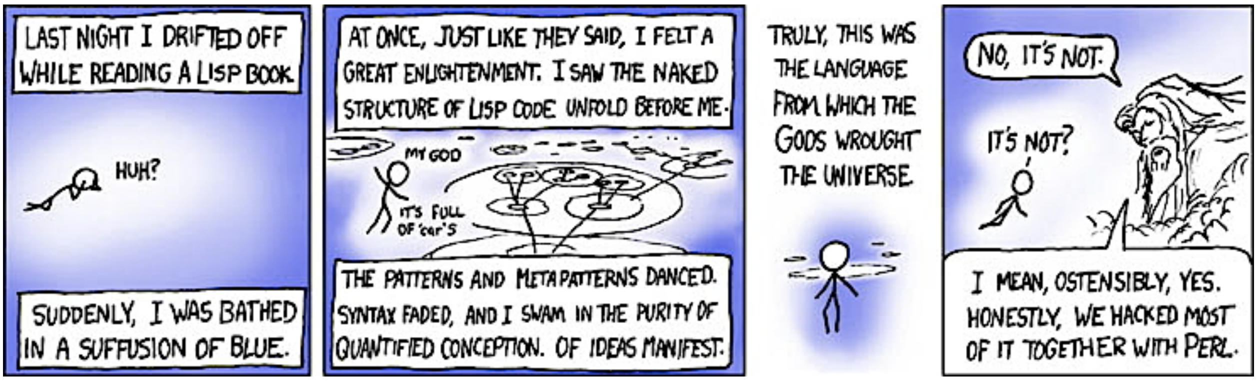
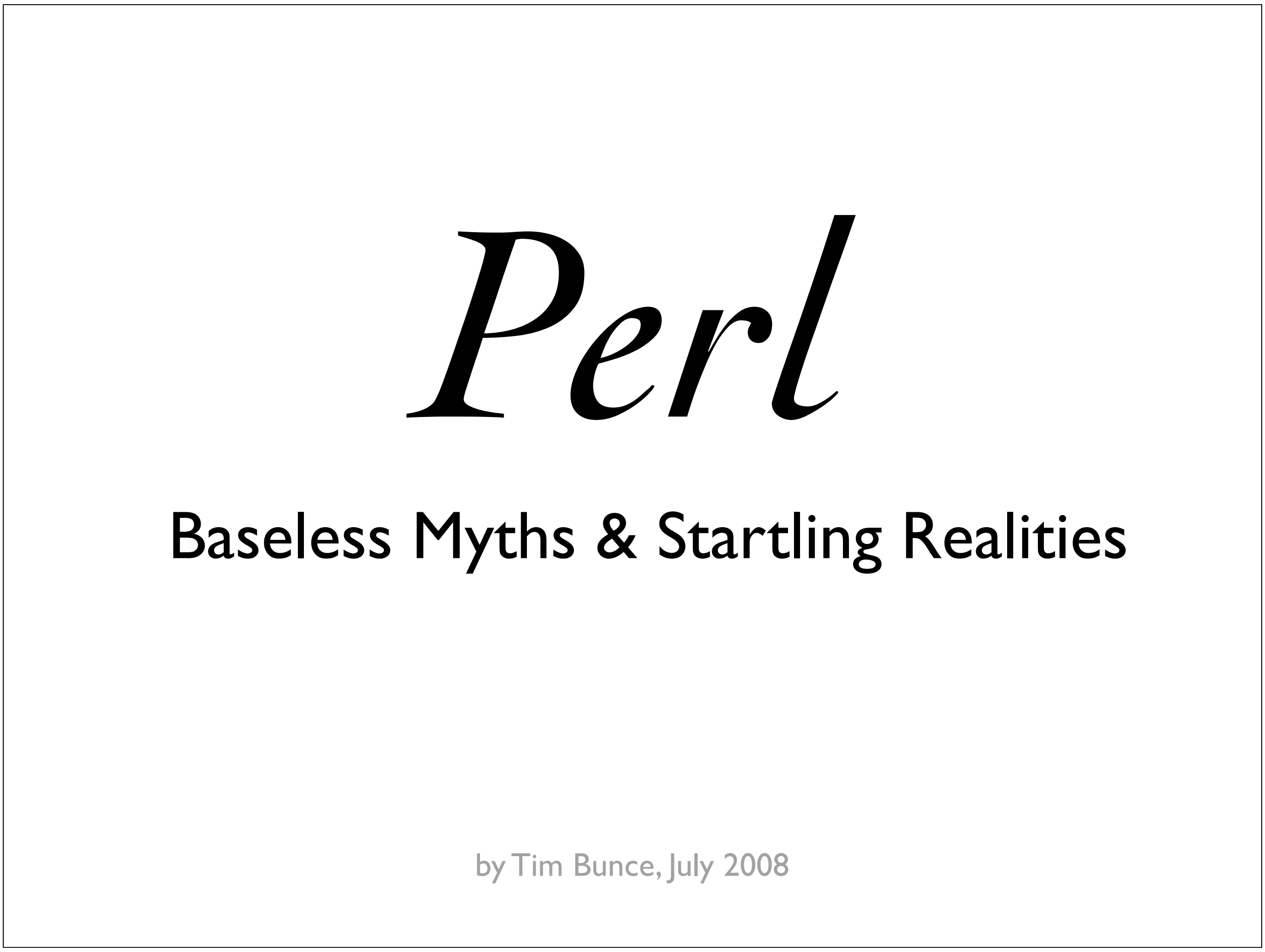

LAST NIGHT I DRIFTED OFF WHILE READING A LISP BOOK.
HUH?
SUDDENLY, I WAS BATHED IN A SUFFUSION OF BLUE.
AT ONCE, JUST LIKE THEY SAID, I FELT A GREAT ENLIGHTENMENT. I SAW THE NAKED STRUCTURE OF LISP CODE UNFOLD BEFORE ME.
MY GOD
IT'S FULL OF 'car'S
THE PATTERNS AND METAPATTERNS DANCED. SYNTAX FADED, AND I SWAM IN THE PURITY OF QUANTIFIED CONCEPTION. OF IDEAS MANIFEST.
TRULY, THIS WAS THE LANGUAGE FROM WHICH THE GODS WROUGHT THE UNIVERSE.
NO, IT'S NOT.
IT'S NOT?
I MEAN, OSTENSIBLY, YES. HONESTLY, WE HACKED MOST OF IT TOGETHER WITH PERL.

http://xkcd.com/224/

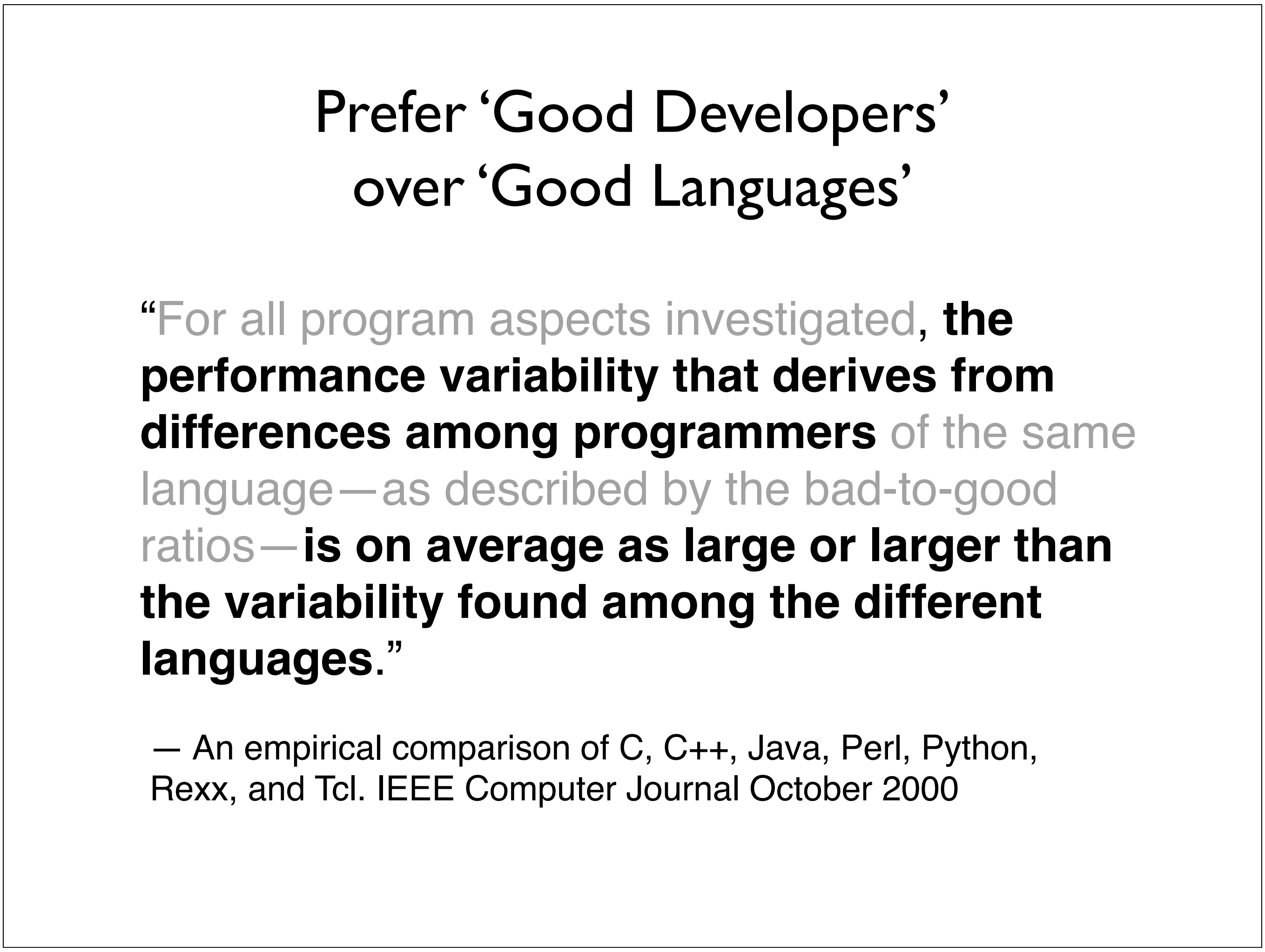


Perl

Baseless Myths & Startling Realities

by Tim Bunce, July 2008

Parrot and Perl 6 portion incomplete
due to lack of time
(not lack of myths!)



Prefer ‘Good Developers’ over ‘Good Languages’

“For all program aspects investigated, **the performance variability that derives from differences among programmers** of the same language—as described by the bad-to-good ratios—**is on average as large or larger than the variability found among the different languages**.”

— An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl. IEEE Computer Journal October 2000

http://www.cis.udel.edu/~silber/470STUFF/article.pdf

“In the script group, the Perl subjects may be more capable than the others, because the Perl language appears more than others to attract especially capable people.” :)

Who am I?

Tim Bunce

Author of the Perl DBI module

Using Perl since 1991

Involved in the development of Perl 5

“Pumpkin” for 5.4.x maintenance releases

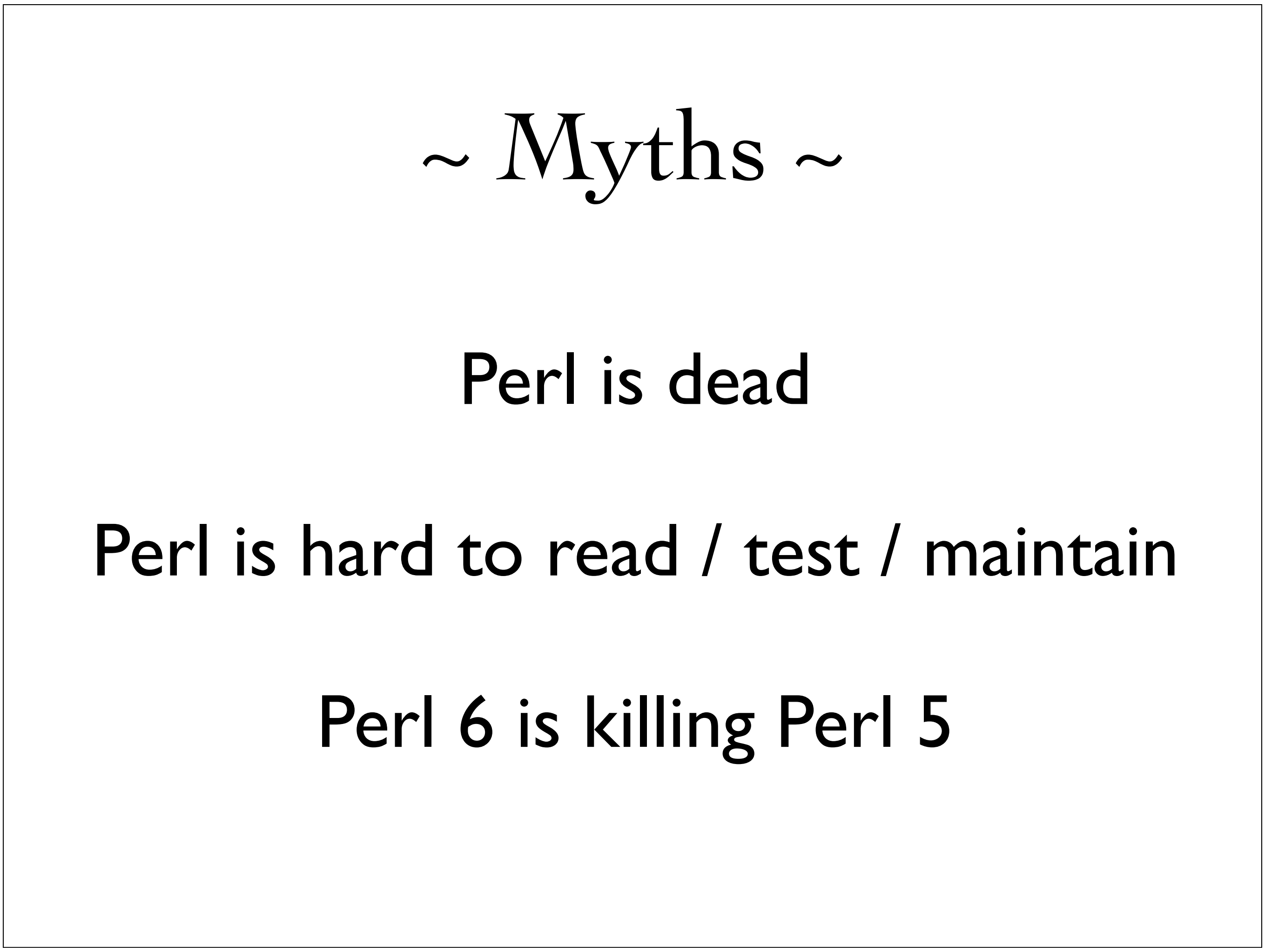
http://blog.timbunce.org

Perl 5.4.x 1997-1998

Living on the west coast of Ireland

~ Myths ~

http://www.bleaklow.com/blog/2003/08/new_perl_6_book_announced.html



~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

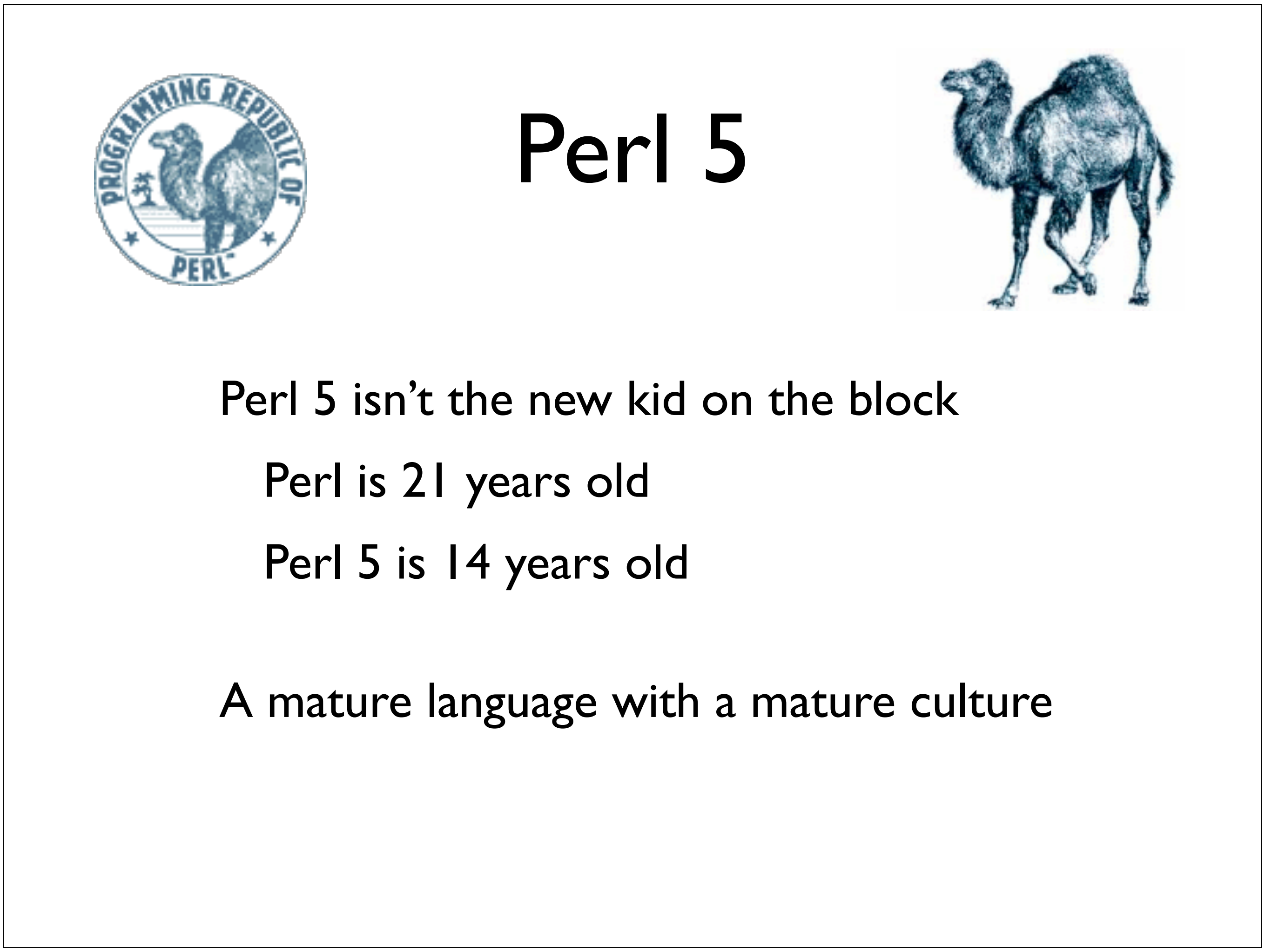
Another myth: Perl is slow:
    http://www.tbray.org/ongoing/When/200x/2007/10/30/WF-Results

