Reversed Tests Pyramid

Wiktor Żołnowski

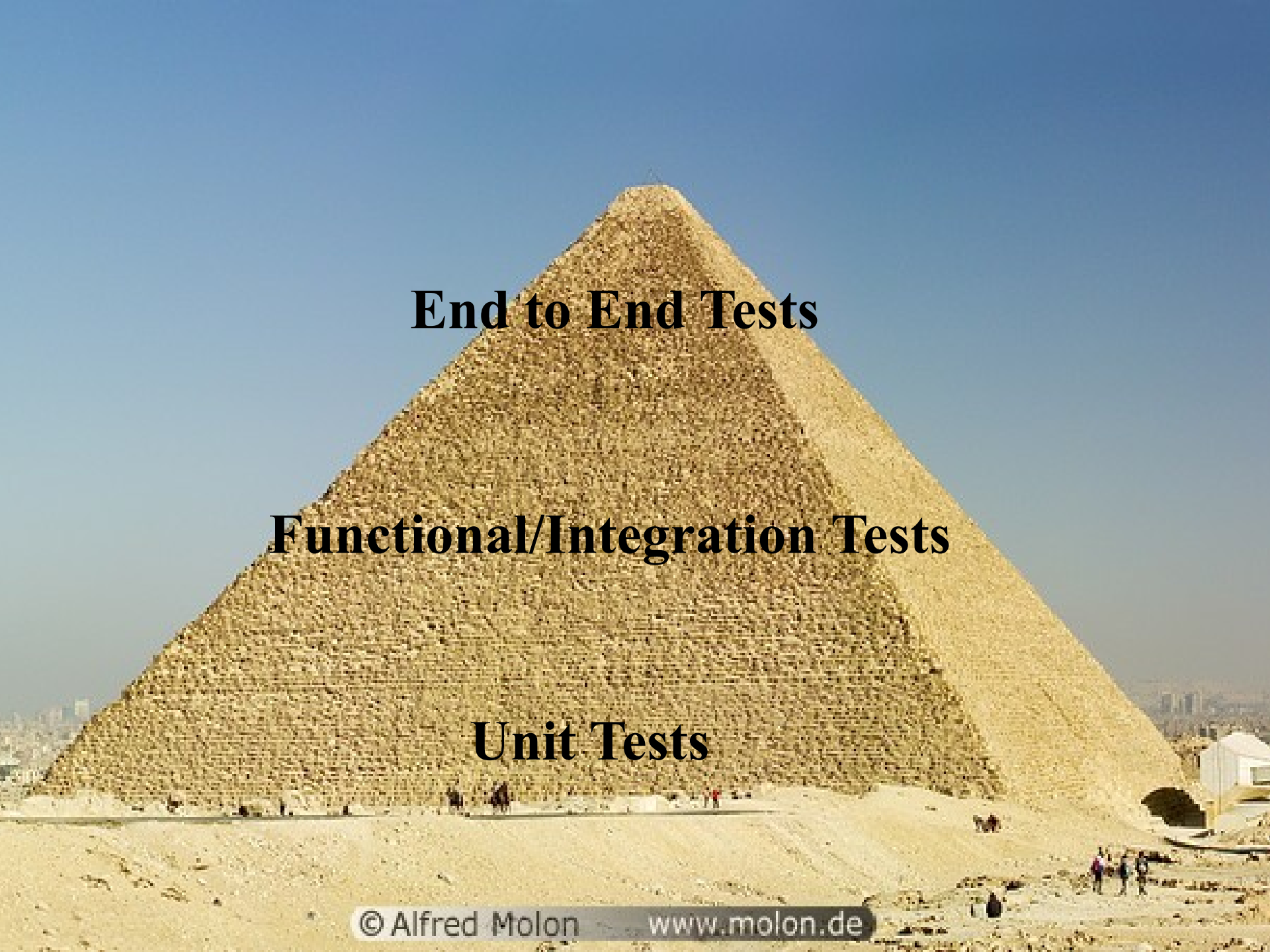
@streser

http://www.agileszkolenia.pl

http://codesprinters.com

Can you imagine perfect software?

CodeSprinters
Experts in Agile


End to End Tests
Functional/Integration Tests
Unit Tests
© Alfred Molon
www.molon.de

It would be perfect to work with perfect software every day...

But...

Our everyday work looks a little bit different...

CodeSprinters
Experts in Agile

It's called...

Legacy Code

CodeSprinters
Experts in Agile

First of all...

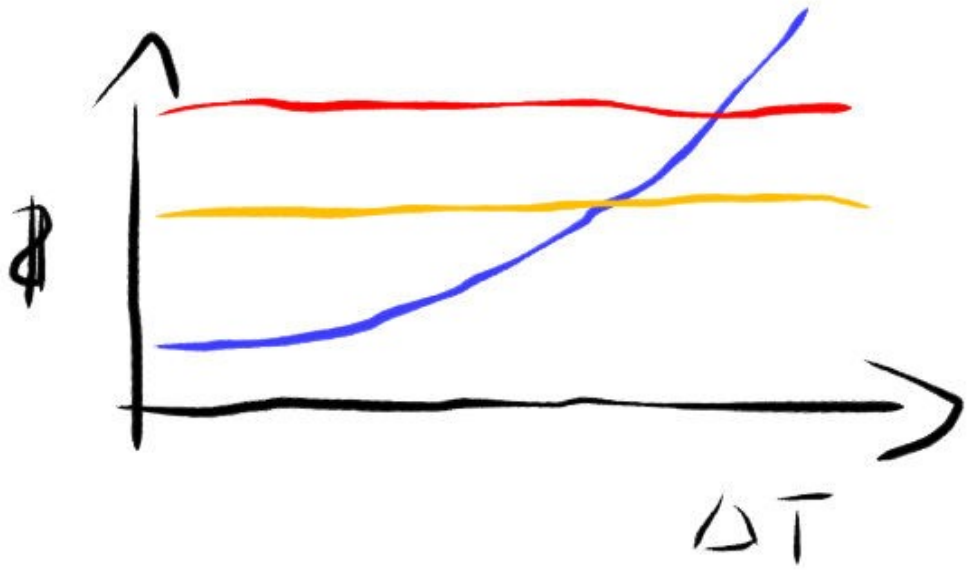
How did we get to this point?

CodeSprinters
Experts in Agile

It's all because of the Technical Debt

CodeSprinters
Experts in Agile

Is it possible to pay Technical Debt back?

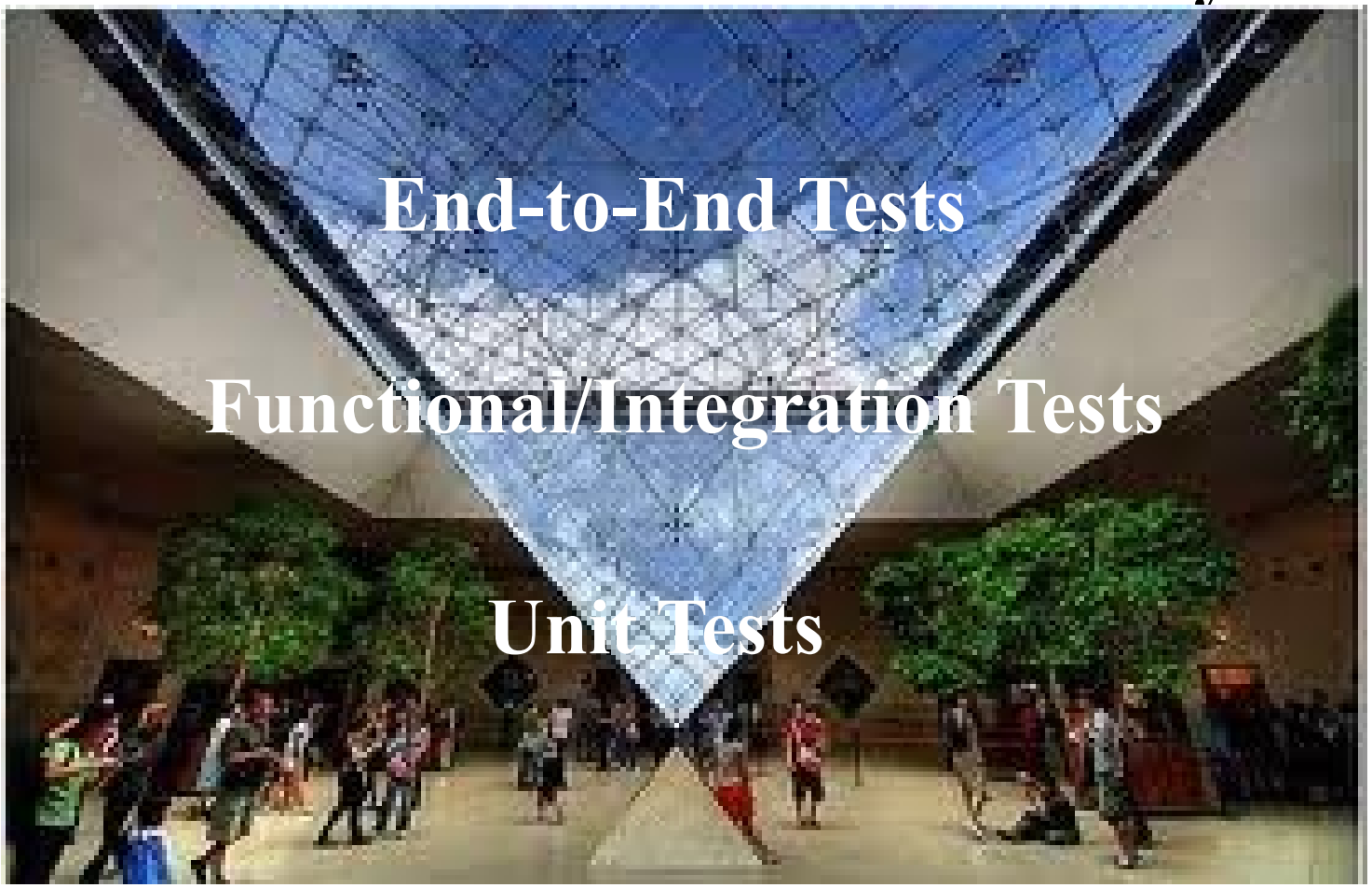


Technical Debt is evil!

- Success in Software Development is something which is not continuous...
- Success is state that you can achieve but also lose very fast if you can't respond to changes fast enough...

CodeSprinters
Experts in Agile

Reversed Tests Pyramid



CodeSprinters
Experts in Agile

- High level tests gives you courage to refactor your code...
- Now you can write some Unit Tests...
- Do it step by step...
- Cover your functionality with Unit Tests...

But.. There are few reasons why you shouldn't reverse tests pyramid

- End-To-End tests are too long...

- End-to-end tests are difficult to maintain...
- If we need end-to-end tests we are probably doing something wrong with our architecture...

So it's all about reversing back our tests pyramid.

But...

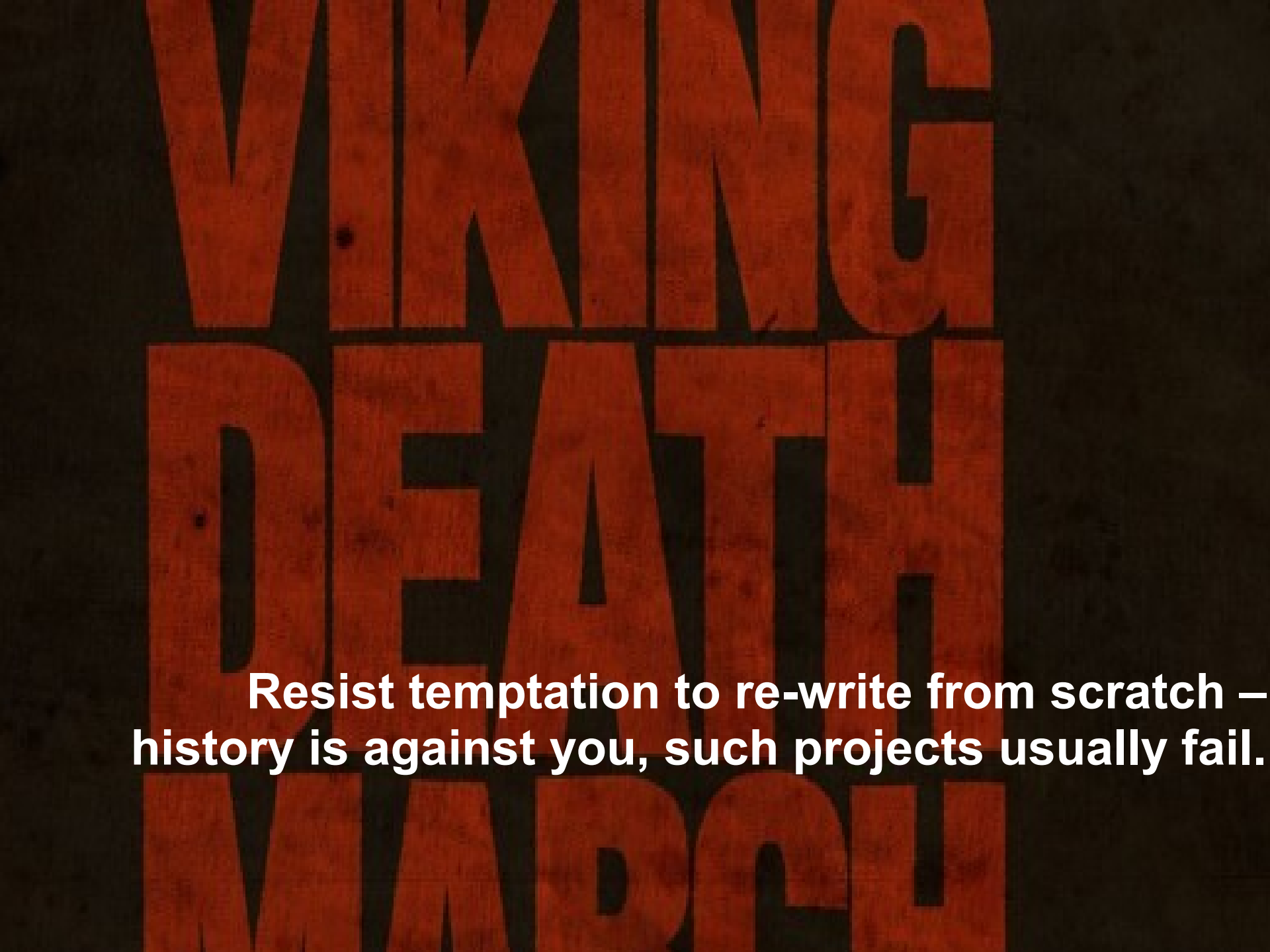
- Remember that creating reversed tests pyramid and reversing it back will take some time...
- You need to deal with it if you want to pay back your technical debt...

CodeSprinters
Experts in Agile

Few final thoughts...

Keep your technical debt as low as possible and try to pay it back every time you can. For example use your slack time for that!

Beware of refactoring just for refactoring!



Resist temptation to re-write from scratch – history is against you, such projects usually fail.


Remember to always remove your (duplicated) tests!

Software quality in many cases could be understood as ability to introduce changes into software!

Wiktor Żołnowski
www.agileszkolenia.pl

Questions?

CodeSprinters
Experts in Agile