CS1530, Lecture 19: Software Craftsmanship

Bill Laboon

Software Engineer != (Assembly Line Worker || Salesperson || Soldier)

- Hard to quantify what you do
- Not repetitive – almost by definition, every challenge is different
- Often your manager will not understand what you are doing
- Being great is subjective and UP TO YOU

Intellectual Humility

"The competent programmer is fully aware of the strictly limited size of his own skull; therefore he approaches the programming task in full humility, and among other things he avoids clever tricks like the plague."

-Edsgar Dikjstra, "The Humble Programmer"

https://www.cs.utexas.edu/~EWD/transcriptions/EWD03xx/EWD340.html

Intellectual Humility

“[O]ne programmer places a one-line program on the desk of another and either he proudly tells what it does and adds the question “Can you code this in less symbols?” —as if this were of any conceptual relevance!— or he just asks “Guess what it does!”.

… I am sorry, but I must regard this as one of the most damning things that can be said about a programming language.”

-Edsgar Dikjstra, “The Humble Programmer”
https://www.cs.utexas.edu/~EWD/transcriptions/EWD03xx/EWD340.html

Intellectual Humility

“Everyone knows that debugging is twice as hard as writing a program in the first place. So if you're as clever as you can be when you write it, how will you ever debug it?”

-Brian Kernighan, “The Elements of Programming Style”

What is Intellectual Humility?

- It is intellectual *honesty*
- Don’t pretend to be an expert if you are not – even if you are, understand your weak points
  - γνῶθι σεαυτόν – Know thyself!
- Readily admit your mistakes, and be prepared to throw away your work if you did it incorrectly

What is Intellectual Humility?

- Be realistic in your status reports and estimates
  - UPOD – