Agile
A Software
Engineering
Discipline

1945

- O(1) programmers
- Writes code in binary (base 32 reversed)
- Invents subroutine, stack, floating point

1945

We shall need a great number of mathematicians of ability because there will probably be a good deal of work of this kind to be done

1945

*One of the difficulties will be the maintenance of an appropriate **discipline**, so that we do not lose track of what we are doing.*

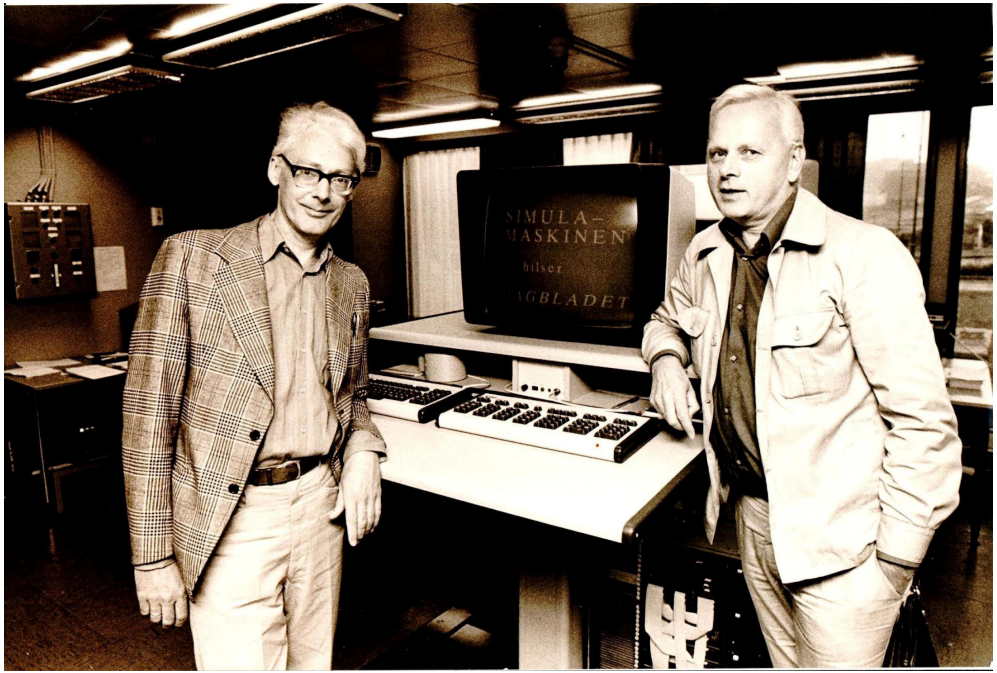
1950/60s

- O(100K) programmers
- Where did they come from?
  - No CS degrees
  - Drawn from best and brightest in Engineering, Science, Mathematics
  - Experienced, displiciplined, mature professionals
  - Understood management, schedules, deadlines etc...
- They performed miracles!

1950/60s

Ole-Johan Dahl and Kristen Nygard

- Mathematics
- Simula-67
- Object Oriented Programming



1950/60s

Ken Thompson and Dennis Ritchie

- Mathematics
- C
- UNIX
- Regular expressions
- Go
- Hacker movement (cathedral and the bazaar)

1950/60s

Edsger W. Dijkstra

- Theoretical Physics / Mathematics
- Structured programming (no GOTO)
- Dijkstra’s algorithm (graph search, routing)
- Mutual exclusion (concurrency, distributed systems)

1950/60s

Margaret Hamilton

- Mathematics
- Lead developer on Apollo missions
- Mother of “software engineering”

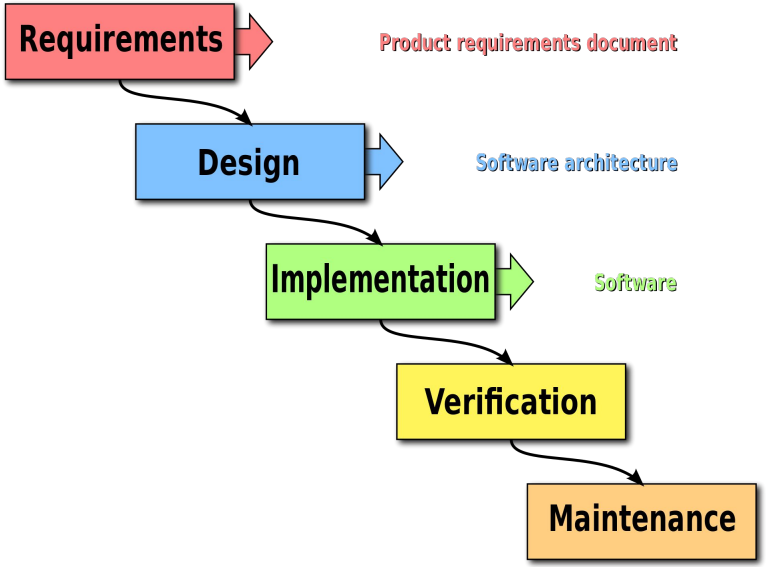
It was a memorable day when one of the most respected hardware gurus explained to everyone in a meeting that he agreed with me that the process of building software should also be considered an engineering discipline, just like with hardware. Not because of his acceptance of the new 'term' per se, but because we had earned his and the acceptance of the others in the room as being in an engineering field in its own right.

1970/80s

- O(1M) programmers
- Where did they come from?
  - CS/EE graduates
  - Young, male aka “brogrammers”
- What they lack in discipline they make up for energy
  - Crazy hours
  - Hyper focused
  - cheap!

1970/80s

- Discipline from above aka management
- Waterfall model
- Structure, process heavy, top down
- Still used today!



1990s

- Original cohort of disciplined professionals retire
- First wave of career programmers come of age (40+)
- Need for change a rebellion starts

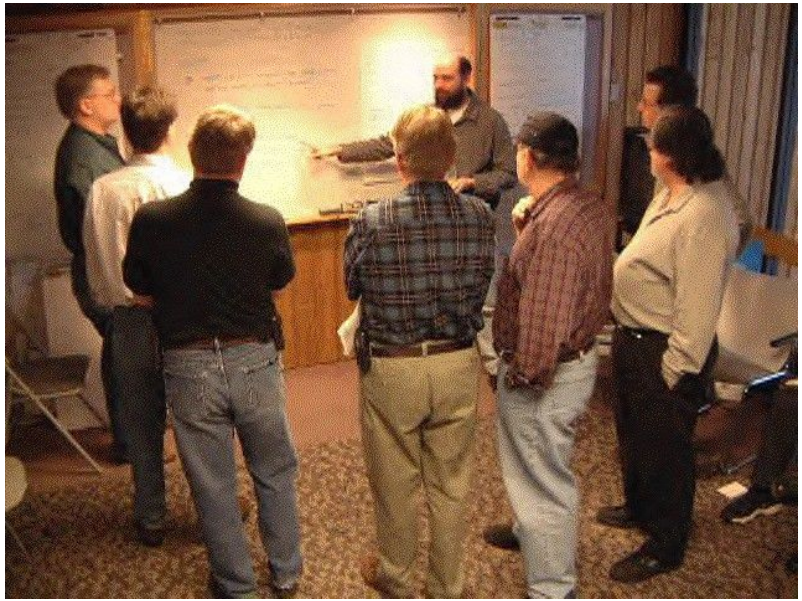
1990s


Manifesto for Agile Software Development
We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
That is, while there is value in the items on
the right, we value the items on the left more.

1990s

Agile requires **discipline**

- Working in fixed time boxes
- Estimating in relative units (points)
- Customer communication
- Continuous integration
- Collaboration (pair/mob programming)
- Test driven development (TDD)
- Technical debt and refactoring
- Simple design



1990s

“Uncle” Bob Martin

- Disciplines, not process steps
- Promises you make, not tasks you follow
- An issue of ethics and morality
  - *I will write tests*
  - *I will refactor*
- Agile is about
  - (Technical) Discipline
  - Professionalism
  - Craftsmanship

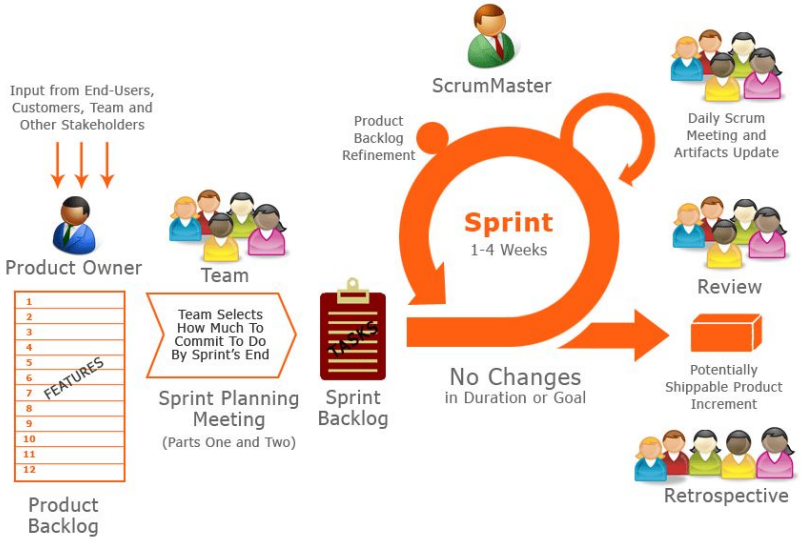
The future of programming
https://www.youtube.com/watch?v=ecIWPzGEbFc

1990s

Business disciplines

- Kanban, Lean, Six Sigma
- Certifications
- Project managers

Do not lose track of what we are doing...



1990s

Martin Fowler (Refactoring)

- Flaccid Scrum: An efficient business discipline coupled to an **undisciplined** engineering team will very rapidly make a mess

1990s

Should we write tests?

No

Business can not approve or endorse technical disciplines. When we ask the business these questions, we are shedding risk we are not willing to take.

As professionals, we must own this risk.

1990s

The agile split

- Agile: Business practices
- Craftsmanship: Technical practices

Kent Beck (creator of XP)

The goal of agile was to heal the divide between business and programming.

Today

Software engineers rule the world. And the world depends on us.

We must define our profession

Choose our practices and disciplines

Set some moral and ethical principles

Heal the divide between business and programming

Before the government starts to regulate us!

1945

*One of the difficulties will be the maintenance of an appropriate **discipline**, so that we do not lose track of what we are doing.*

Let’s not disappoint him!