Going fast slowly

If I count in my source tree, right here and now, Varnish has 100K lines of sourcecode:

```
75619 lines in .c files
18489 lines in .h files
 2625 lines in .py files
  670 lines in .vcc files
  501 lines in .vcl files
```

A little over 20K lines of testcases:

```
21777 lines in .vtc files
```

A little over 20K lines of documentation:

```
22169 lines in .rst files
```

And probably about 5K lines of "misc":

```
1393 lines in .am files
 712 lines in .ac files
 613 lines in .lnt files
```

For the sake of simplicity, let us call it a round 150K total lines [1].

Varnish has been in existence for 10 years, so that's 15K lines per year.

200 workdays a year makes that 75 lines a day.

7.5 hours of work per day gives 10 lines per hour.

Even though I have written the vast majority of the source code, Varnish is far from a one-person project.

I have no way to estimate the average number of full time persons over the last ten years, so lets pick the worst case and say that only two persons were full time.

It follows that there is *no way* average output of those two persons exceeded 5 lines per hour, measured over the ten year history of the project.

Does that number seem low or high to you ?

Anyway, What do programmers do all day?

(Yeah, yeah, yeah, I know...)

Back before the dot-com disaster, people had actually spent considerable time and effort to find out what kind of productivity to expect from a programmer, after all, how could you ever estimate a project without knowing that crucial number?

The results were all over the place, to put it mildly, but they were universally much lower than everybody expected.

With his seminal The Mythical Man-Month, Frederick P. Brooks brought the ballpark estimate "10 lines per programmer per day" into common use, despite everything he wrote in the text surrounding that number arguing for the exact opposite.

With the ultimate focus on quality and correctness, for instance the Apollo and Space Shuttle software, productivity drops to less than one line of code per day per employee.

The estimated upper bound on Varnish productivity is almost an order of magnitude above Brooks ball-park estimate, and another easily ignorable magnitude away from the unrealistic goal of being the same quality as the software for the Space Shuttle.

So we are inside Brooks ball-park, even if a bit on the high side [2],

What took us so long ?

The surprise over the 5LOC/h number is undoubtedly inversely proportional to the age of the reader.

Back when I was a kid I could write 1000 lines in a single sleep-deprived session across midnight [3], but it didn't take that long before I discovered that I had to throw out most if it once I woke up again.

I was 40 years old when I started Varnish and I had 22 years of professional experience, a *lot* of them staring at, and often fixing/improving/refactoring, other peoples source code.

Over the years I came to appreciate Antoine de Saint-Exupéry's observation:

```
Perfection is attained, not when there is nothing more
to add, but when there is nothing more to remove.
```

And eventually I no longer think about code lines as an asset to be accumulated, but rather as an expenditure to be avoided.

When I started Varnish, one of my main personal goals was to make it my highest quality program - ever [4].

This is why Varnish is written in "pidgin C" style and lousy with asserts which don't do anything [5], except clarify programmer intent [6], and in case of mistakes, stop bad things before they get out of hand.

And this is why there are other "pointless overheads" in the Varnish source code, from the panic/backtrace code over the "miniobj" type-safety to obscure hints to Gimpel Softwares FlexeLint product.

Needless to say, it is also not by accident that the 20K lines of testcases exercise over 90% of the varnishd source code lines.

And insisting on doing things right, rather than *"we can fix it properly later"* which is so widespread in FOSS source code [7], is not going to boost your line count either.

But did it work ?

A 10 year project aniversary is a good reason to stop and see if the expected roses are there to be smelled.

We have lots of numbers, commits (10538), bugreports (1864), CVEs (2) [8] or Coverity detections (a dozen?) but It is pretty nigh impossible to measure program quality, even though we tend to know it when we see it.

There are also uncountable events which should be in the ledger, 503s [9], crashes, hair-tearing, head-scrathing, coffee-drinking, manual- and source-code thumbing and frustrated cries of help on IRC.

In the other cup there are equally intangible positives, pats on the shoulder, free beers, X-mas and birthday presents from my Amazon wish-list (Thanks!), and more snarky tweets about how great Varnish is than I can remember.

All in all, the best I have been able to do, to convince myself that I have not *totally* missed my goal, is a kind of "The curious case of the dog in the night-time" observation:

I have never yet had a person tell me Varnish made their life more miserable.

I'll take that.

phk

Footnotes

[1] We can do a better and more precise estimate if we want. For instance we have not typed in the 30 line BSD-2 Blurp *all* 314 times, and upwards of 30% of the rest are blank lines. However, there is no way we can reduce the number by an order of magnitude, in particular not because code that was written and subsequently removed is not part of the base data.

[2] Which is to be expected really: We don't program on punched cards.

[3] And I did. Migrating an oilcompany from IBM mainframes to 16-bit UNIX computers in 198x was an interesting challenge.

[4] Having half the world adopt your hastily hacked up md5crypt with a glaringly obvious, but fortunately harmless, bug will do that to you.

[5] Roughly 10% of the source code lines were asserts last I looked.

[6] I prefer asserts over comments for this, since the compiler can also see them. The good news is, the compiler can also see that they don't do anything so a lot fewer are present in the binary program. Interestingly, a couple of them allows the compiler to optimize much harder. No, I won't tell you which those are.

[7] Only code where that is a bigger problem is phd-ware: Software written as proof-of-concept and abandonned in haste when the diploma was in hand.

[8] Obviously, a high count of CVE's should be a real reason for concern, but there is no meaningful difference between having one, two or three CVE's over the course of ten years. The two CVEs against Varnish were both utterly bogus "trophy-hunter" CVEs in my opinion. (But don't take my word for it, judge for yourself.)

[9] There used to be a link back to the Varnish project on the default.vcl's 503 page, but we removed it after a large national institution in a non-english country showed it to a *lot* of people who clicked on the only link they could see on the page.