Why We need Clean Code?

Clean Code

- Have you written bad code ?
- Why ?
  - Were you trying to go fast? Were you in a rush? Perhaps you felt that you didn’t have time to do a good job?

Clean Code

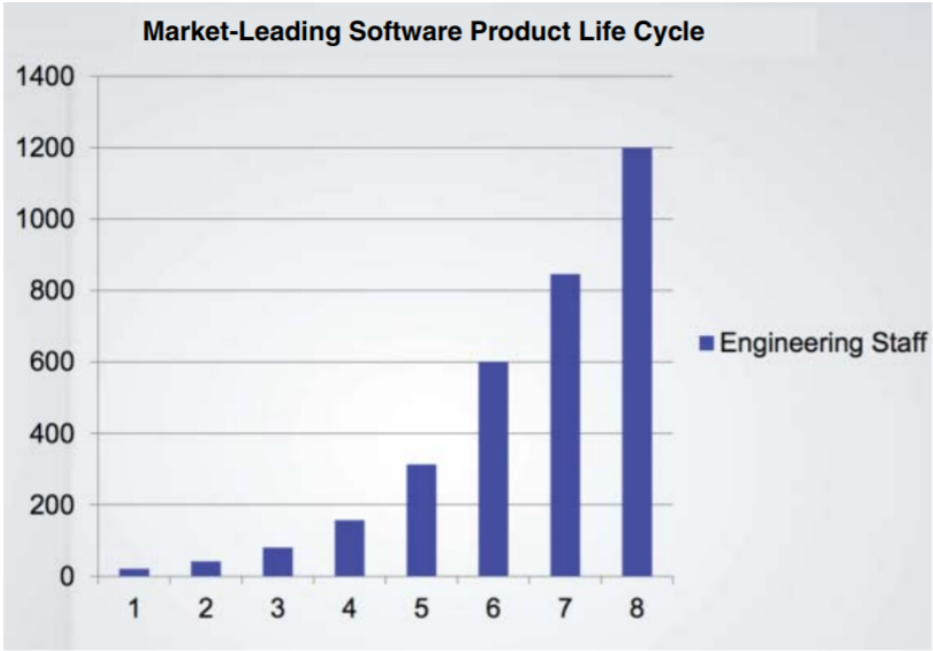
- LeBlanc’s law: Later equals **never**.

The Total Cost of Owning a Mess

- The mess builds
- The productivity of the team continues to decrease, asymptotically approaching zero.
- Management does the only thing they can; they add more staff to the project in hopes of increasing productivity.