~ Myths ~

Perl is dead

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5



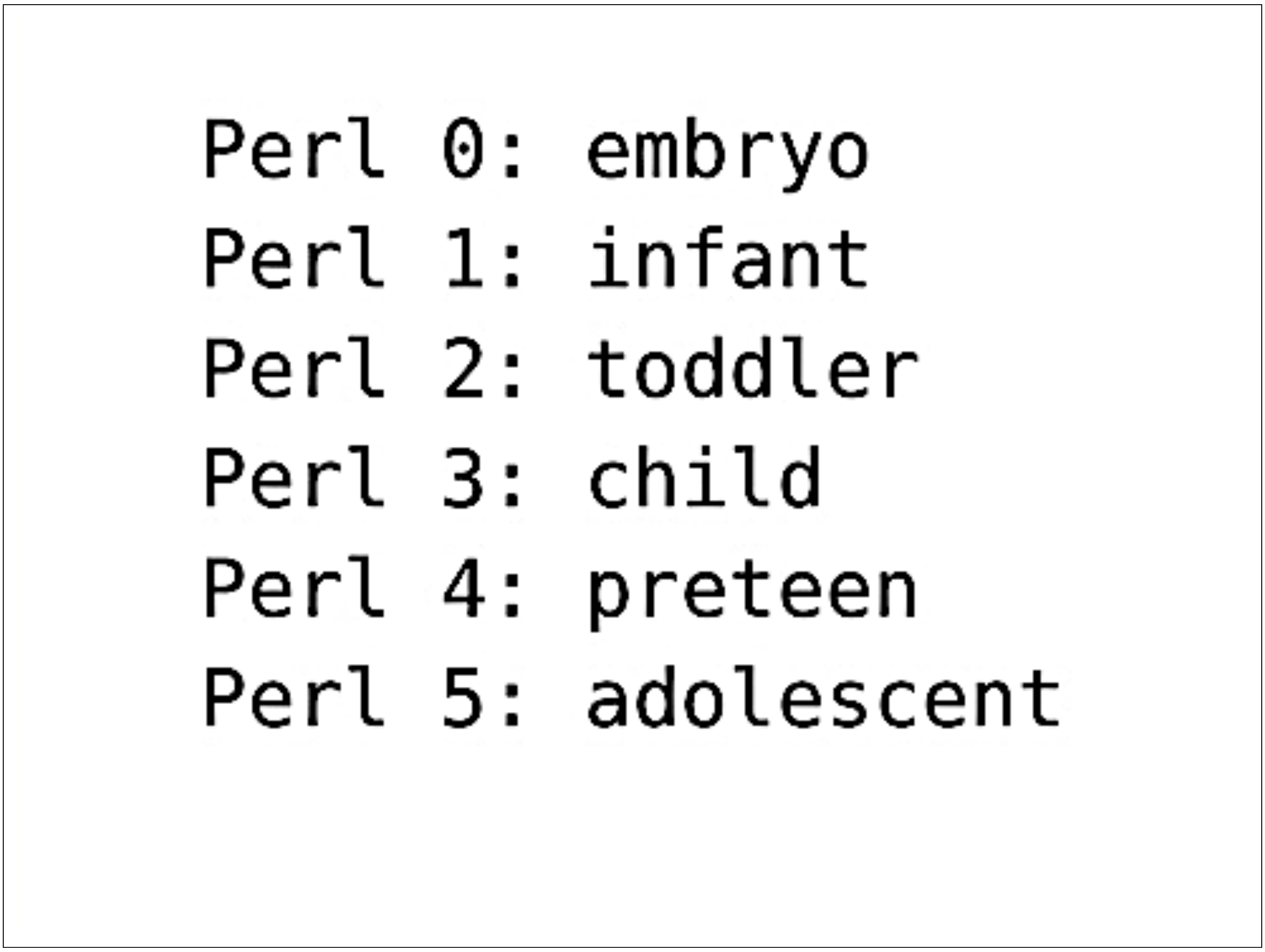
Perl 5

Perl 5 isn’t the new kid on the block

- Perl is 21 years old
- Perl 5 is 14 years old

A mature language with a mature culture

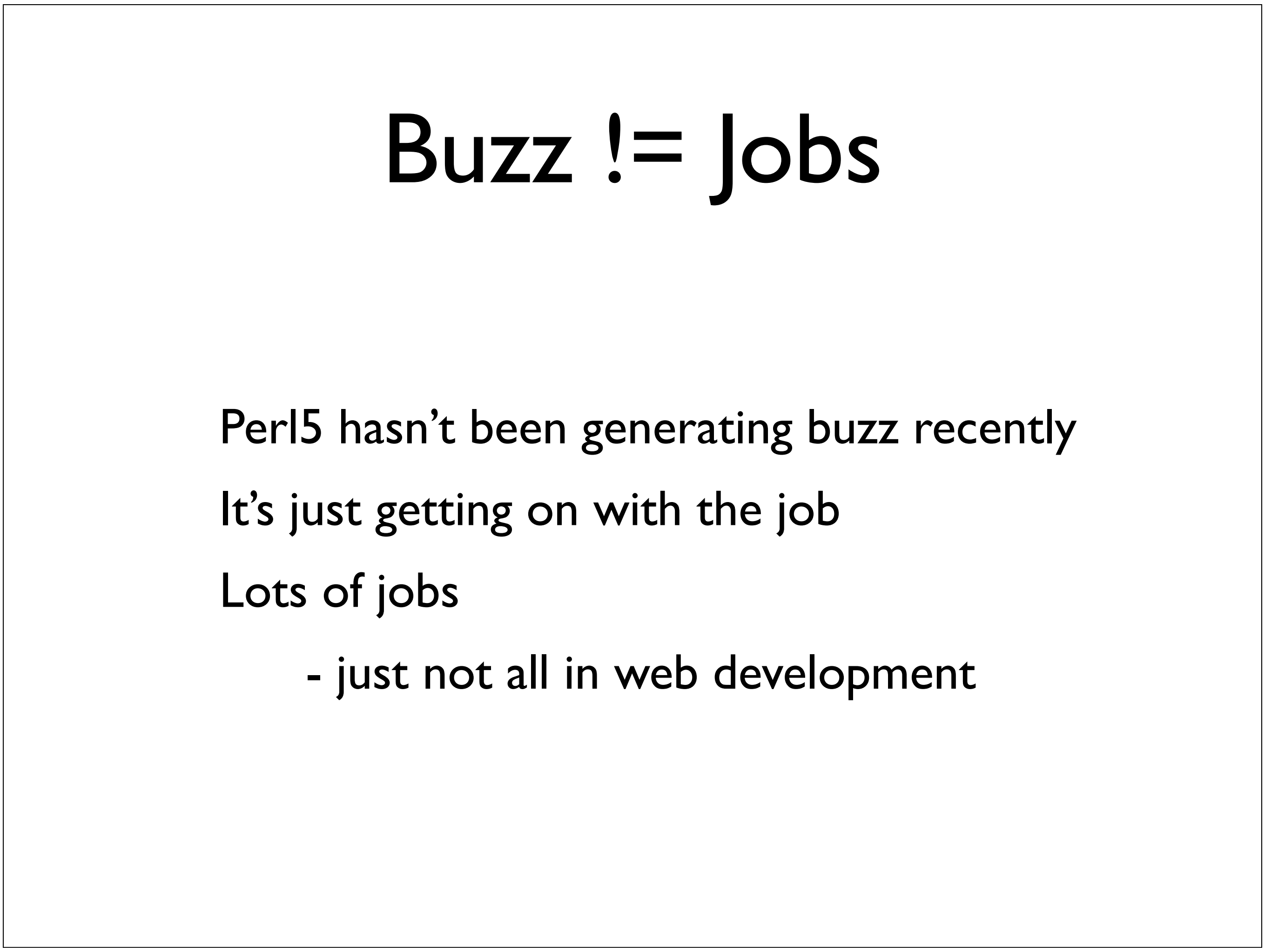
How many times Microsoft has changed developer technologies in the last 14 years...



```
Perl 0: embryo
Perl 1: infant
Perl 2: toddler
Perl 3: child
Perl 4: preteen
Perl 5: adolescent
```

You can guess where that’s leading...

From “The State of the Onion 10” by Larry Wall, 2006
http://www.perl.com/pub/a/2006/09/21/onion.html?page=3



Buzz != Jobs

Perl5 hasn’t been generating buzz recently

It’s just getting on with the job

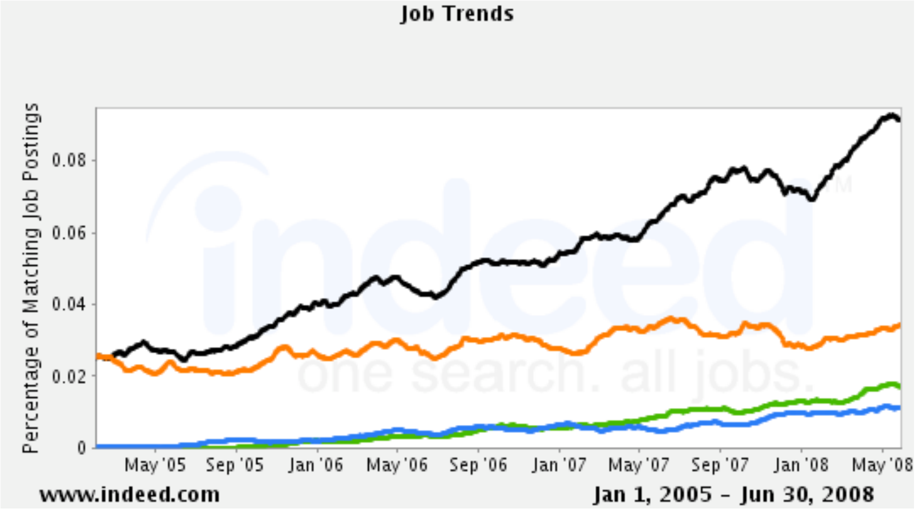
Lots of jobs

- just not all in web development

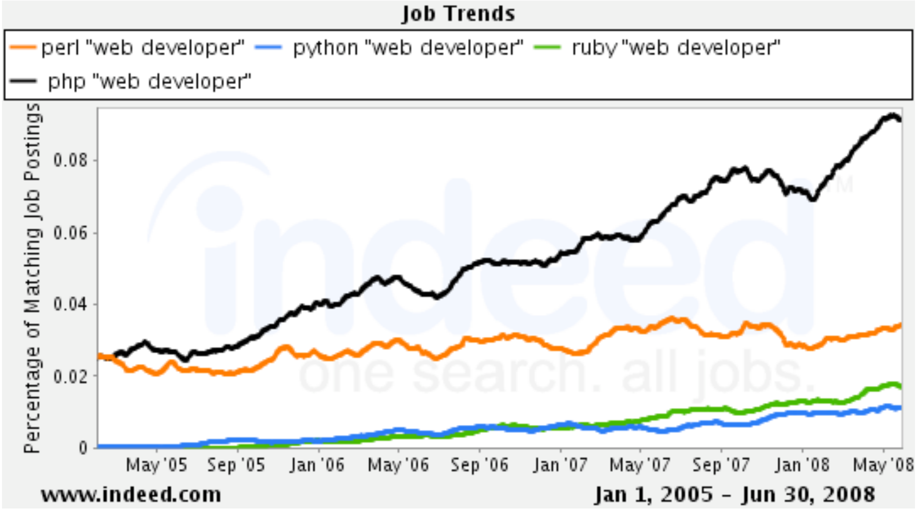
Web developers tend to have a narrow focus.

“At a recent finance technology conference in New York, the top 3 foundational technologies (by number of references) mentioned over the course of the conference by its speakers were:
#3 - XML
#2 - SQL
#1 - Perl
There were no others mentioned.” -- Richard Dice, president, The Perl Foundation,
in reference to O'Reilly's Money:Tech conference, New York, Feb 6 & 7, 2008.
By "foundational technology" I mean a building block technology.

Guess the Languages



“web developer”

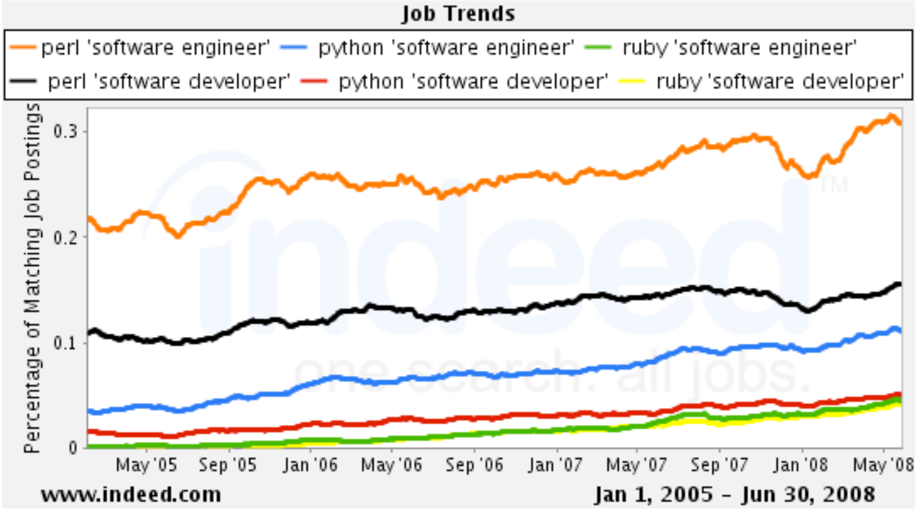


Yes, Perl is growing more slowly than others
but these are just “web developer” jobs

I think this graph captures the essence of why people think Perl is stagnant.
It’s because Perl hasn’t been growing much *in the ‘web developer’ world*

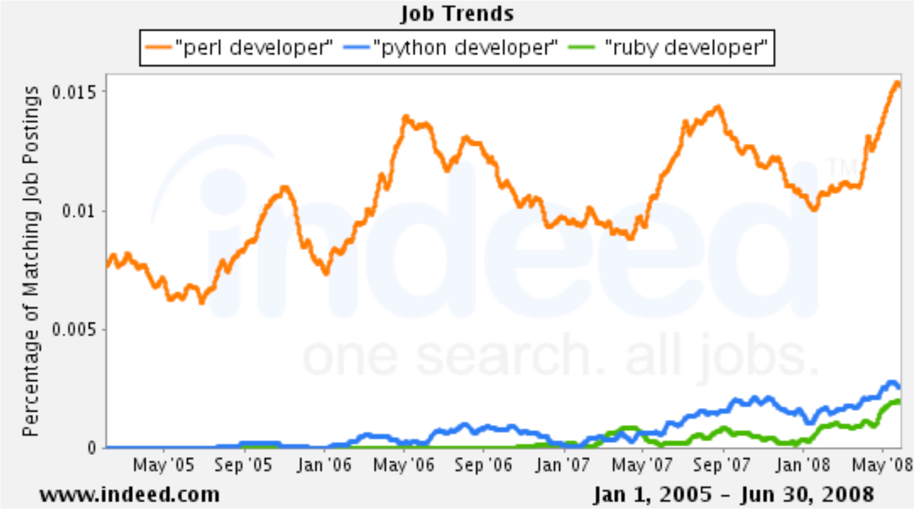
PHP - I’m not going to focus on PHP because it’s not a general purpose language.

“software engineer”



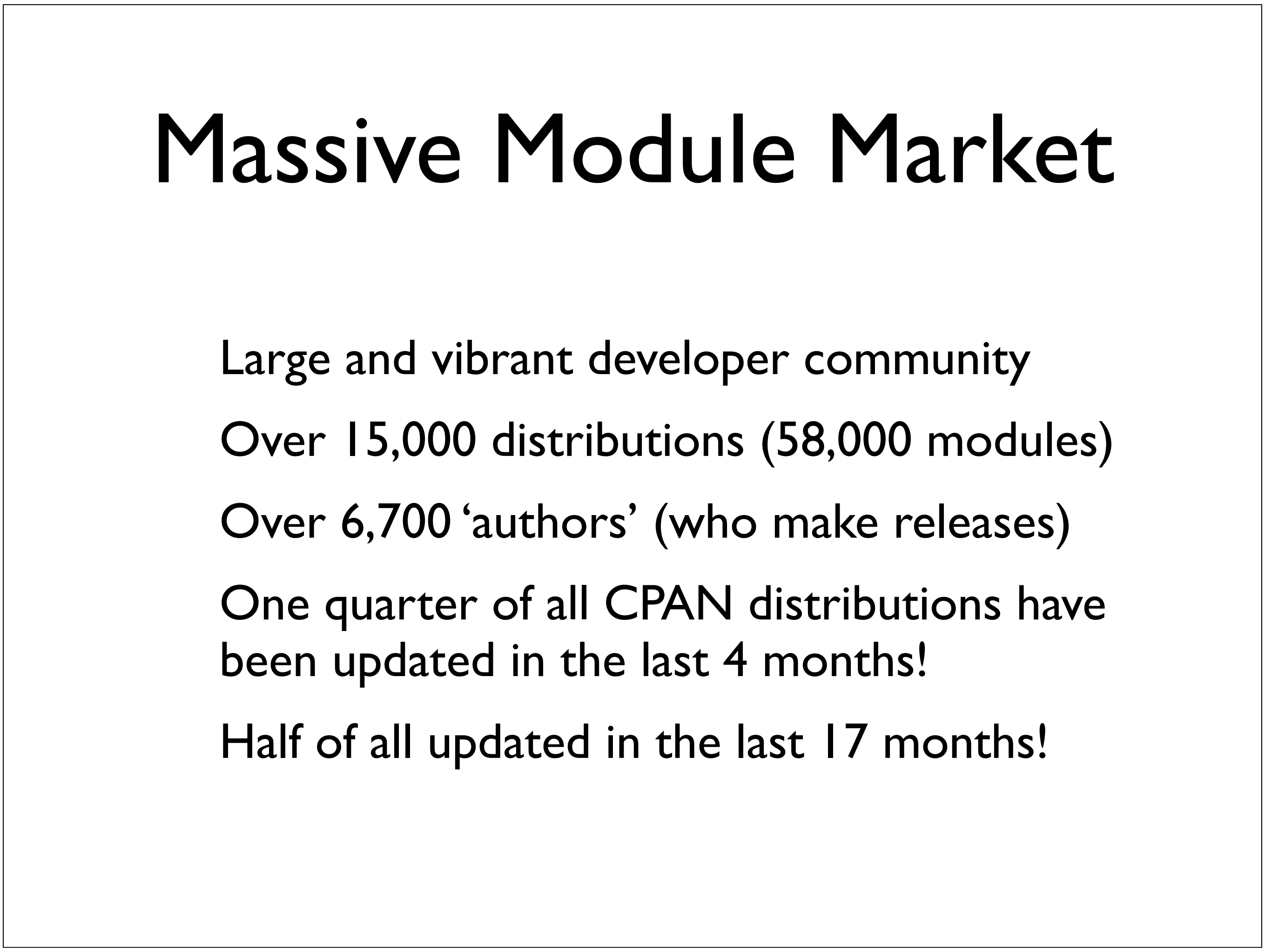
Perl is mentioned in many more software engineer/developer jobs.

"*foo* developer"



Perl is the primary focus of more developer jobs.
Want a fun new job? Become a Perl developer!

The existence of "*foo* developer" job titles is a sign of maturity.



Massive Module Market

Large and vibrant developer community

Over 15,000 distributions (58,000 modules)

Over 6,700 ‘authors’ (who make releases)

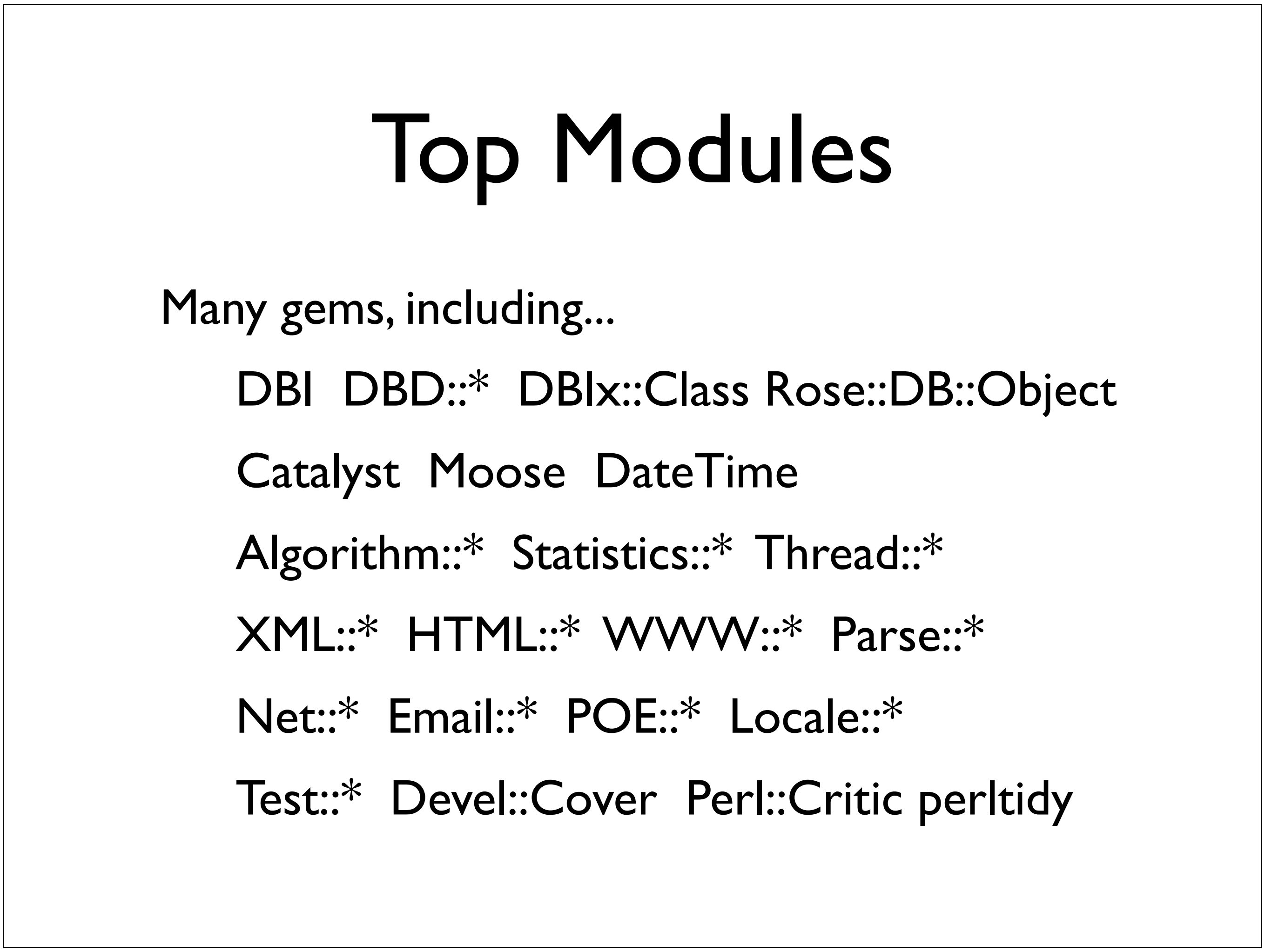
One quarter of all CPAN distributions have been updated in the last 4 months!

Half of all updated in the last 17 months!

Libraries are more important than languages.

Large user community leads to large contributor community
(given good community tools - more on that later)

http://search.cpan.org/recent
http://schwern.org/~schwern/talks/Perl%20is%20unDead%20-%20YAPC-NA-2008.pdf



Top Modules

Many gems, including...

- DBI DBD::* DBIx::Class Rose::DB::Object
- Catalyst Moose DateTime
- Algorithm::* Statistics::* Thread::*
- XML::* HTML::* WWW::* Parse::*
- Net::* Email::* POE::* Locale::*
- Test::* Devel::Cover Perl::Critic perltidy

Quality varies on CPAN, naturally.
(See Acme::* for some fun.)

http://www.serpentine.com/blog/2008/02/19/peruse-popular-perl-packages/

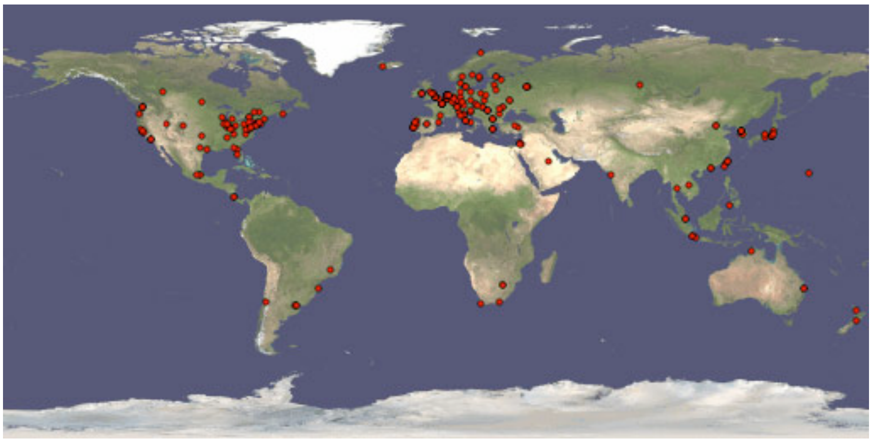
“Moose is pretty much the most exciting thing I’ve seen come out of the Perl 5 world in quite some time, and I’m really enjoying using it for my projects”

(My apologies if your favourite module isn’t included here.)

CPAN

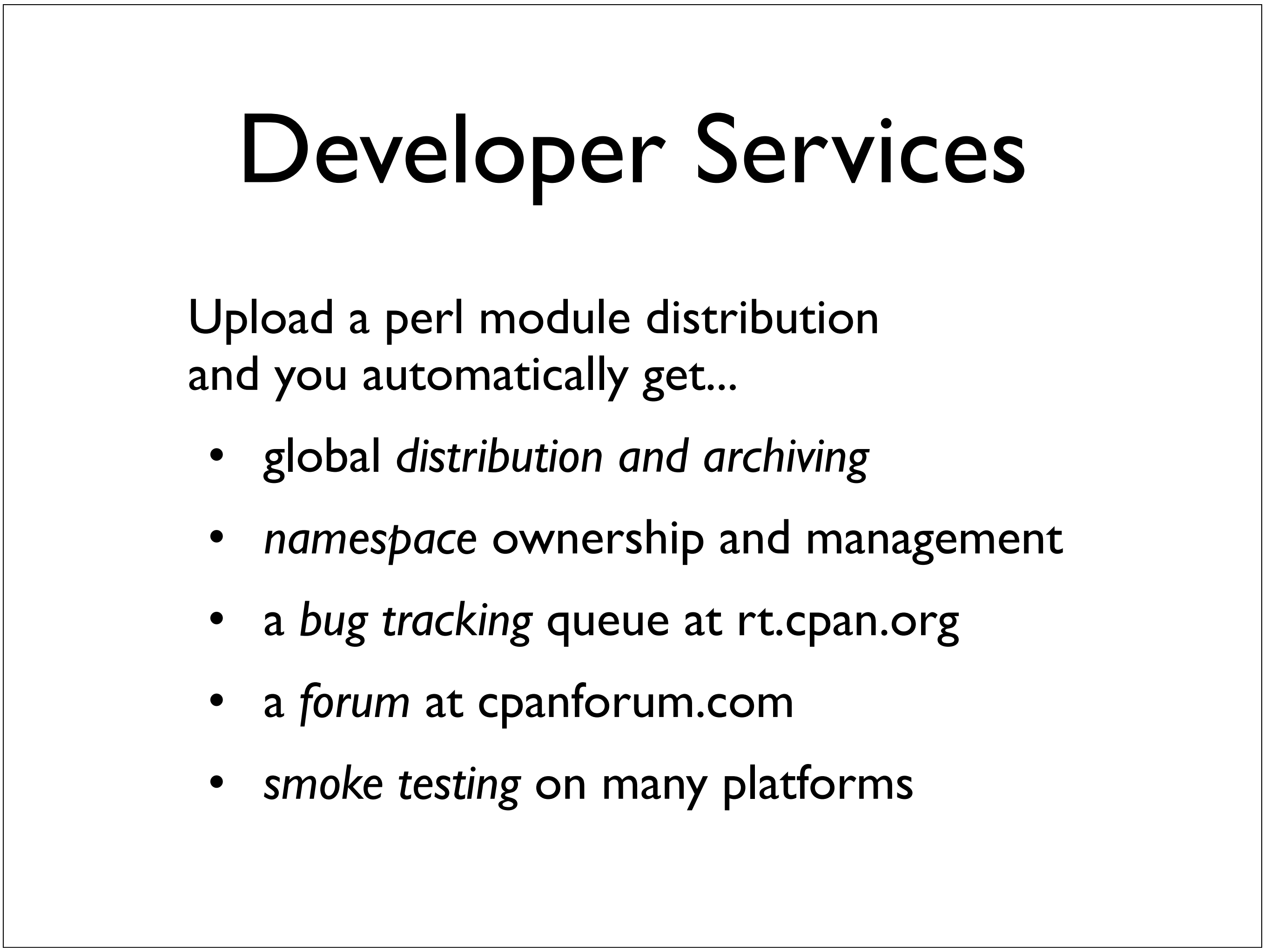
Comprehensive Perl Archive Network

360 mirrors in 51 regions (TLDs)



CPAN handles the global distribution.

Mirror status http://www.cs.uu.nl/stats/mirmon/cpan.html



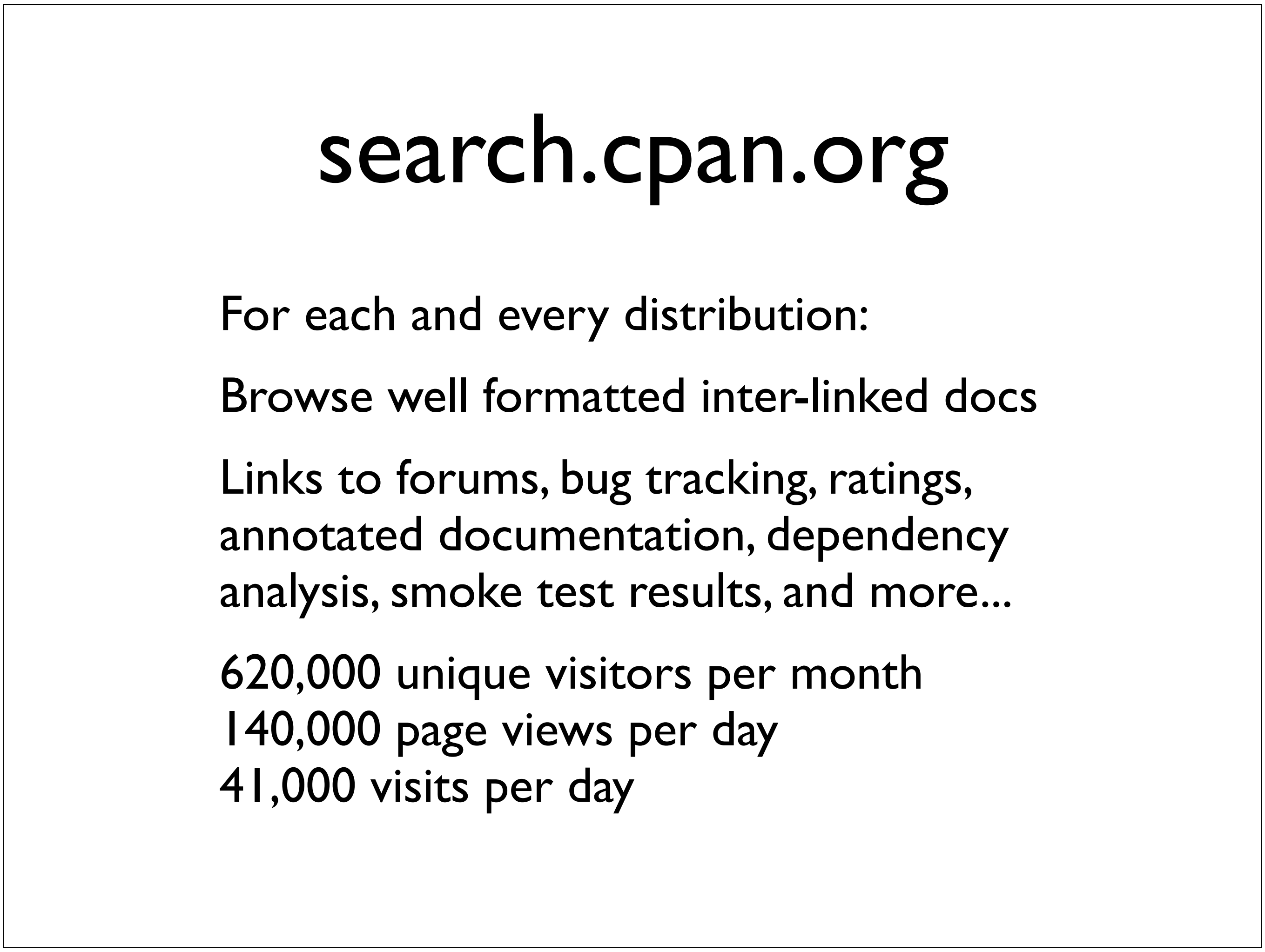
Developer Services

Upload a perl module distribution
and you automatically get...

- global *distribution and archiving*
- *namespace* ownership and management
- a *bug tracking* queue at rt.cpan.org
- a *forum* at cpanforum.com
- *smoke testing* on many platforms

This is a mature environment with rich services.

I'll discuss smoke testing and quality a little later



search.cpan.org

For each and every distribution:

Browse well formatted inter-linked docs

Links to forums, bug tracking, ratings, annotated documentation, dependency analysis, smoke test results, and more...

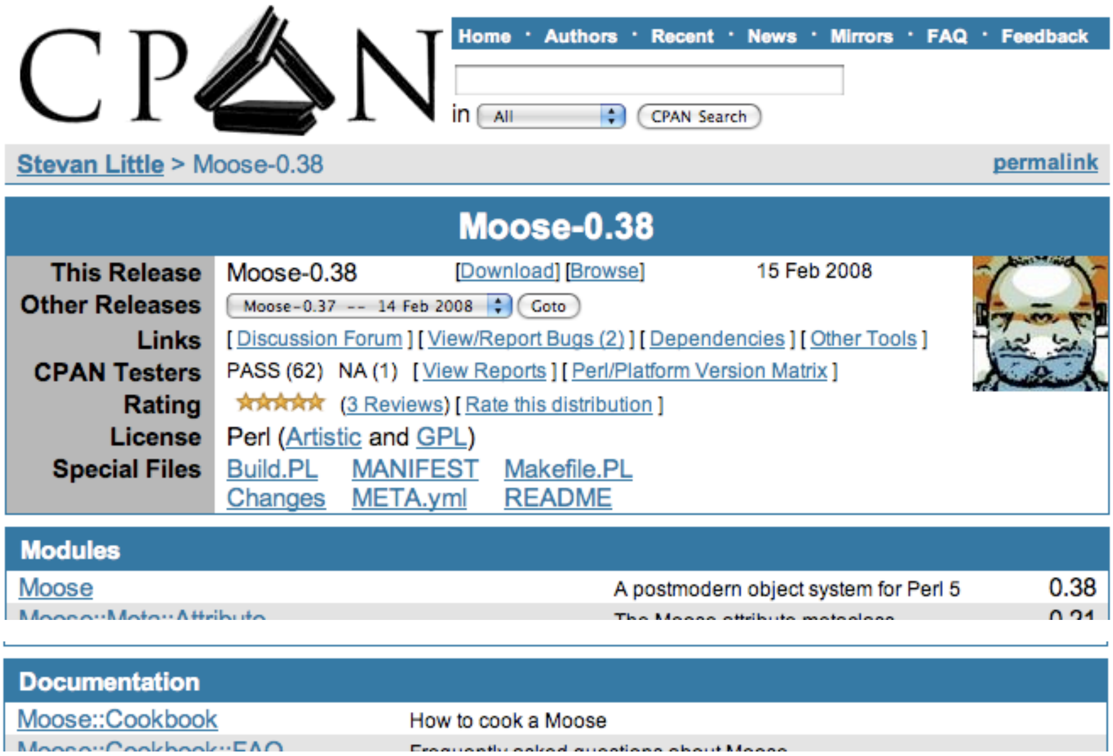
620,000 unique visitors per month
140,000 page views per day
41,000 visits per day

The primary face of CPAN

TOOLS Also has other tools, like grep'ing and diff'ing the distributions without downloading them.

DOWNLOAD You can download distributions but most people use automated tools for that.

Source for stats: google analytics (added 31st Jan, so unique visitor count is probably too low and its too soon to see trends)



Stevan Little > Moose-0.38 permalink

Moose-0.38

This Release	Moose-0.38 [Download] [Browse] 15 Feb 2008
Other Releases	Moose-0.37 -- 14 Feb 2008 Goto
Links	[Discussion Forum] [View/Report Bugs (2)] [Dependencies] [Other Tools]
CPAN Testers	PASS (62) NA (1) [View Reports] [Perl/Platform Version Matrix]
Rating	★★★★★ (3 Reviews) [Rate this distribution]
License	Perl (Artistic and GPL)
Special Files	Build.PL MANIFEST Makefile.PL Changes META.yml README

Modules

Moose	A postmodern object system for Perl 5	0.38
Moose::Meta::Attribute	The Moose attribute metaclass	0.21

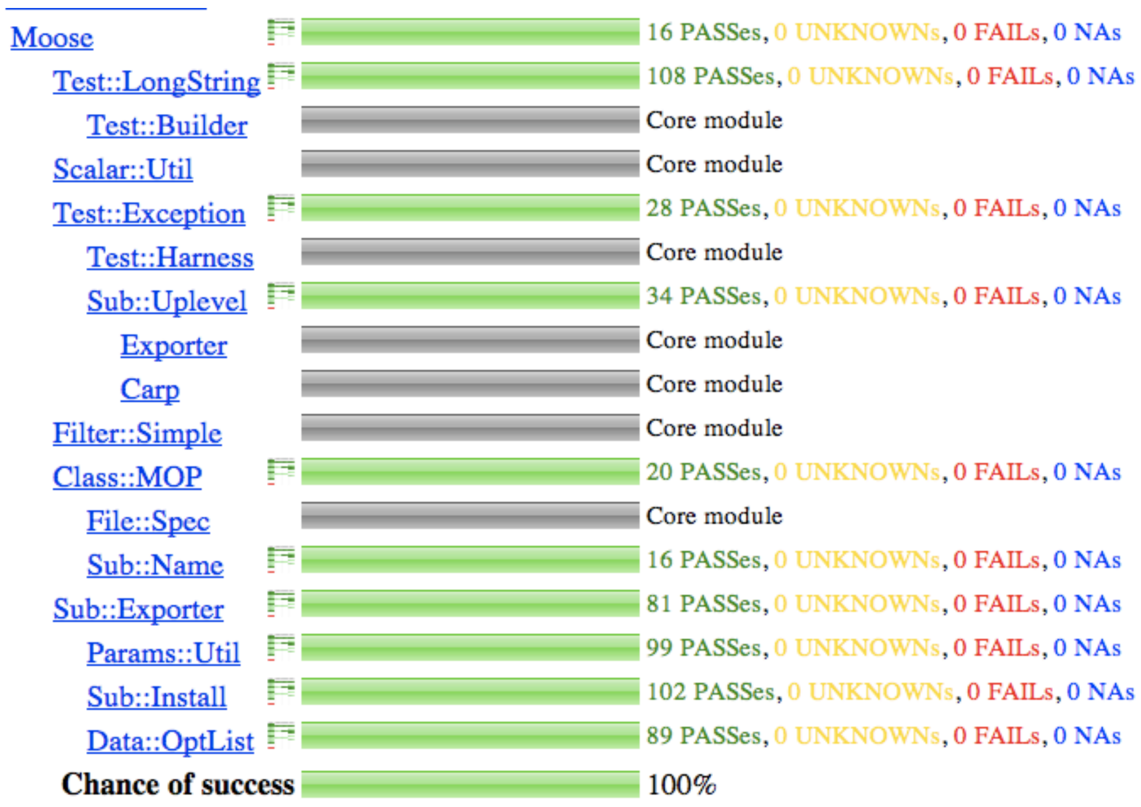
Documentation

Moose::Cookbook	How to cook a Moose
Moose::Cookbook::FAQ	Frequently asked questions about Moose

Example page for a distribution
Showing rich set of features

Dependency Analysis available for all Modules

http://bbbike.radzeit.de/~slaven/cpantestersmatrix.cgi

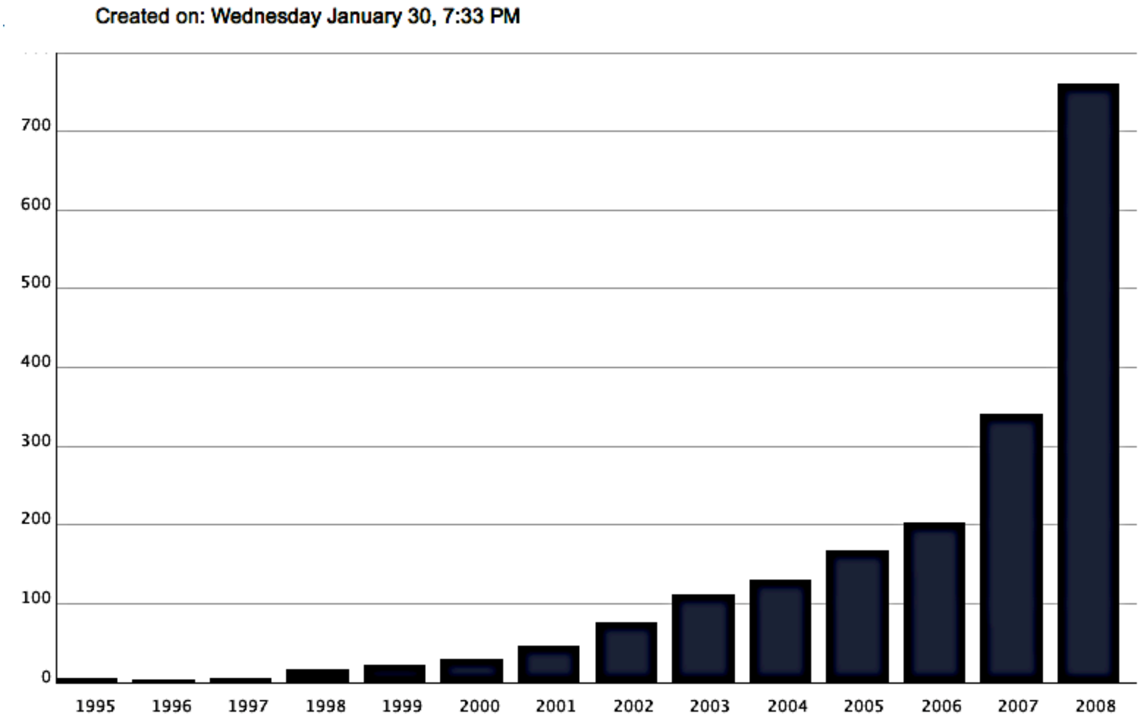


```
Moose                        16 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
  Test::LongString           108 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    Test::Builder            Core module
  Scalar::Util               Core module
  Test::Exception            28 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    Test::Harness            Core module
    Sub::Uplevel             34 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
      Exporter               Core module
      Carp                   Core module
  Filter::Simple             Core module
  Class::MOP                 20 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    File::Spec               Core module
    Sub::Name                16 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
  Sub::Exporter              81 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    Params::Util             99 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    Sub::Install             102 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
    Data::OptList            89 PASSes, 0 UNKNOWNs, 0 FAILs, 0 NAs
Chance of success            100%
```

Dependencies need not be hell!

Shows tree of dependencies
Identify risks.
Refine the view to match you particular operating system and perl version

Average Uploads/Month

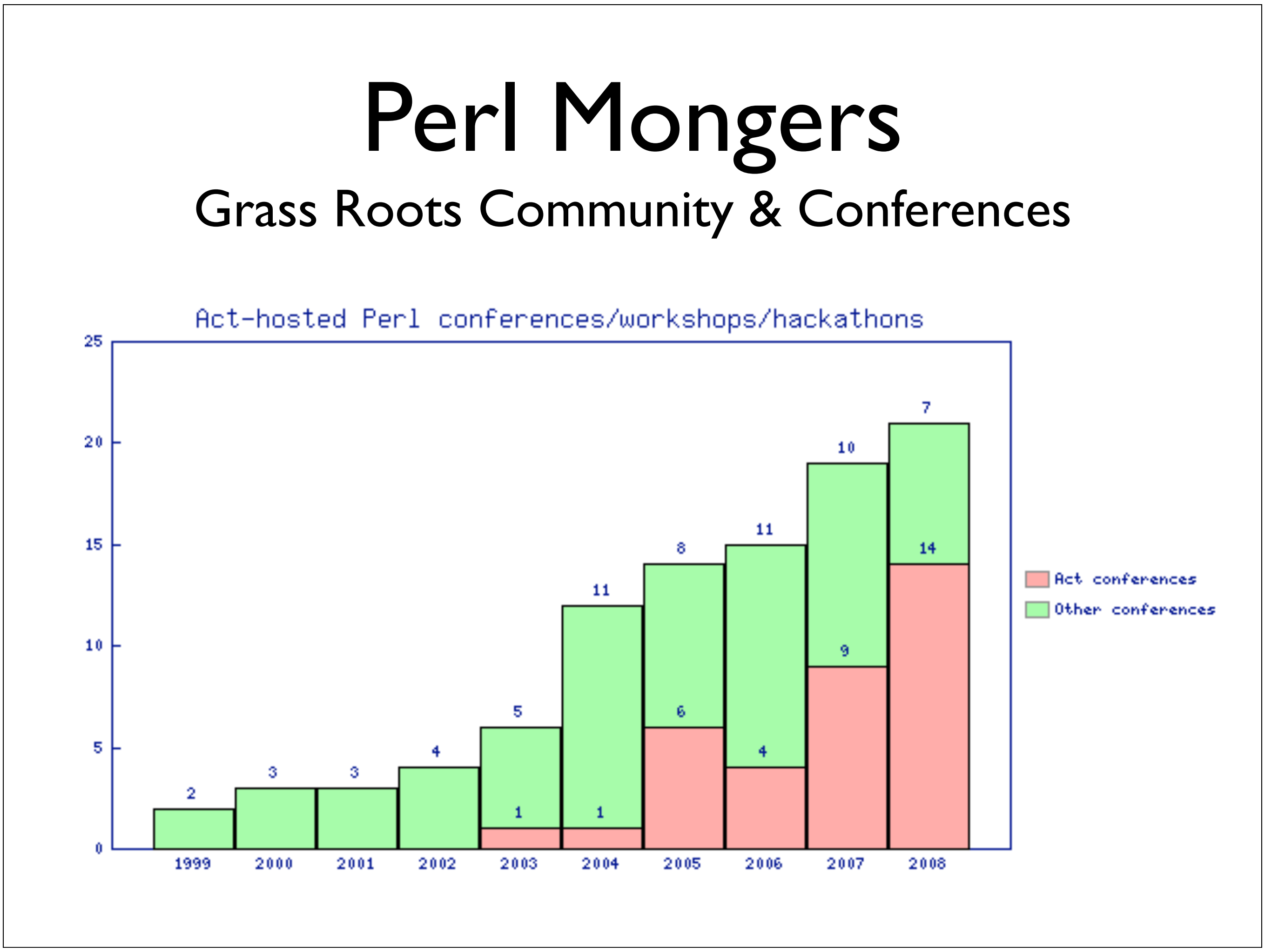


Did I say it was a vibrant and growing developer community?

(2008 may be slightly exaggerated as it’s extrapolated from January data.)
The trend is clear.

Certainly doesn’t look like a dead language!

http://services.alphaworks.ibm.com/manyeyes/view/SmAgULsOtha6C7W~7CInL2~



Perl Mongers

Grass Roots Community & Conferences

http://use.perl.org/article.pl?sid=08/04/29/0512200

(“Act (A Conference Toolkit) is a multilingual, template-driven, multi-conference web site that can manage the users, talks, schedule and payment for your conference”)

~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5

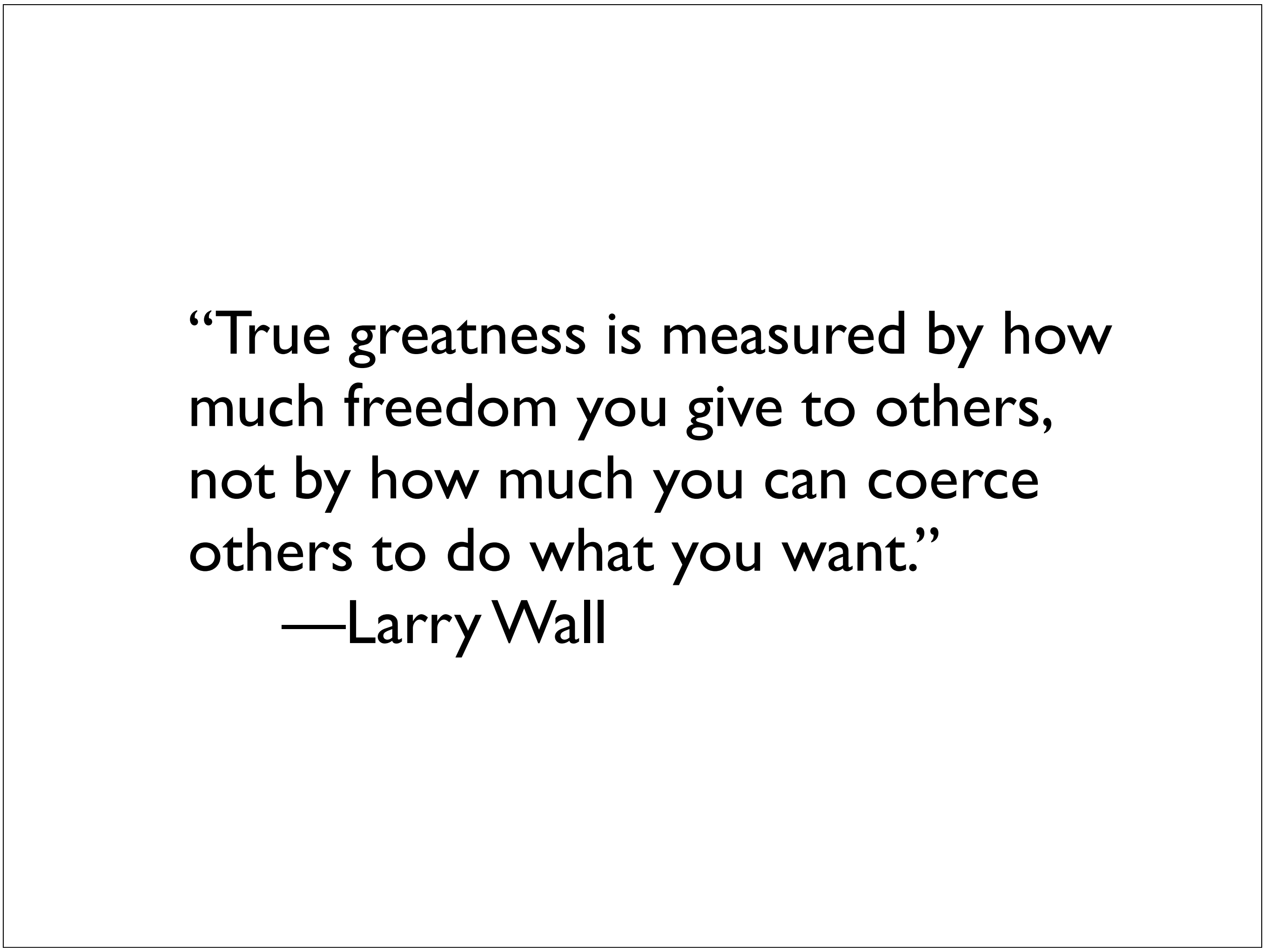
~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

Perl 6 is killing Perl 5



> “True greatness is measured by how much freedom you give to others, not by how much you can coerce others to do what you want.”
>
> —Larry Wall

I greatly value the freedoms perl give me
Freedoms in the language, the community, the technology, the culture.
With freedom comes responsibility
You can write poorly in any language.
You can write beautiful code in Perl.
http://jeremy.zawodny.com/blog/archives/009873.html#comment-39486

Guidelines and Tools

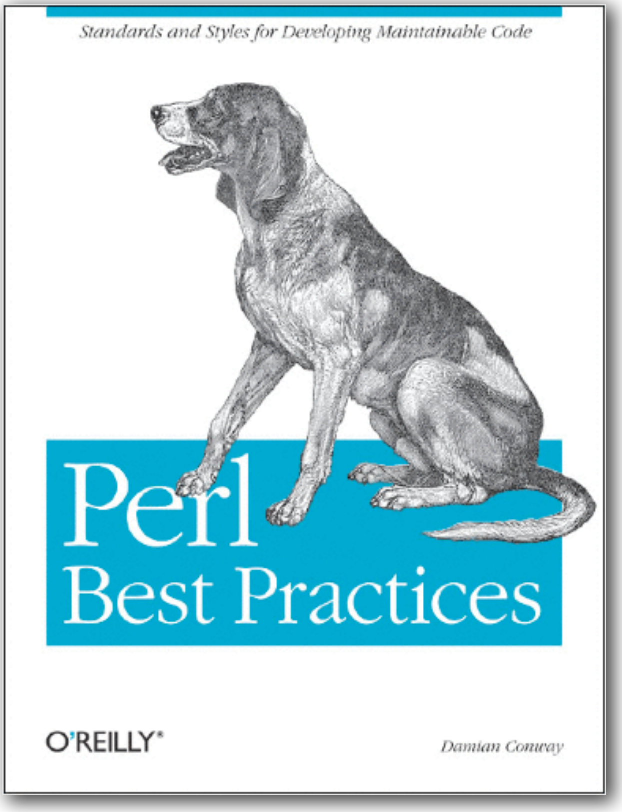
Perl Best Practices

Perl::Tidy

Perl::Critic

Test::*

Devel::Cover



More than one way to do it, but...

Already a growing culture of quality and testing when the book came out in July 2005
256 guidelines

Agreeing on one set of sane guidelines
is more important than the exact details of the guidelines.

PBP makes it easy for a team to agree “We’ll just follow PBP guidelines”.

Perl::Tidy

Perl code beautifier

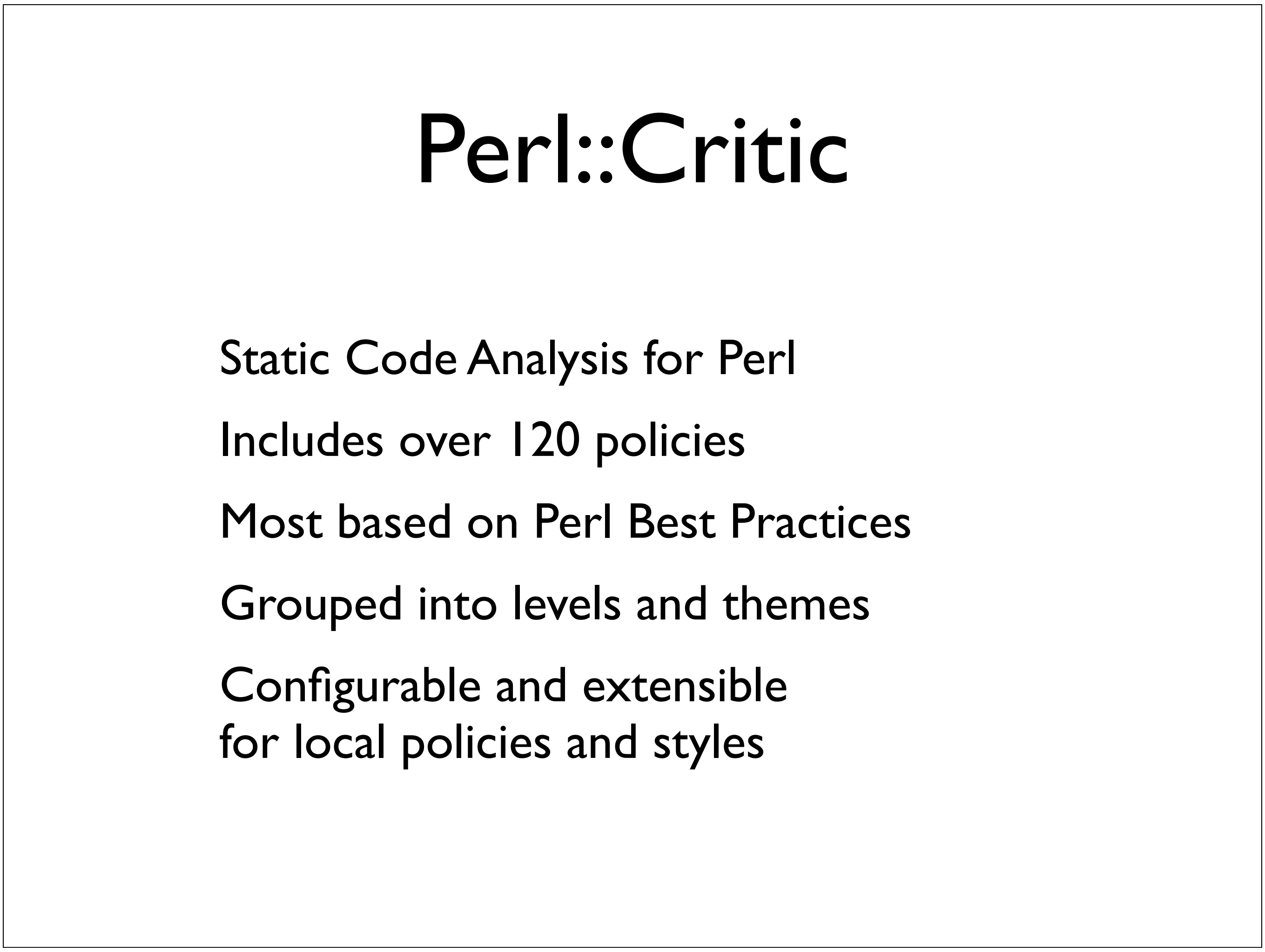
Works beautifully - can be trusted

Supports *many* options for personal styles

Perl Best Practices recommended options

Normalise the coding style of existing code.
Very simple and effective way to add clarity to a code base.

Now you’ve got pretty code,
    but is it good code?
CRITIC (next)



Perl::Critic

Static Code Analysis for Perl

Includes over 120 policies

Most based on Perl Best Practices

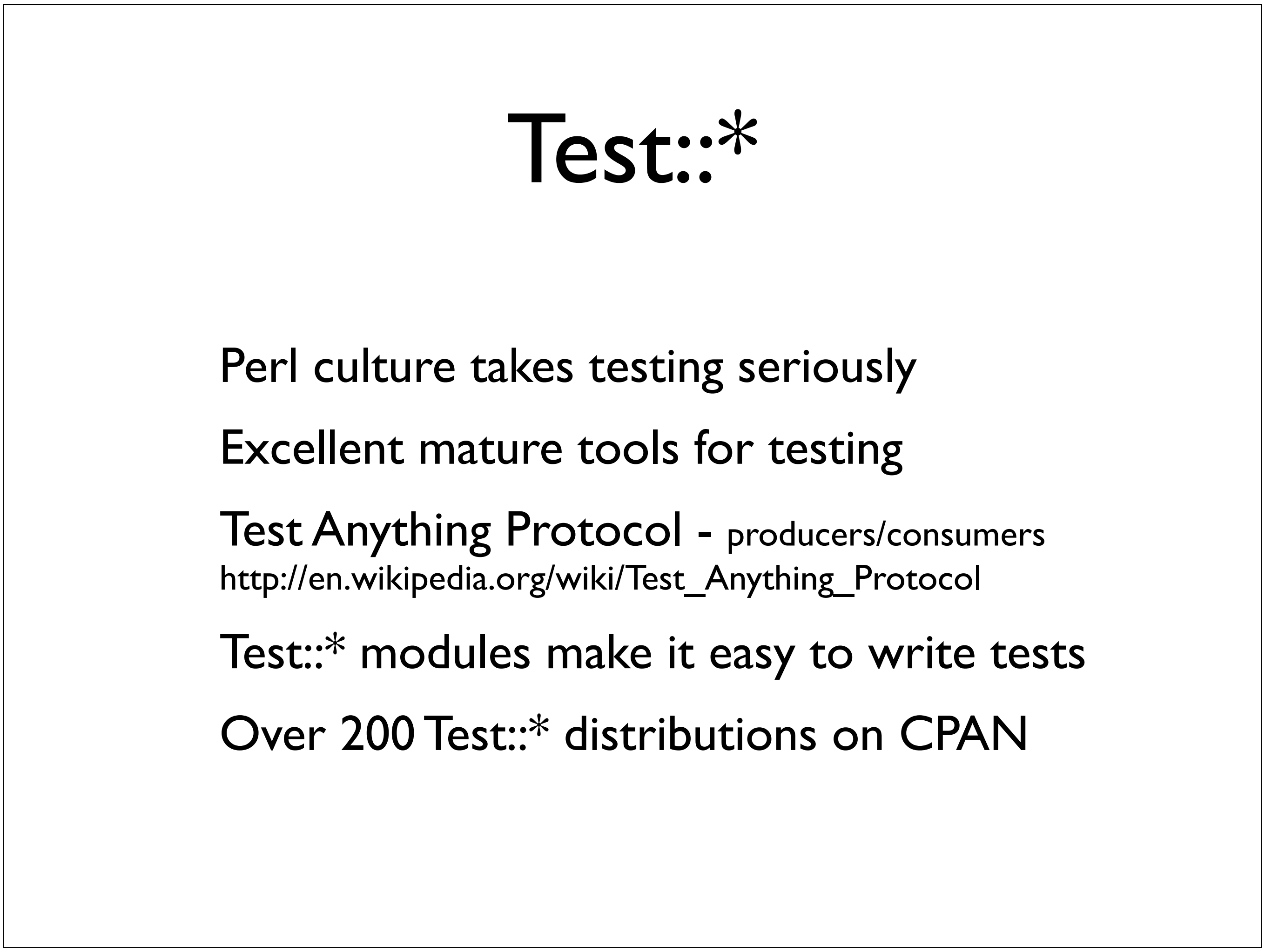
Grouped into levels and themes

Configurable and extensible
for local policies and styles

An extensible framework for creating and applying coding standards to Perl source code

Perltidy address the layout of code.
Perlcritic addresses the semantics.

Now you've got pretty code,
    that follows best practices,
    but does it work?
... TESTING (next)



Test::*

- Perl culture takes testing seriously
- Excellent mature tools for testing
- Test Anything Protocol - producers/consumers
  http://en.wikipedia.org/wiki/Test_Anything_Protocol
- Test::* modules make it easy to write tests
- Over 200 Test::* distributions on CPAN

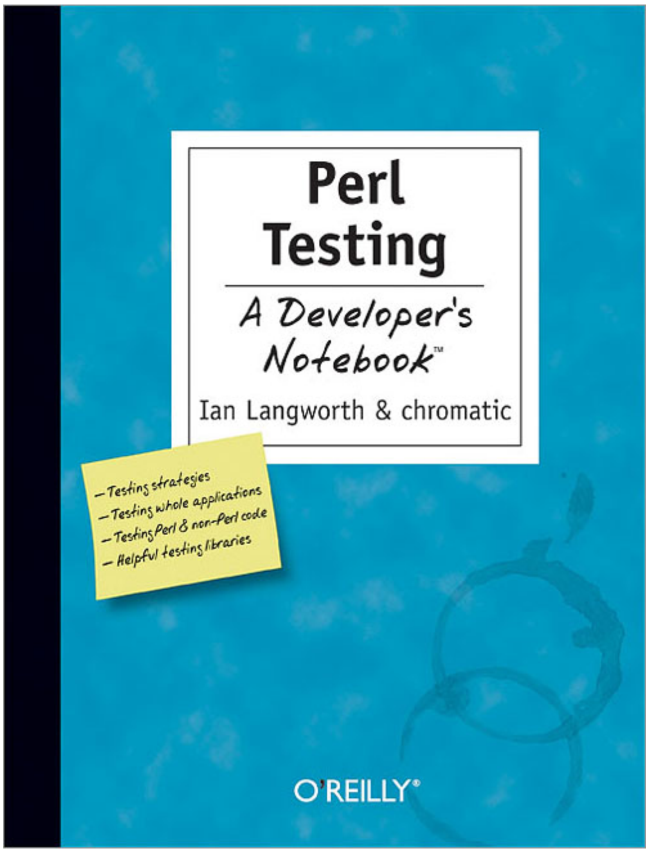
Also many ‘mock’ modules for mocking objects and other functionality to ease testing

Test::Class provides xUnit styles tests

The test modules work together.

http://search.cpan.org/search?m=dist&q=Test%3A%3A&s=1&n=100

Now...
    you’ve got pretty code,
    that follows best practices,
    has tests and the tests pass,
    but how much code is exercised by the tests?
COVERAGE (next)



Perl Testing

A Developer's Notebook™

Ian Langworth & chromatic

- Testing strategies
- Testing whole applications
- Testing Perl & non-Perl code
- Helpful testing libraries

O'REILLY®

Published July 2005

http://www.oreilly.com/catalog/perltestingadn/

Devel::Cover

Coverage Analysis for Perl

Tells you what code has been executed

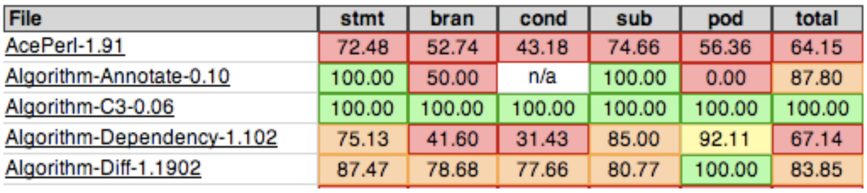
Statement, branch, condition, subroutine, pod

Produces drill-down reports in HTML

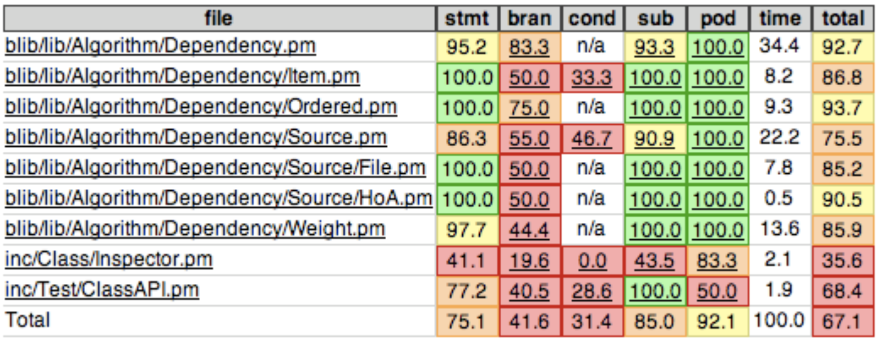
Includes documentation coverage analysis.

Devel::Cover Reports

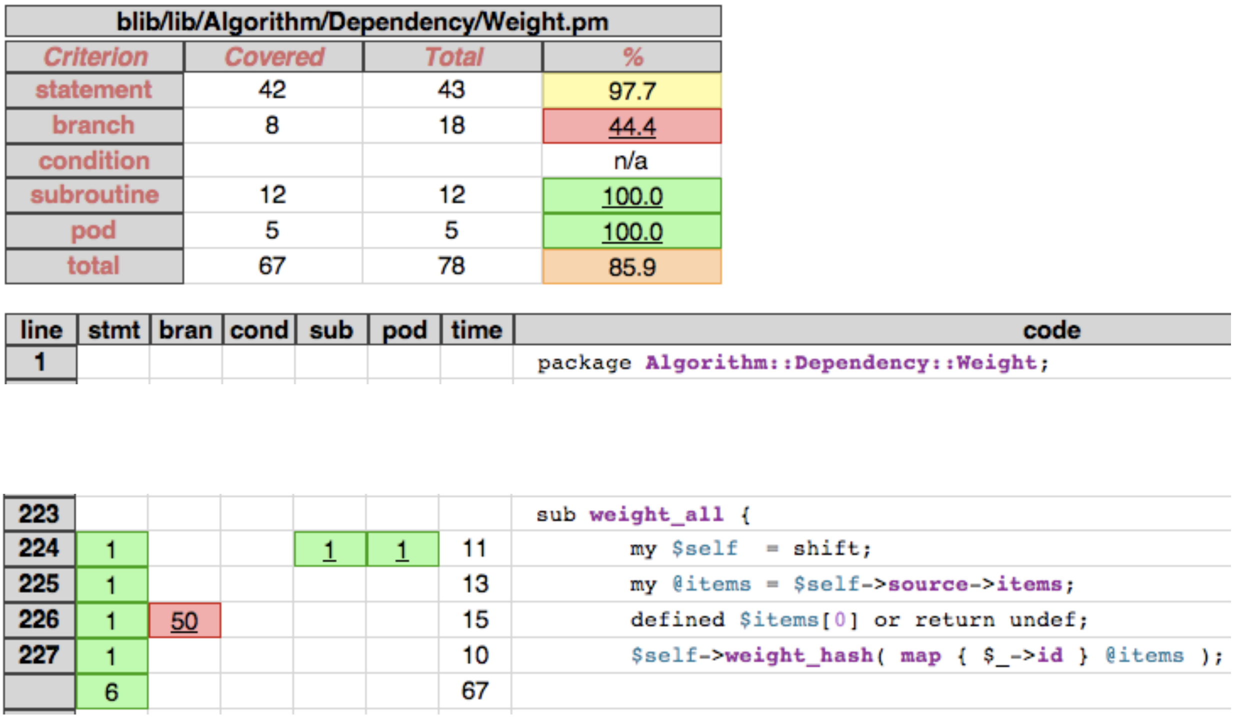
http://pjcj.sytes.net/cover/latest/



File	stmt	bran	cond	sub	pod	total
AcePerl-1.91	72.48	52.74	43.18	74.66	56.36	64.15
Algorithm-Annotate-0.10	100.00	50.00	n/a	100.00	0.00	87.80
Algorithm-C3-0.06	100.00	100.00	100.00	100.00	100.00	100.00
Algorithm-Dependency-1.102	75.13	41.60	31.43	85.00	92.11	67.14
Algorithm-Diff-1.1902	87.47	78.68	77.66	80.77	100.00	83.85

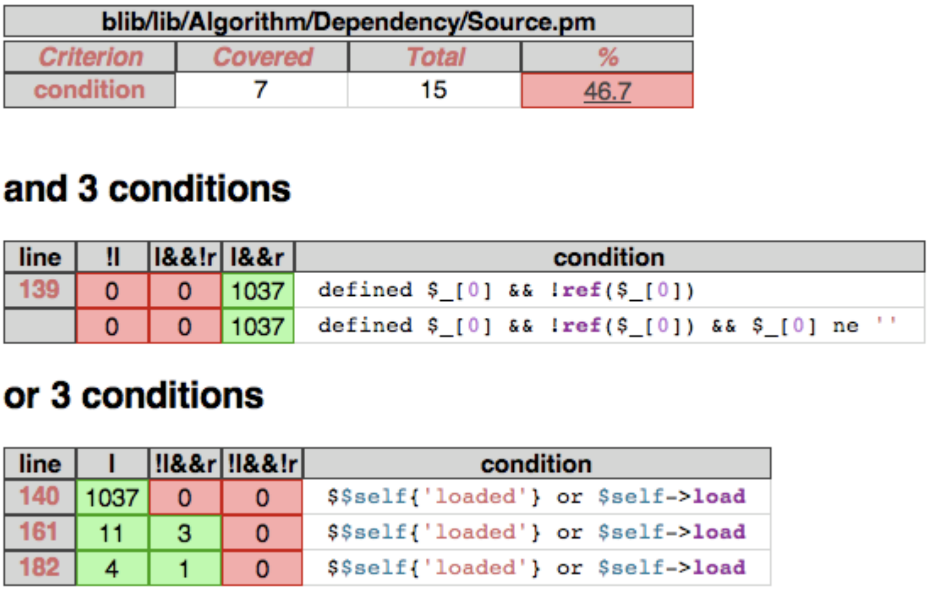


file	stmt	bran	cond	sub	pod	time	total
blib/lib/Algorithm/Dependency.pm	95.2	83.3	n/a	93.3	100.0	34.4	92.7
blib/lib/Algorithm/Dependency/Item.pm	100.0	50.0	33.3	100.0	100.0	8.2	86.8
blib/lib/Algorithm/Dependency/Ordered.pm	100.0	75.0	n/a	100.0	100.0	9.3	93.7
blib/lib/Algorithm/Dependency/Source.pm	86.3	55.0	46.7	90.9	100.0	22.2	75.5
blib/lib/Algorithm/Dependency/Source/File.pm	100.0	50.0	n/a	100.0	100.0	7.8	85.2
blib/lib/Algorithm/Dependency/Source/HoA.pm	100.0	50.0	n/a	100.0	100.0	0.5	90.5
blib/lib/Algorithm/Dependency/Weight.pm	97.7	44.4	n/a	100.0	100.0	13.6	85.9
inc/Class/Inspector.pm	41.1	19.6	0.0	43.5	83.3	2.1	35.6
inc/Test/ClassAPI.pm	77.2	40.5	28.6	100.0	50.0	1.9	68.4
Total	75.1	41.6	31.4	85.0	92.1	100.0	67.1



blib/lib/Algorithm/Dependency/Weight.pm			
Criterion	*Covered*	*Total*	*%*
statement	42	43	97.7
branch	8	18	44.4
condition			n/a
subroutine	12	12	100.0
pod	5	5	100.0
total	67	78	85.9

line	stmt	bran	cond	sub	pod	time	code
1							`package Algorithm::Dependency::Weight;`
223							`sub weight_all {`
224	1			1	1	11	`my $self  = shift;`
225	1					13	`my @items = $self->source->items;`
226	1	50				15	`defined $items[0] or return undef;`
227	1					10	`$self->weight_hash( map { $_->id } @items );`
	6					67	



blib/lib/Algorithm/Dependency/Source.pm			
Criterion	*Covered*	*Total*	*%*
condition	7	15	46.7

and 3 conditions

line	!l	l&&!r	l&&r	condition
139	0	0	1037	defined $_[0] && !ref($_[0])
	0	0	1037	defined $_[0] && !ref($_[0]) && $_[0] ne ''

or 3 conditions

line	l	!l&&r	!l&&!r	condition
140	1037	0	0	$$self{'loaded'} or $self->load
161	11	3	0	$$self{'loaded'} or $self->load
182	4	1	0	$$self{'loaded'} or $self->load

~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

BUSTED!

Perl 6 is killing Perl 5

~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

BUSTED!

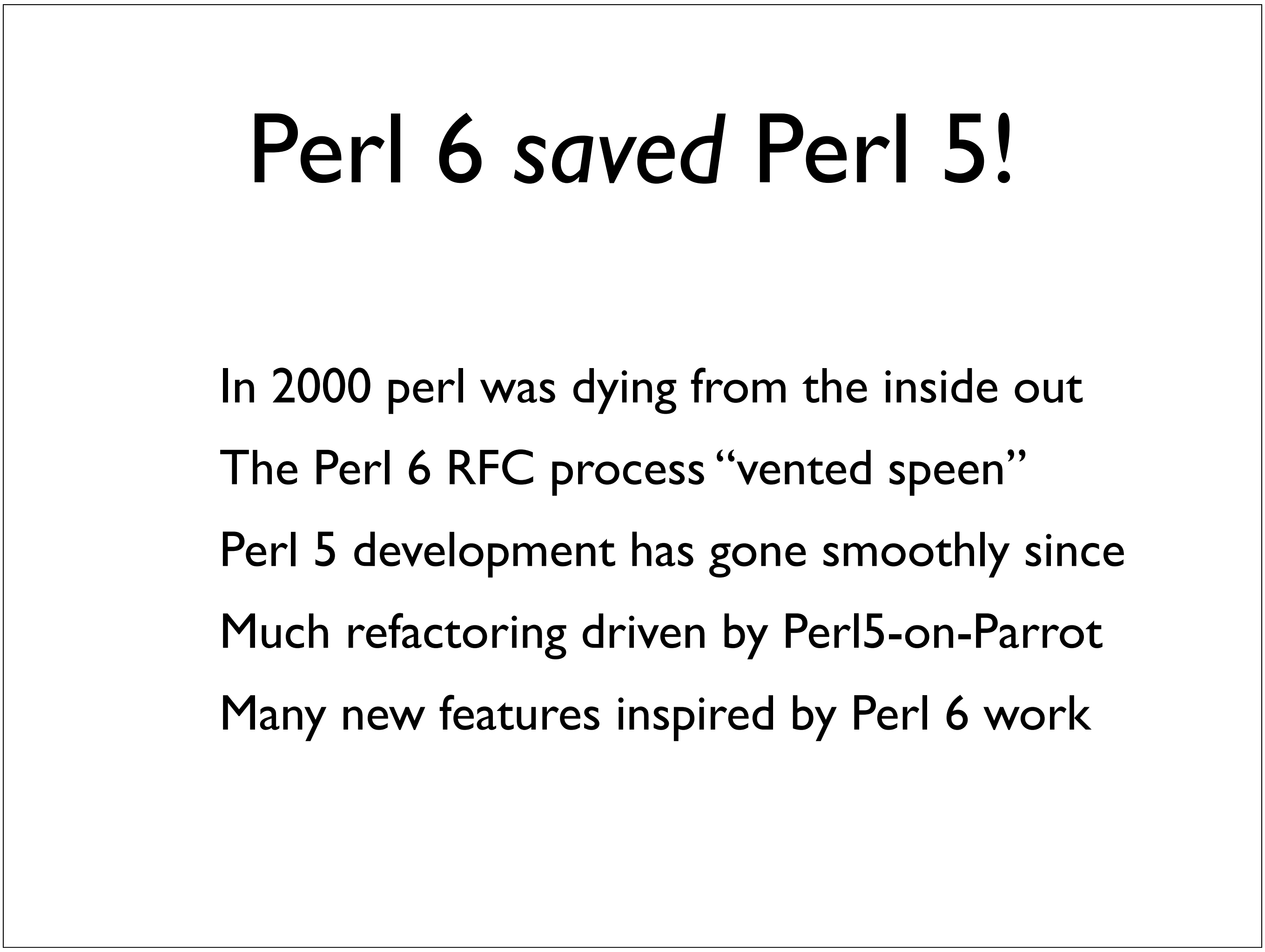
Perl 6 is killing Perl 5

Perl 6 *saved* Perl 5!

“Perl 5 had already started dying, because people were starting to see it as a dead-end language.

It seemed odd at the time, but when we announced Perl 6, Perl 5 suddenly took on a new life.”
—Larry Wall, 2002

http://www.perlfoundation.org/perl6/index.cgi?state_of_the_onion



Perl 6 *saved* Perl 5!

In 2000 perl was dying from the inside out

The Perl 6 RFC process “vented speen”

Perl 5 development has gone smoothly since

Much refactoring driven by Perl5-on-Parrot

Many new features inspired by Perl 6 work

Back around 2000...

- Perl5 was getting very hard to maintain
- Bickering on the mailing lists
- Few volunteers for any real work
- Lack of direction

Using Perl 5 as one of the backends for Perl 6.

Perl 5.10

Perl 5.10 was released in December 2007
Five years after 5.8.0, two years after 5.8.8

Refactored internals

many fixes, more speed, less memory

Switch statement, smart matching, named captures, state variables, defined-or, say, field hashes, pluggable regex engines, trie-based non-recursive pattern matching, and more...

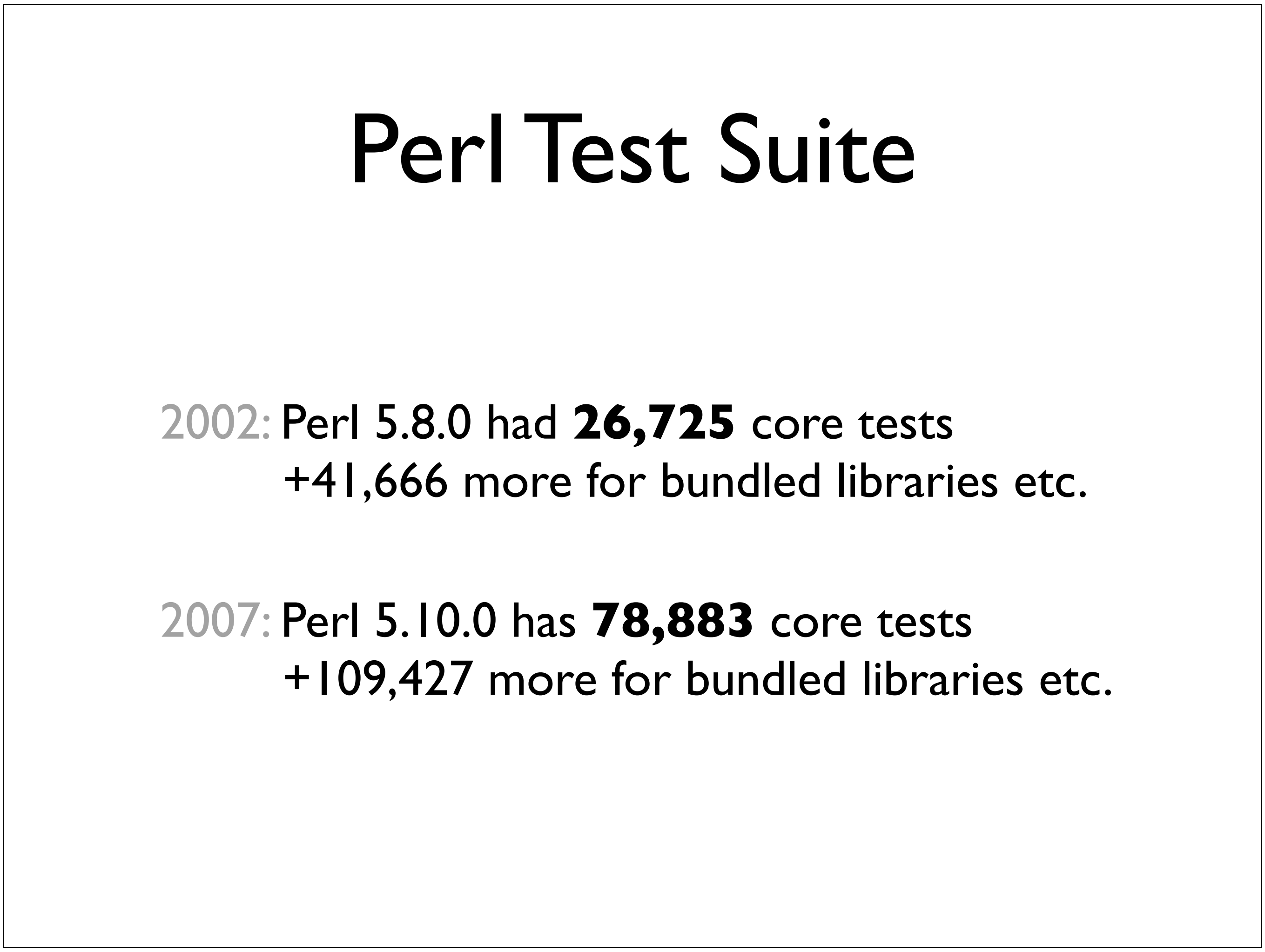
http://www.slideshare.net/rjbs/perl-510-for-people-who-arent-totally-insane

A Culture of Testing

Another bonus from Perl 6:

Strong test suites for Perl 5 code are needed to ensure backwards compatibility

Strengthened culture of testing



Perl Test Suite

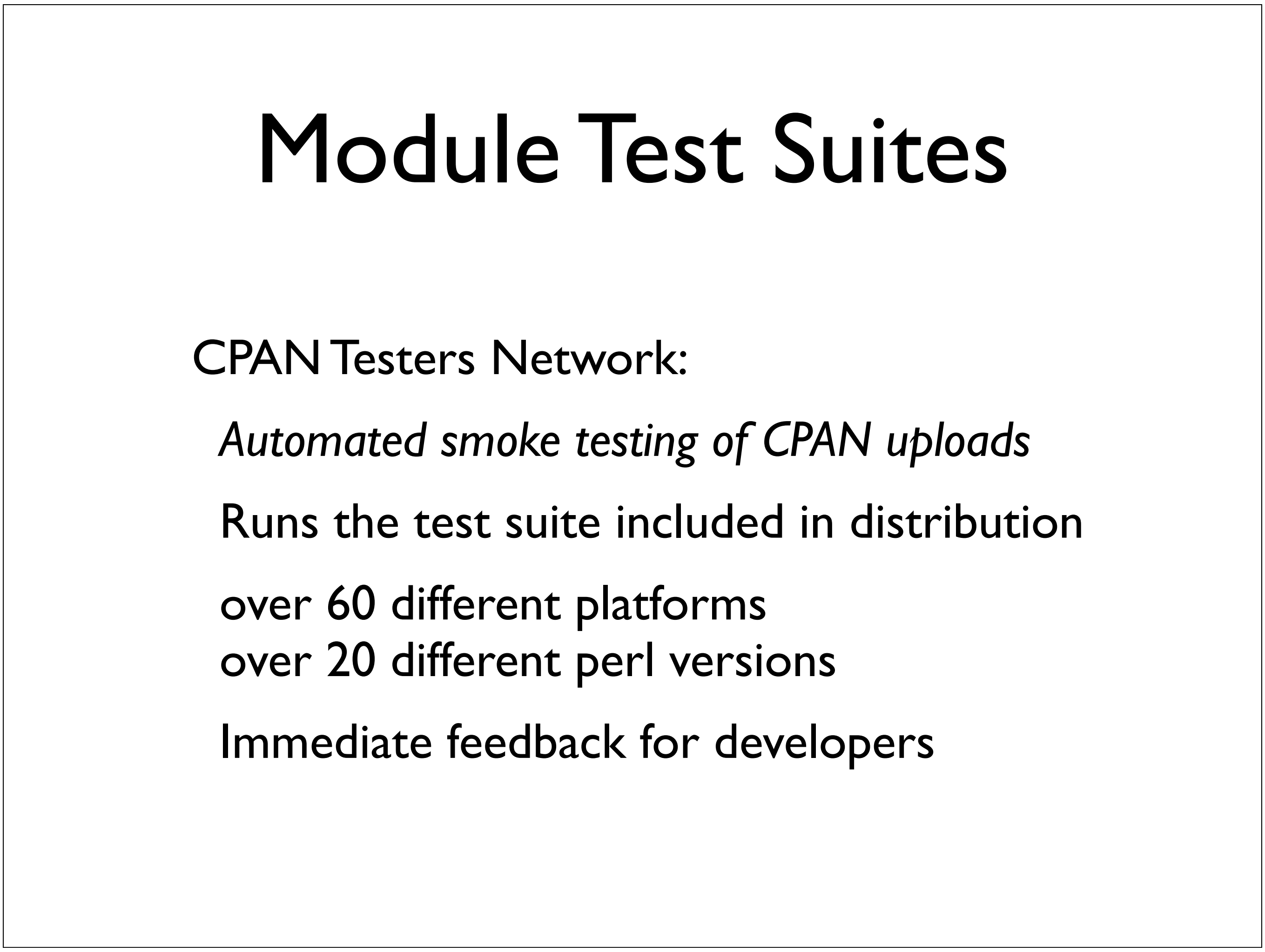
2002: Perl 5.8.0 had **26,725** core tests
+41,666 more for bundled libraries etc.

2007: Perl 5.10.0 has **78,883** core tests
+109,427 more for bundled libraries etc.

(I couldn’t find code coverage stats.)

For perspective: Ruby has ~1,400 core tests plus ~14,000 for bundled libraries etc.

See comments in http://www.oreillynet.com/onlamp/blog/2007/05/trust_but_verify.html
and http://reddit.com/info/1uzda/comments/



Module Test Suites

CPAN Testers Network:

Automated smoke testing of CPAN uploads

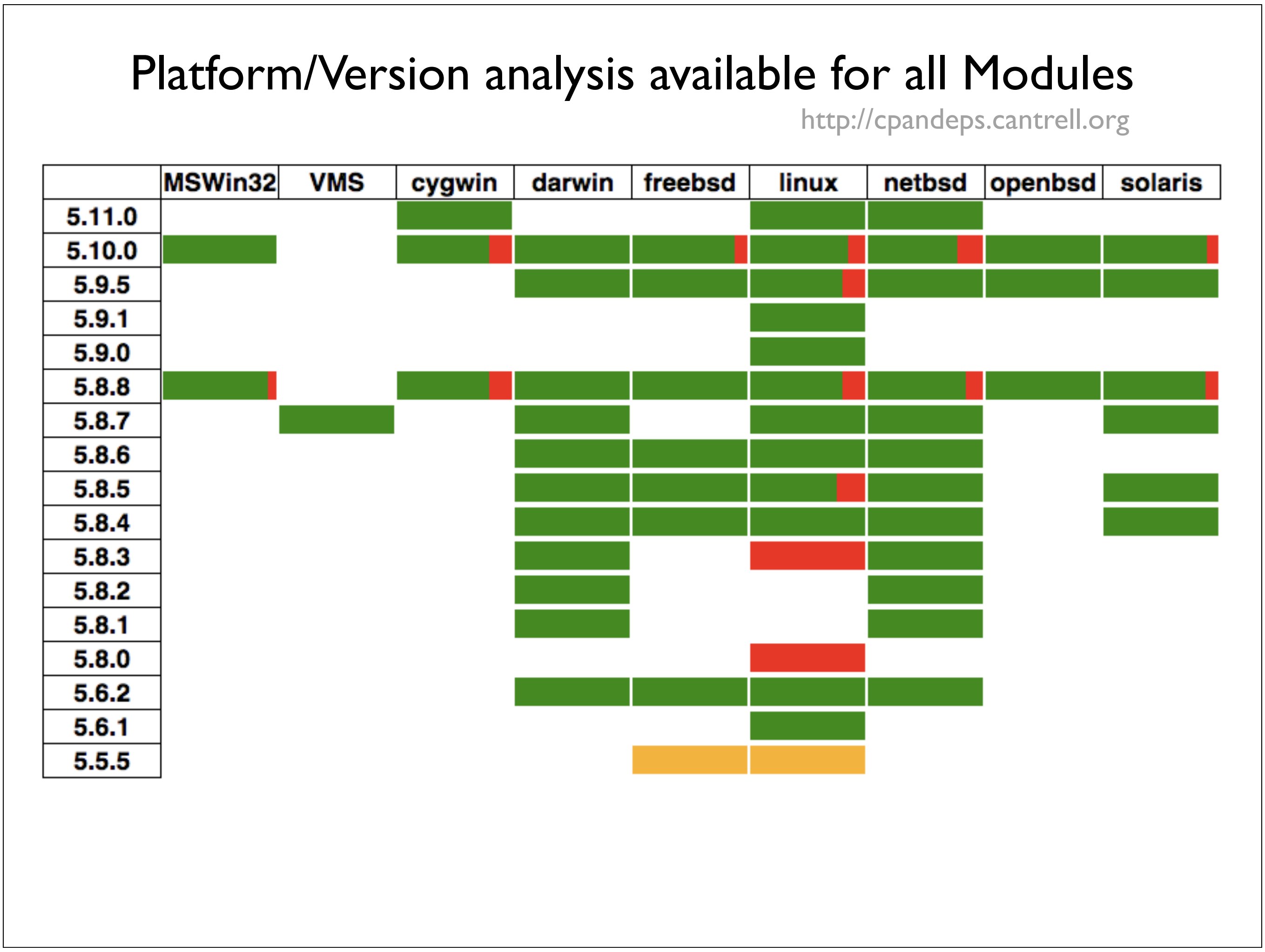
Runs the test suite included in distribution

over 60 different platforms
over 20 different perl versions

Immediate feedback for developers

Some by completely automated robots in virtual machines.

Some by users who submit reports as they download and test via the installer.



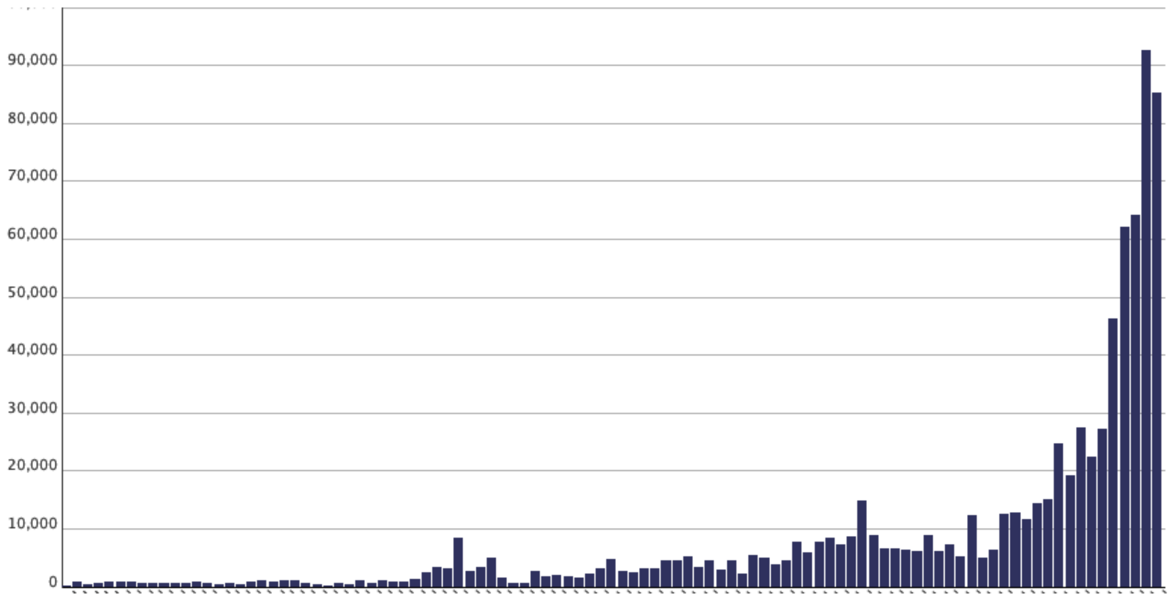
Platform/Version analysis available for all Modules

http://cpandeps.cantrell.org

	MSWin32	VMS	cygwin	darwin	freebsd	linux	netbsd	openbsd	solaris
5.11.0			🟩			🟩	🟩		
5.10.0	🟩		🟩🟥	🟩	🟩🟥	🟩🟥	🟩🟥	🟩	🟩🟥
5.9.5				🟩	🟩	🟩🟥	🟩	🟩	🟩
5.9.1						🟩			
5.9.0						🟩			
5.8.8	🟩🟥		🟩🟥	🟩	🟩	🟩🟥	🟩🟥	🟩	🟩🟥
5.8.7		🟩		🟩		🟩	🟩		🟩
5.8.6				🟩	🟩	🟩	🟩		
5.8.5				🟩	🟩	🟩🟥	🟩		🟩
5.8.4				🟩	🟩	🟩	🟩		🟩
5.8.3				🟩		🟥	🟩		
5.8.2				🟩			🟩		
5.8.1				🟩			🟩		
5.8.0						🟥			
5.6.2				🟩	🟩	🟩	🟩		
5.6.1						🟩			
5.5.5					🟧	🟧			

This one is for the DBI

~90,000 Reports/Month



Monthly - August 1999 to December 2007

http://services.alphaworks.ibm.com/manyeyes/view/SmAgULsOtha6g7GcD5KnL2-

Is perl6 killing perl5? Certainly doesn’t look like it!

~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

BUSTED!

Perl 6 is killing Perl 5

BUSTED!

~ Myths ~

Perl is dead

BUSTED!

Perl is hard to read / test / maintain

BUSTED!

Perl 6 is killing Perl 5

BUSTED!

Perl 6 Myths

“Perl 6 is not Perl”

“It'll never be finished”

“There's no code written in Perl 6”

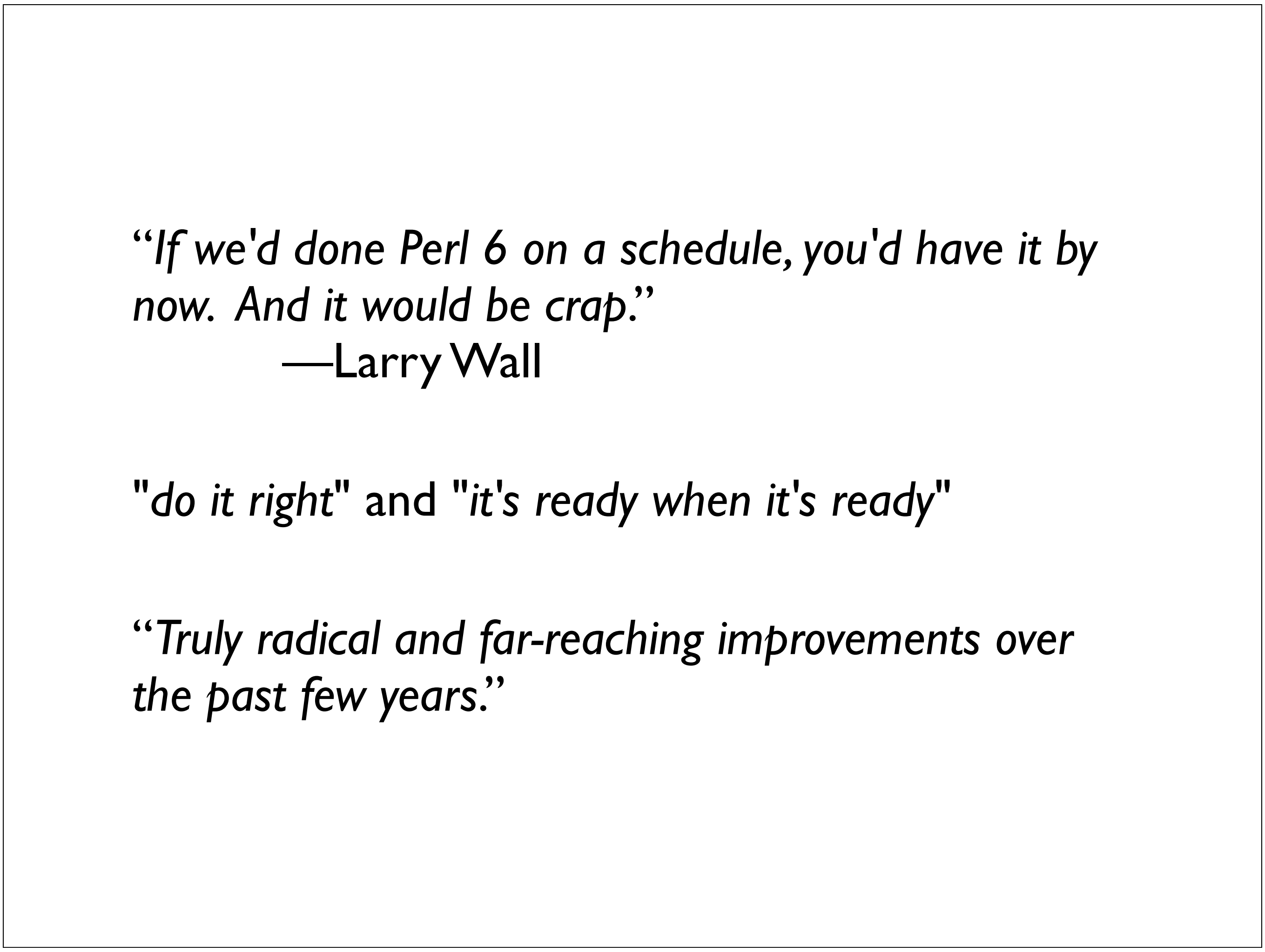
“Perl 6 is not Perl”

“Perl 6 [...] is clearer, more direct, more expressive, and without many of the old false leads and rough edges in Perl 5.

So, Perl 6 is still Perl in that a programmer looking at a Perl 6 program will instantly recognize that it is "Perl", even if some of the details are different.”

—Larry Wall

“It’ll never be finished”



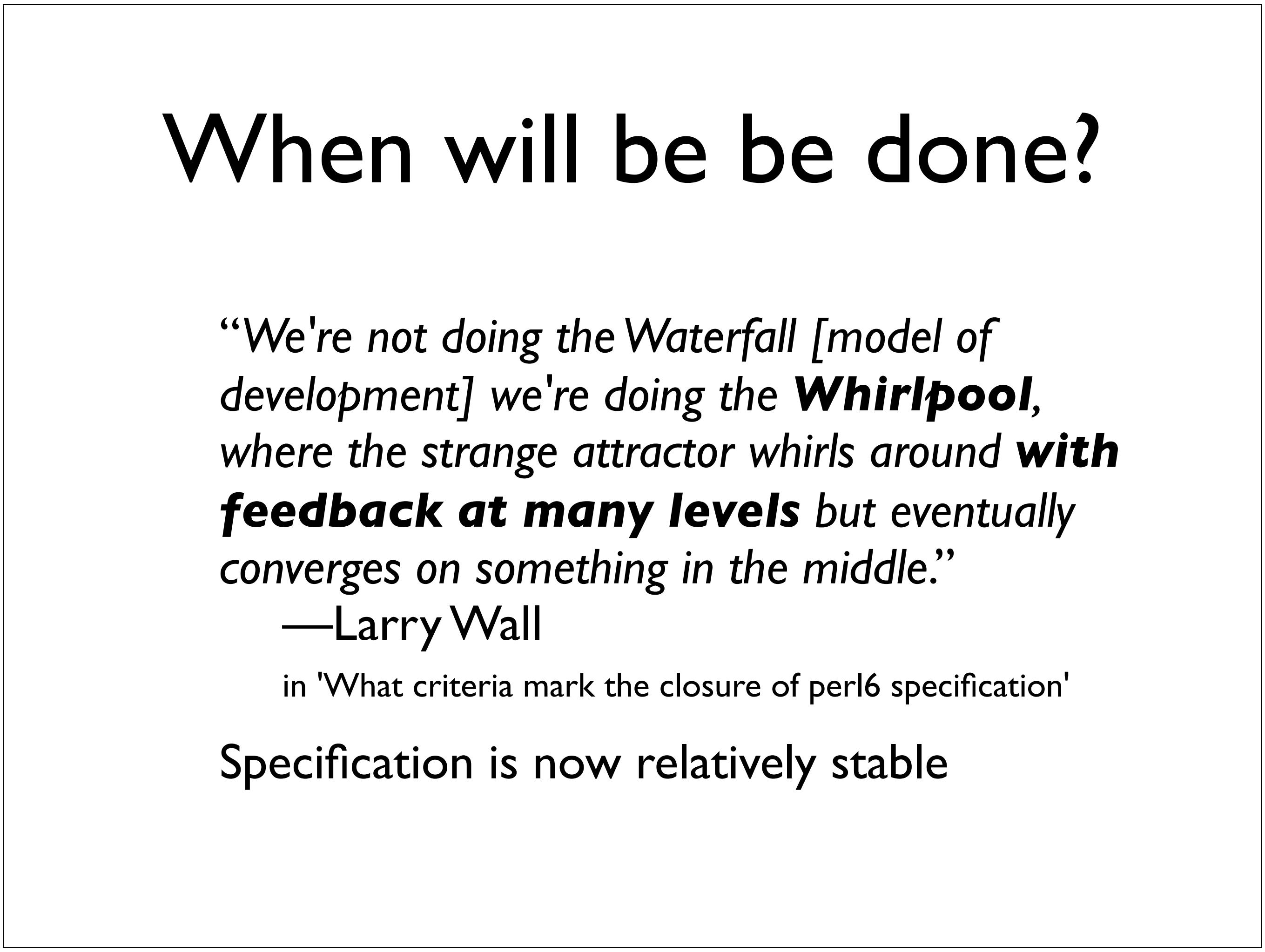
“*If we'd done Perl 6 on a schedule, you'd have it by now. And it would be crap.*”
—Larry Wall

"*do it right*" and "*it's ready when it's ready*"

“*Truly radical and far-reaching improvements over the past few years.*”

It’s given the design team the freedom to look deeply into issues.

To re-balance the design and grammar as it evolved, in ways that wouldn’t be possible after a release.



When will be be done?

"We're not doing the Waterfall [model of development] we're doing the ***Whirlpool,*** *where the strange attractor whirls around* ***with feedback at many levels*** *but eventually converges on something in the middle."*

—Larry Wall

in 'What criteria mark the closure of perl6 specification'

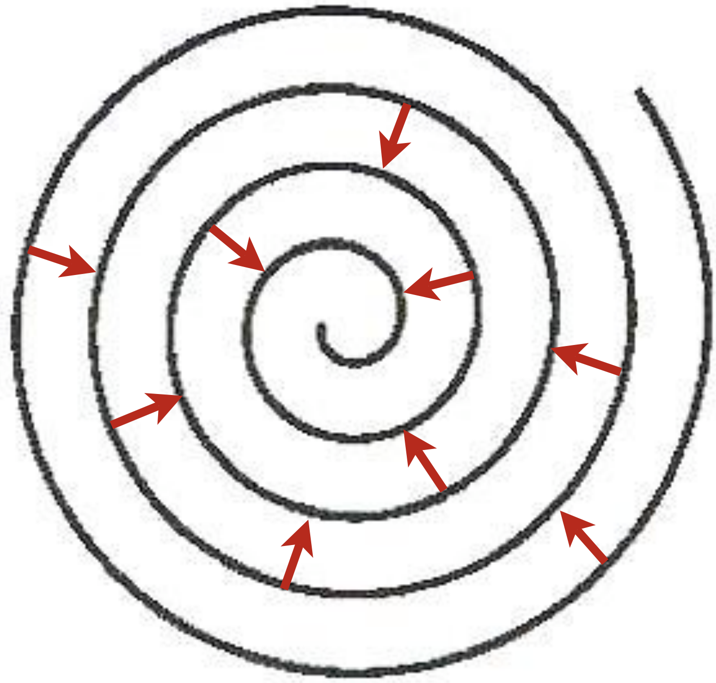
Specification is now relatively stable

"*In other words, a whirlpool sucks, but the trick is to position your whirlpool over your intended destination, and you'll eventually get there, though perhaps a bit dizzier than you'd like.*"

-- Larry Wall, in 'What criteria mark the closure of perl6 specification'

Mid-2007 thru mid 2008:

New features:..............................	6%
Refinements to existing features:.........	16%
Clarifications of semantics:..............	44%
Typo fixes:..............................	35%

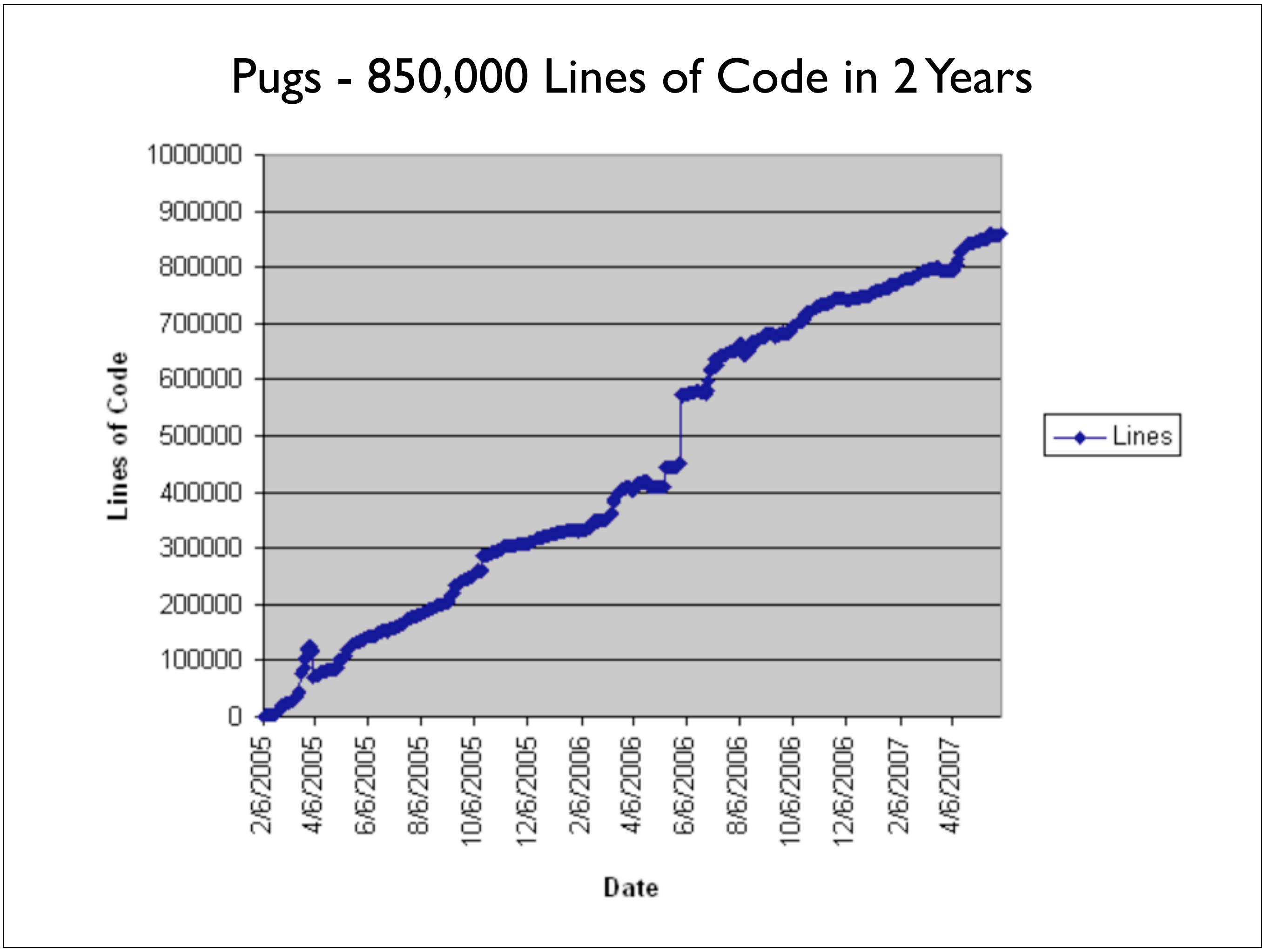


"feedback at many levels"

Multiple implementations

- Perl 6 compilers
  - Pugs - *initial experimentation in Haskel*
  - KindaPerl6 - *perl5 - aiming to self-host*
  - SMOP - *Simple Meta Object Protocol*
  - Rakudo - *built on Parrot Compiler Toolchain*
- Sharing a common test suite

http://www.sgeier.net/fractals/indexe.php

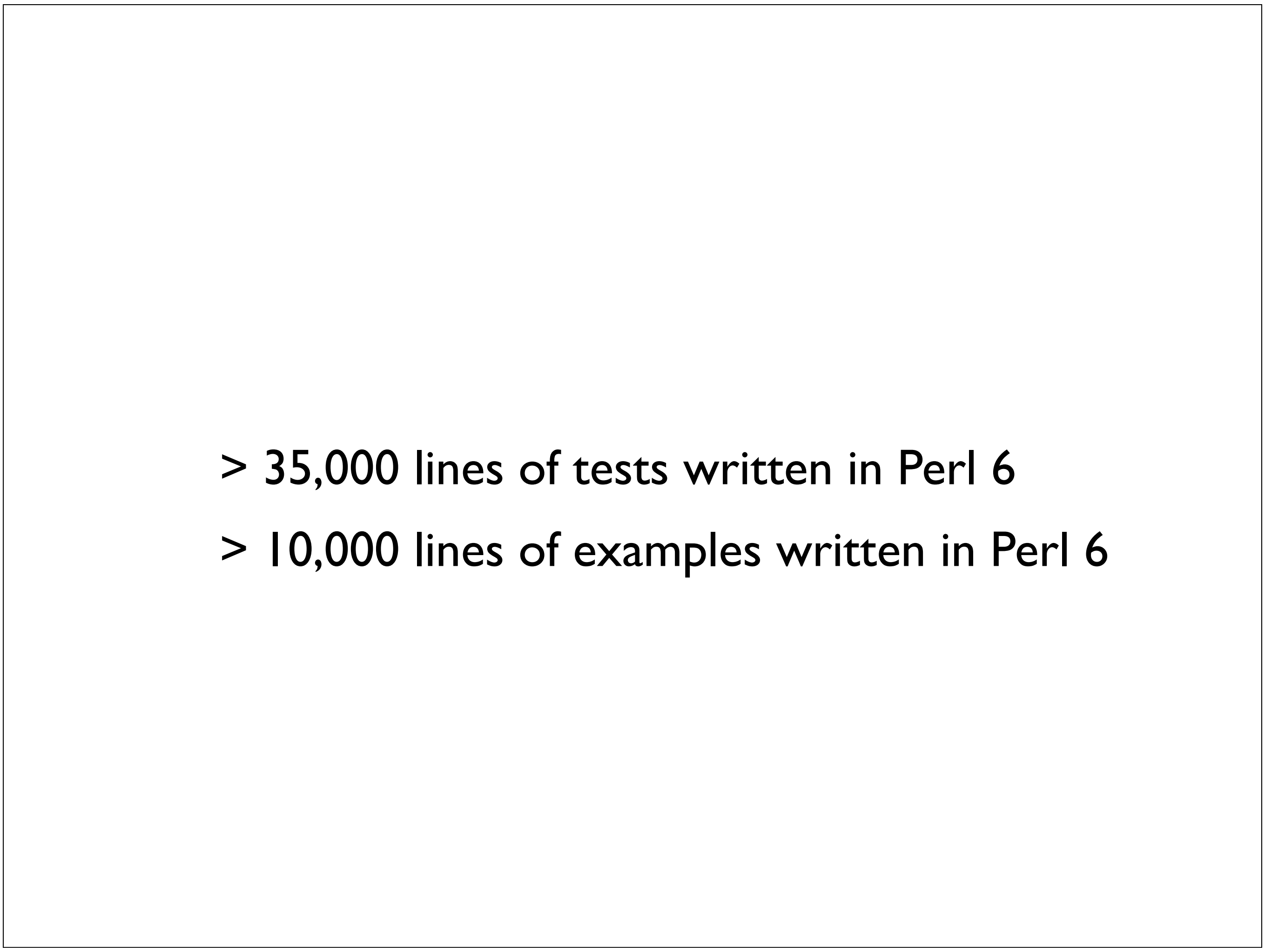


An extraordinary amount of work.

Valuable feedback into the ‘whirlpool’ of design and evolution.

Test suite is the official Perl 6 test suite

“There's no Perl 6 code”



> 35,000 lines of tests written in Perl 6

> 10,000 lines of examples written in Perl 6

http://m19s28.dyndns.org/iblech/stuff/pugs-smokes/pugs-smoke-6.2.13-r19912-linux-normal--1202939667-1239--19328-18024-1304-1909-769-30--0d335bb4cd34fb4ab68763b9f9f7ae15.html

http://svn.pugscode.org/pugs/ext/Muldis-DB/

parrot

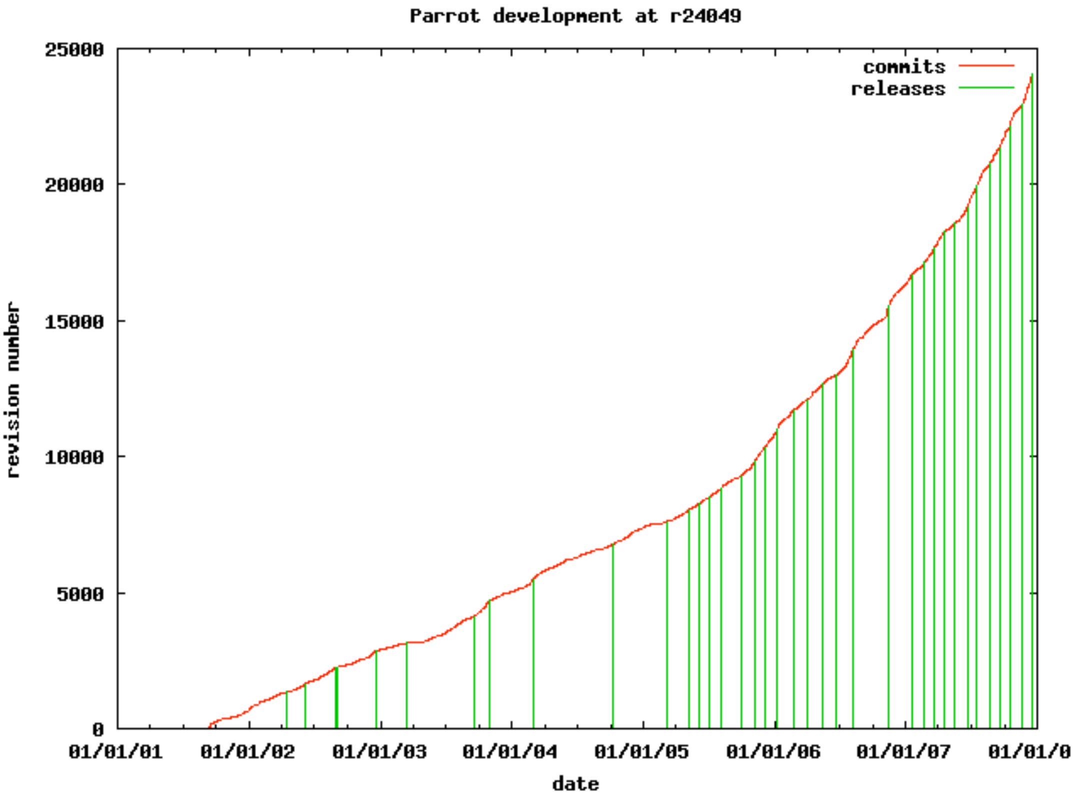
parrot

An advanced virtual machine for dynamic languages

Advanced capabilities with JIT compiling

Already supports over 50 languages
about 20 of which are non-toy

Python, Ruby, PHP, Lua, Lisp, TCL, ...


Parrot development at r24049
commits
releases
revision number
25000
20000
15000
10000
5000
0
01/01/01
01/01/02
01/01/03
01/01/04
01/01/05
01/01/06
01/01/07
01/01/0
date

Parrot Test Suite

11,086 tests in 557 files

53,884 tests if run for all 7 runcores

(in Jan 2008, many more now)

Parrot Compiler Toolkit

Write a compiler in an afternoon!

“Parrot is really quite wonderful. [...] Parrot lets you implement your own languages using Perl 6 rules for the grammar and Perl 6 for the compiler.”
- Simon Cozens

http://blog.simon-cozens.org/post/view/1323

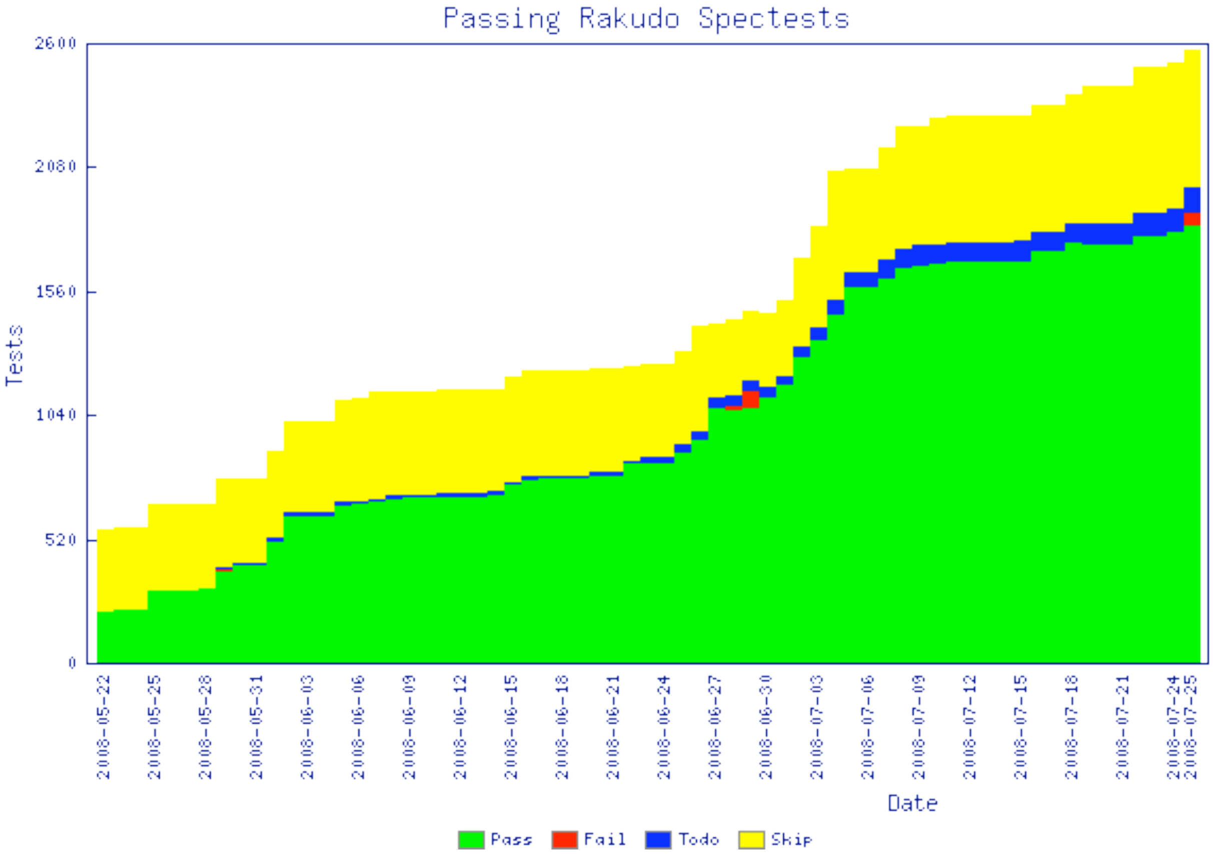
Rakudo

Perl 6 on Parrot

"Rakudo" short for "rakuda-do"

  Japanese for "Way of the Camel"

"rakudo" also means "paradise"


Passing Rakudo Spectests
2600
2080
1560
1040
520
0
Tests
2008-05-22
2008-05-25
2008-05-28
2008-05-31
2008-06-03
2008-06-06
2008-06-09
2008-06-12
2008-06-15
2008-06-18
2008-06-21
2008-06-24
2008-06-27
2008-06-30
2008-07-03
2008-07-06
2008-07-09
2008-07-12
2008-07-15
2008-07-18
2008-07-21
2008-07-24
2008-07-25
Date
Pass
Fail
Todo
Skip

Community Resources

"Awesome community.
Perl people tend to be laid-back and friendly."

perlmonks.org - Meditations and wisdom

use.perl.org - News and blogs

perlbuzz.com - Headlines and articles

comp.lang.perl.* - Newsgroups

Quote taken from comment on http://blog.timbunce.org/2008/02/14/perceptions-of-perl-views-from-the-edge/

Any questions?

http://perlmonks.org
http://search.cpan.org
http://blog.timbunce.org