Software engineer’s sacred resource

W. Adam Koszek
Koszek ORG
wojciech@koszek.com

2022-10-05T17:47:33Z

For a salesman the critical resource is a resistance to stress: faced with a customer pushback and complaints, he fights against defeat: the purchase rejection. Get the hell out of here. Don’t call anymore. Take me out of your mailing list. Unsubscribe.

To not take things personally, and not give up and reject “no” as an answer, this would cost you a lot, wouldn’t it? After all, it’s your job. That’s what you do eight hours a day. All day. Every day.

I think every professional has their critical resource. What is it for software engineers? This is what I would like to talk to you about today. I think it’s important that you understand it well, because whatever bothers you will result in your next project being delayed or cancelled.

On software work

As an engineer you have your mental workshop. Small, tiny, cozy little place where you wander off to do real work. To get there you deal with a combination of steps, procedures and machines. The thing is: you must be there every day. And if getting there takes effort, it’s a severe problem. You better fix it now, or you’ll start burning out your most important asset.

On patience

You know what it is when you loose it. It stands for “the capacity to accept or tolerate delay, trouble, or suffering without getting angry or upset”. Notice that list: delay, trouble, and suffering. These may be the three words that best describe software engineering as well. Go ahead and read this definition again, but say “software engineering” at the beginning. It sounds about right? It does makes sense, doesn’t it? Software engineering and patience are cousins. Lets go through this list together.

On Latency

Meet delay, patience's assassin. Patience fears delay, thus where patience is, patience is not. In a software projects more understood way is "latency":

If it's long, it frustrates you.

Managers understood it, so this example rarely happens anymore. As a new hire at the company you'll get a very fast computer, very fast Internet connection. It's in fact hard to mae

The easier way to understand delay is The reason why you wander off with your mind is that is that

Most of software projects calculate delay as "How much time did the software engineers wait for their dependencies to get done". What this way of measuring time aspect doesn't count is

When you write software for fun, as a hobby or for the project which you do in your free time, the problem of patience as a limited resource may never come up.

Waiting

Microsoft stated Windows source code is 250GB. First clone of this code repository from their server to your computer takes 12 hours, the pull which synchronized the changes other people made takes 4 hours and getting the status of your changes takes 10 minutes. It's barely usable environment for a developer.

My first job writing software was with Xilinx. Our disk volume was mounted from the network through NFS protocol. It worked well for