Exhausted gods with broken tools

writing software

Julien Kirch

This text has been written for the "A machine for gods" jam organized by Strange Pact. The published version is here.

We were meant to be gods, or so we have been told.

In theory, writing software should feel like being a god: you are in control of a little world where your words have absolute power. You can create and destroy things, make anything unique, or clone it millions of times. Most bad choices can be canceled, you can rewind history and pick another destiny. When you care about something else, everything waits quietly for you to care about it again.

A computer is a little machine that makes you a god.

Coding on the shoulders of elders gods

We even keep long and documented mythologies about the elders that created the tools we use. From sand and mathematics they forged them, or so we've been told.

The socio-economic realities of their stories often go politely unmentioned, so it looks like they were working in a vacuum, following their own whims. There is no Prometheus in our folklore: hubris is seen as a source of inspiration instead of being punished.

Welcome to the grind

But *in reality*, writing software often feels like hell.

We're gods but most of what we do is tedious instead of joyful and our tools are often broken in one way or another.

Instead of trying to create an Eden we often settle for things mostly or barely working. Setting the bar higher means more effort, more fragile results, and more maintenance in the long term.

Writing software often feels like a grind, one day after another, the brokenness eroding your good faith.

I'm curious if someone were to rewrite the Bible with this kind of setup, how would the people react to this kind of god?

Instead of Prometheus you feel like Sisyphus, pushing a misshapen boulder that tries to roll on your toes.

Wait, that's all?

For many people I know, the worst part is the horrifying realization that that's all that programming is about.

After listening to the folk tales, and thinking that creating software would probably feel even better than using it. And their expectations are often based on the programs they prefer, like a video game or a great site.

At first there is so much to learn so everything being a pain seems normal. And then they start to understand how stuff gets done, and understand that, indeed, the problem was not a lack of understanding or knowledge: we've been lied to. Our tools are bad and programming often feels like misery.

And it's the same for even the best and brightest. Thanks to social networks we can see that their work is mostly the same. The only difference seems to be that they are better at dealing with the bad parts, and have enough willpower to keep going.

It's still often a comfortable and well paid job, if you can deal with the mess.

It probably won't improve

The even sadder realization is that you probably can't make a difference: some software can be a joy to use, but creating software is still mostly not.

Productivity definitely improved, but not the overall experience.

You can try to improve things on a very small scale, trying to help people, trying to improve tools or trying to write replacements. Many people do and some of them burn out doing it.

But when you consider the size of the problem, and the fact that it’s not an accident but a direct consequence of the whole setup, including ultra-capitalism and exploitation of free work from some people, it won’t change a thing.

The only escape is to find an isolated niche with good enough tools and people, and don’t wander too far from it, to avoid being caught up in the mess.