process closely and sees that your progress is much worse than expected, you might be told to "forget it" and mod_perl might not get a second chance.

Advocacy is a great thing for the open-source software movement, but it's best done quietly until you have confidence that you can show productivity. If you can demonstrate to your boss a heavy CGI which is running much faster under mod_perl, that may be a strong argument for further evaluation. Your company may even sponsor a portion of your learning process.

Learn the technology by working on sample projects. Learn how to support yourself and learn how to get support from the community; then advocate your ideas to your boss. Then you'll have the knowledge; your company will have the benefit; and mod_perl will have the reputation it deserves.

1.4 A summary of perl/CGI discussion at slashdot.org

Well, there was a nice discussion of merits of Perl in CGI world. I took the time to summarize this thread, so here is what I've got:

Perl Domination in CGI Programming? http://slashdot.org/askslashdot/99/10/20/1246241.shtml

- Perl is cool and fun to code with.
- Perl is very fast to develop with.
- Perl is even faster to develop with if you know what CPAN is. :)
- Math intensive code and other stuff which is faster in C/C++, can be plugged into Perl with XS/SWIG and may be used transparently by Perl programmers.
- Most CGI applications do text processing, at which Perl excels
- Forking and loading (unless the code is shared) of C/C++ CGI programs produces an overhead.
- Except for Intranets, bandwidth is usually a bigger bottleneck than Perl performance, although this might change in the future.
- For database driven applications, the database itself is a bottleneck. Lots of posts talk about latency vs throughput.
- mod_perl, FastCGI, Velocigen and PerlEx all give good performance gains over plain mod_cgi.
- Other light alternatives to Perl and its derivatives which have been mentioned: PHP, Python.
- There were almost no voices from users of M$ and similar technologies, I guess that's because they don't read http://slashdot.org :)
- Many said that in many people's minds: 'CGI' eq 'Perl'

1.5 Maintainers

Maintainer is the person(s) you should contact with updates, corrections and patches.

- Stas Bekman <stas (at) stason.org>

1.6 Authors

- Stas Bekman <stas (at) stason.org>

Only the major authors are listed above. For contributors see the Changes file.

Table of Contents: