Programming best practices and other stuff about the barbarians

(the first day)

The first SW principle

“F**k you my code is better than yours”

Principle that is always valid no matter contexts, technologies or skills.

(2016 M. Merola)

The naming convention and the programming rules

The first step before start a new project is to choose (and to write) clear rules

Order first

- Naming conventions
- Programming rules

My hidden hint

https://google.github.io/styleguide/cppguide.html

The comments and the names

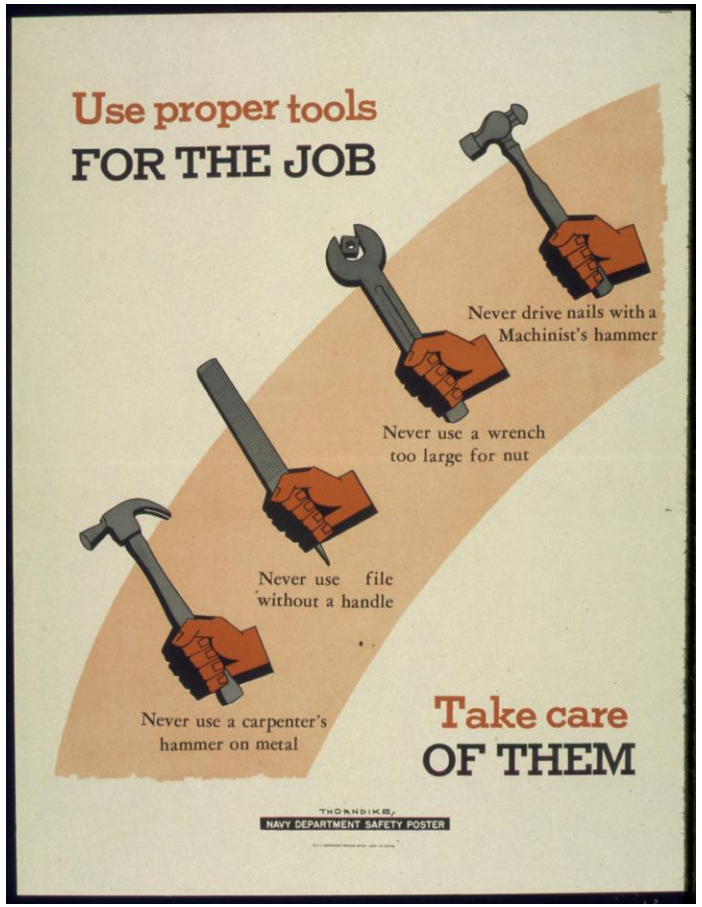
To comment or not to comment?

The key: choose right names

Note interesting by Joel:
the infamous Hungarian naming convention
https://www.joelonsoftware.com/2005/05/11/making-wrong-code-look-wrong/

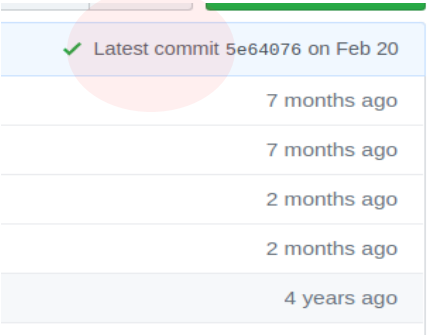
Tools

Use proper tools for the job



Check new technology with care

My hidden hint



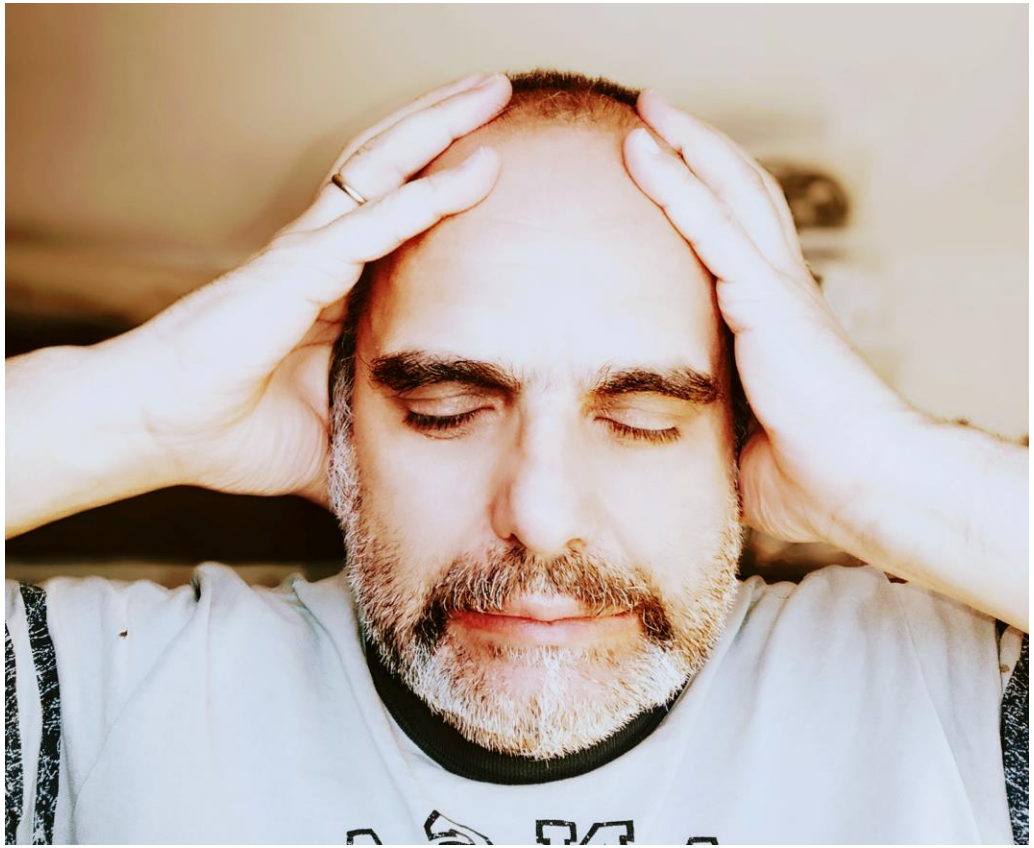
The second SW principle

“Compila quindi funziona”

Again, principle that is always valid no matter contexts, technologies or skills.

(2007 L. Nardella)

Software quality I
The Coke problem(Siemens 2010, OEE market)

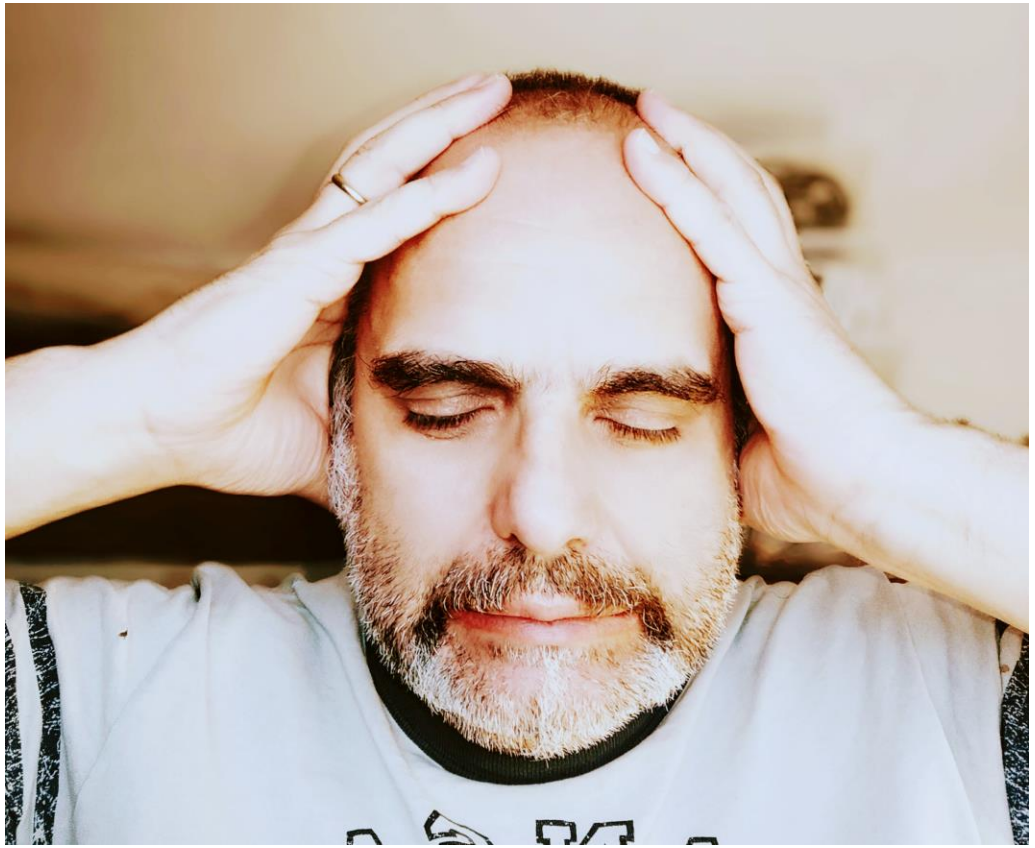


The third SW principle

“It works on my machine”

Again, principle that is always valid no matter contexts, technologies or skills.

Software quality II
The delivery problem (Siemens 2012, OEE market)



- Unit test
- Integration test
- System test
- Agile group collaboration

https://www.fingent.com/blog/collaborating-testers-and-developers-in-hybrid-agile-environment

My hidden hint

Create mix test/devel group

Debugging time

Debugger VS Logger

Simple development

Complex development
Production

My hidden hint → Learn to use good logging library

Code your own wheel

Close code wheel

Check licenses
Check library lifetime and OS support
Check cost
Test functionality
Test code efficiency
Test code fault
Test and test

Open code wheel (GitHub)

Check licenses
Check library lifetime and OS support
Test functionality
Test code efficiency
Test code fault
All is under my control (*I hope*)

My wheel

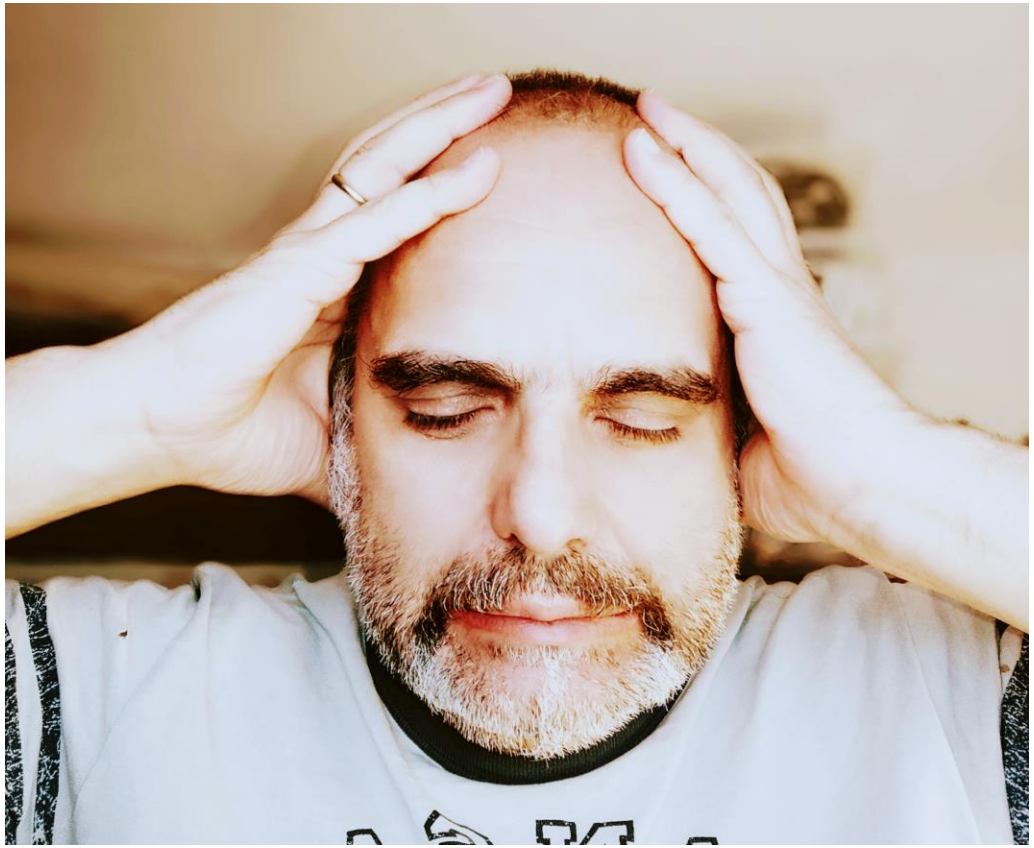
Never use it, but you can learn
All is under my control

The Mongolian horde

Some delay occur → PM add more programmers to the team → Usually adding people to a delayed software project delays it even more

My hidden hint → Add people only if you have a good planning

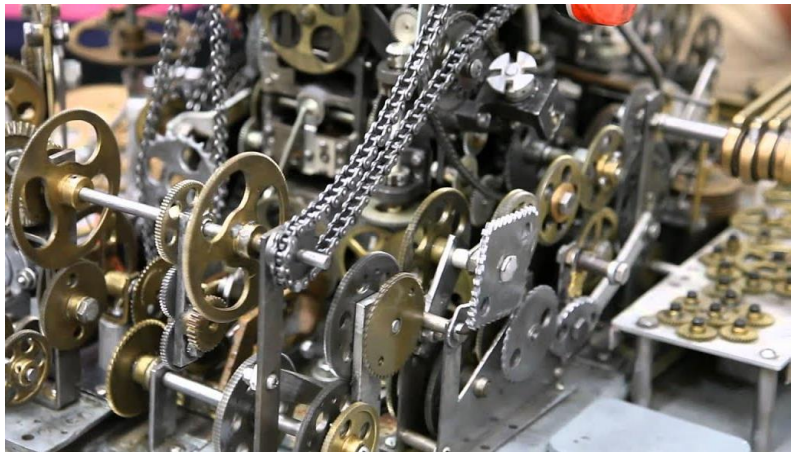
Software quality III
The dynamometric wrench problem (Atlas-copco 2017)



Always think generic

(not only what you guess they can ask you in the future)

How can a banana fit in a gearbox?

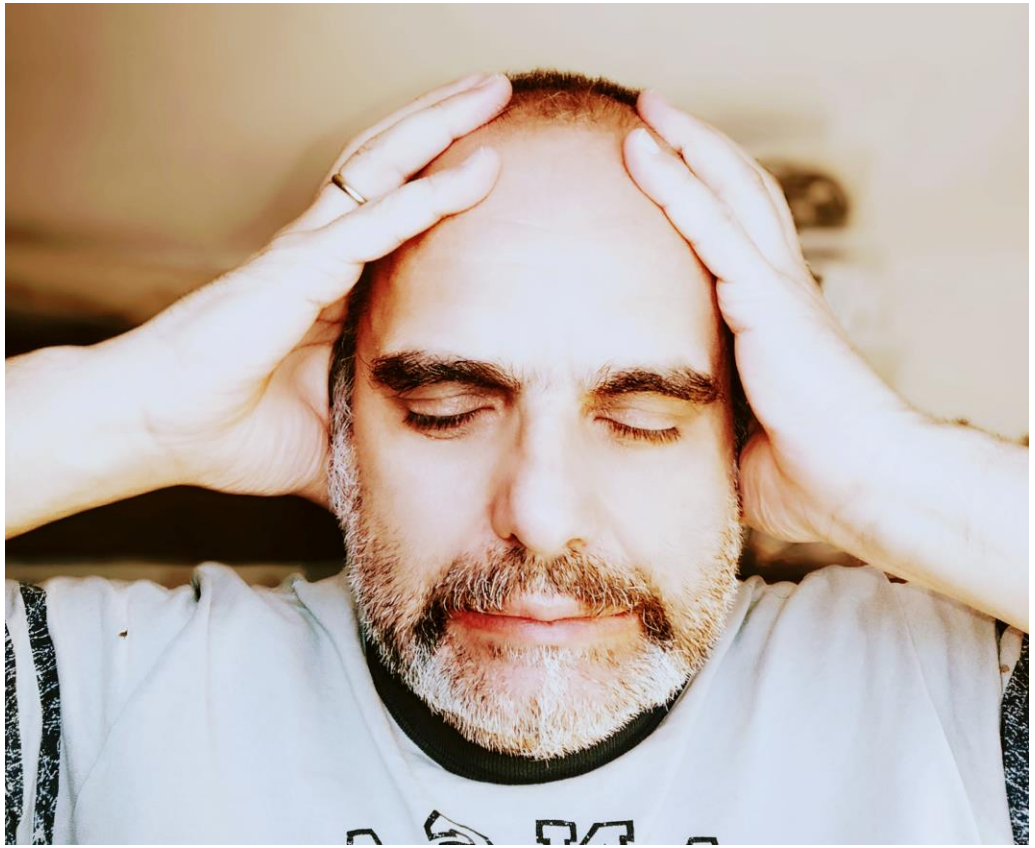


Too late now

My hidden hint

Add or remove complexity?