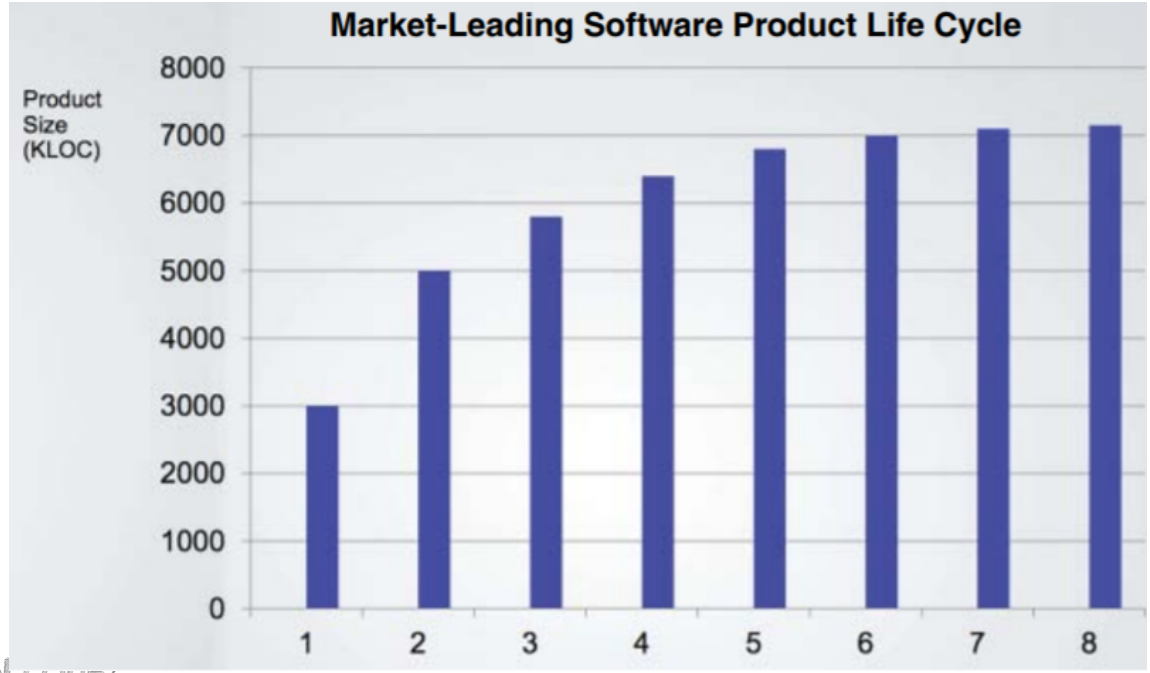
Case Study

- Growth of engineering staff for each cycle.



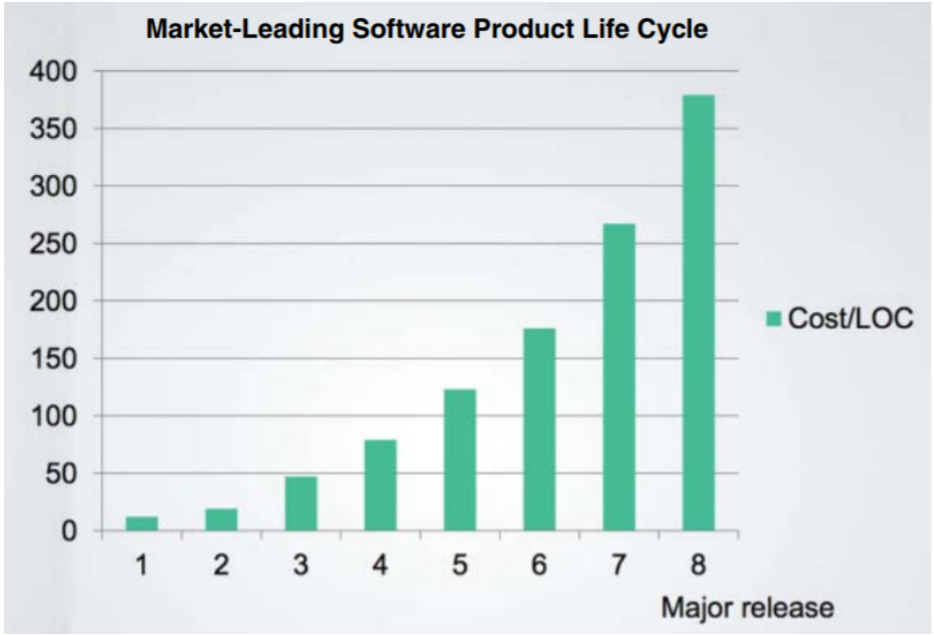
Case Study

- Company productivity measured in KLOC.



Case Study

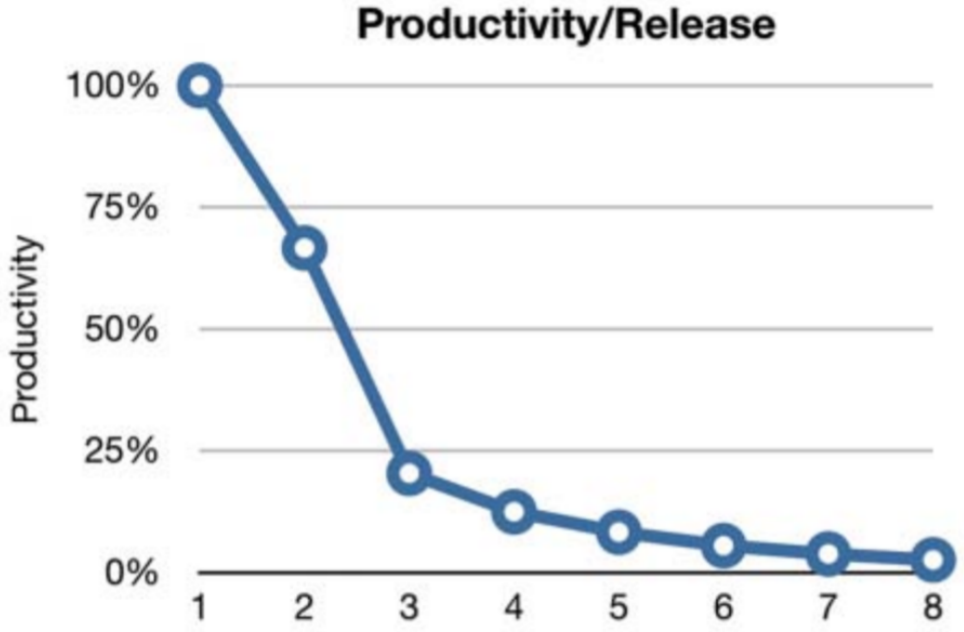
- Cost of LOC.



The Signature of a Mess

- When systems are thrown together in a hurry
- When the sheer number of programmers is the sole driver of output
- When little or no thought is given to the cleanliness of the code or the structure of the design
- You can bank on riding this curve to its **ugly end**.

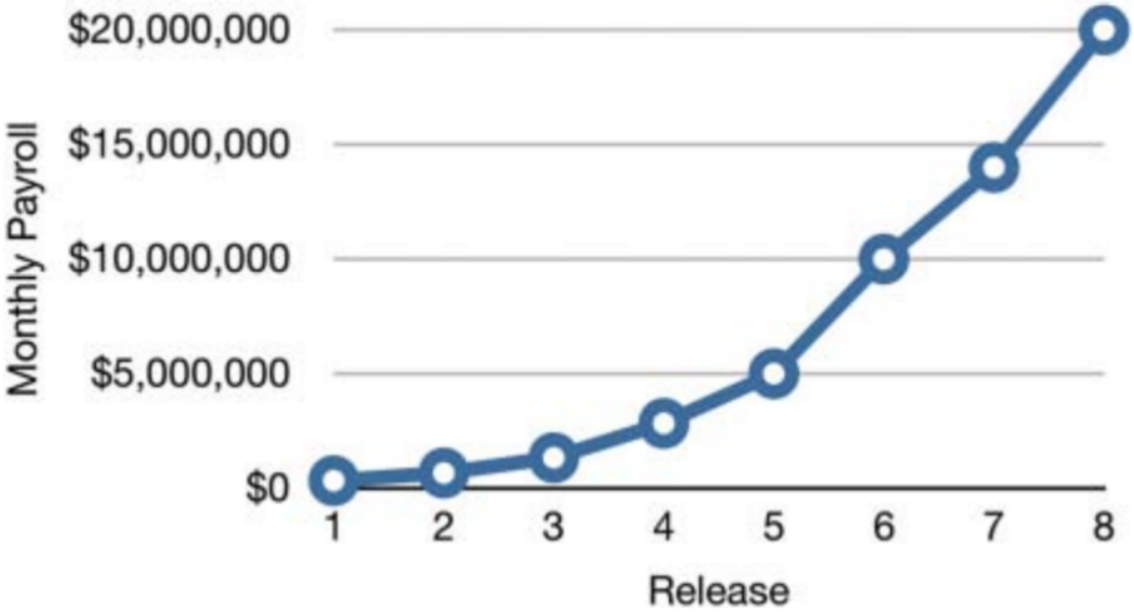
The mess curve from developers view



The mess curve from developers view

- From the developers' point of view, this is tremendously frustrating
  - Everyone is working hard.
  - Despite all their heroics, overtime, and dedication, they simply aren't getting much of anything done anymore.
  - All their effort has been diverted away from features and is now consumed with managing the mess.

The Executive View



What went wrong ?

- Nearly 2600 years ago, Aesop told the story of the Tortoise and the Hare. The moral of that story has been stated many times in many different ways:
  - Slow and steady wins the race.
  - The race is not to the swift, nor the battle to the strong.
  - The more haste, the less speed.

Lies and Developer

- I can clean it up later; We just have to get to market first!

- Writing messy code makes me go fast in the short term, and just slows me down in the long term.

Conclusion

- Making messes is always slower than staying clean, no matter which time scale you are using.
- The only way to go fast, is to go well.
- The only way to reverse the decline in productivity and the increase in cost is to get the developers to stop thinking like the overconfident Hare and start taking responsibility for the mess that they've made.

The Art of Clean Code?

- Do you ask yourself: “How do I write clean code?”

- It’s no good trying to write clean code if you don’t know what it means for code to be clean!

The Art of Clean Code?

- Writing clean code is a lot like painting a picture.
  - Most of us know when a picture is painted well or badly. But being able to recognize good art from bad does not mean that we know how to paint!
  - A programmer without “code-sense” can look at a messy module and recognize the mess but will have no idea what to do about it.

What Is Clean Code?

- Bjarne Stroustrup, inventor of C++ and author of The C++ Programming Language:
  - I like my code to be elegant and efficient. The logic should be straightforward to make it hard for bugs to hide, the dependencies minimal to ease maintenance, error handling complete according to an articulated strategy, and performance close to optimal so as not to tempt people to make the code messy with unprincipled optimizations. Clean code does one thing well.

What Is Clean Code?

- Ron Jeffries, author of Extreme Programming Installed:
  - Runs all the tests;
  - Contains no duplication;
  - Expresses all the design ideas that are in the system;
  - Minimizes the number of entities such as classes, methods, functions, and the like.

Of these, I focus mostly on duplication. When the same thing is done over and over, it's a sign that there is an idea in our mind that is not well represented in the code. I try to figure out what it is. Then I try to express that idea more clearly. Etc.

The Boy Scout Rule

- Do you know it ?

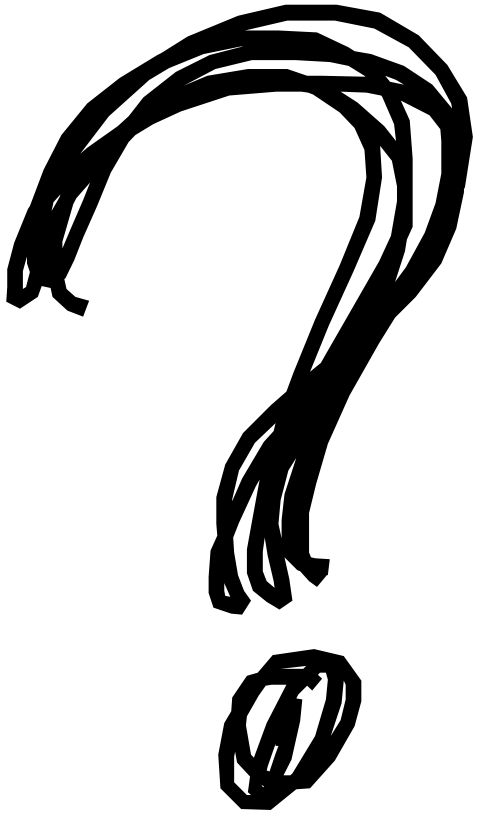
Things you can start doing now!

- Care About Your Craft
  - There is no point in developing software unless you care about doing it well.
- Think!
  - In order to be good developer, think about what you're doing while you're doing it. This isn't a one-time audit of current practices—it's an ongoing critical appraisal of every decision you make, every day, and on every development. **Never run on auto-pilot.**

Things you can start doing now!

- Take Responsibility
- Provide Options, Don't Make Lame Excuses
- Don't Live with Broken Windows


СВ. КЛИМЕНТ ОХРИДСКИ



THANK YOU