Software quality IV
The Tetra nightmare (Marconi 1998, the TETRA standard with Nokia)



Refactoring (the code smells)

Why and When

To keep your code clean, **the code smells like a wet camel**
To save yourself time and **money** in the future
To reduce your technical **debt**

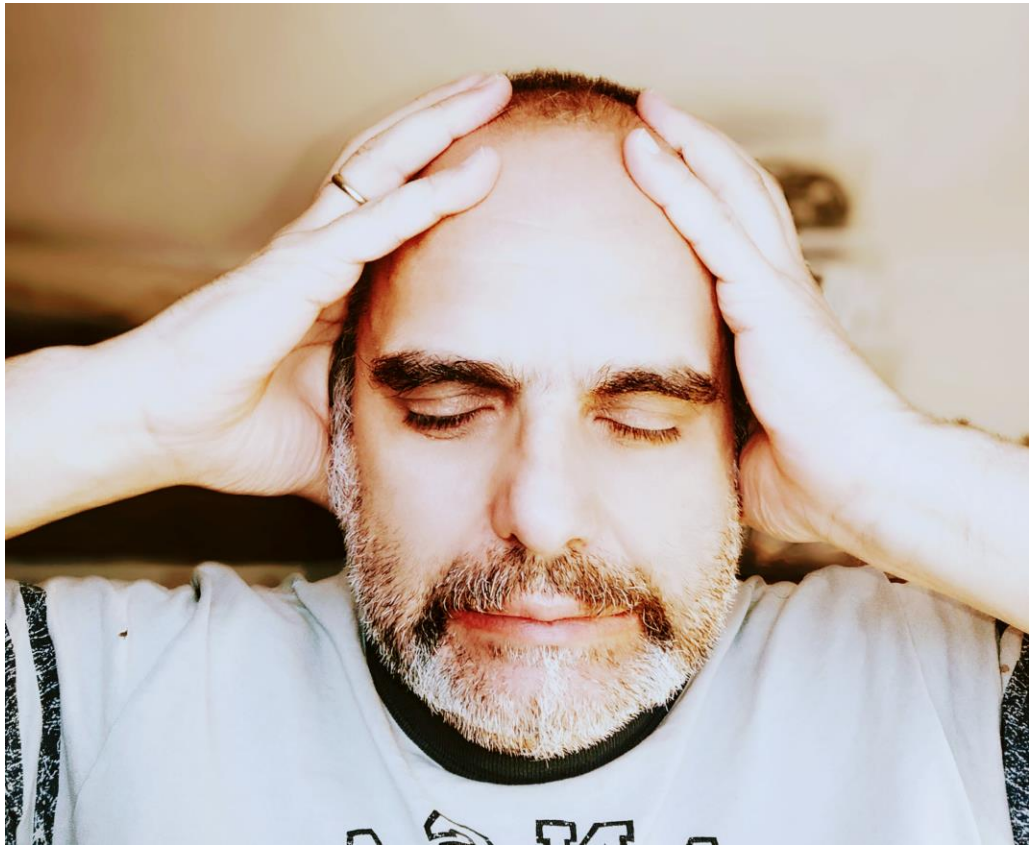
Why not

You have some new features to develop

You don't have tests

My hidden hint → Only if you can test the software

Software quality V
100K Dollars(Ratioconsulta 2009, the telephone calls market)



Code optimization

(Clear code first)

Optimization code rules(Michael A. Jackson):

1. Don’t do it.
2. *(For experts only!)* Don’t do it *yet*.

Why and When

If you really have performance problems

If you want to add some bugs to your code

Why not

For joke

Someone has already done this (stl,boost ...)

Be careful

Use the right tools, don't waste your time

Writing less code is not optimization!

My hidden hint → Check where to optimize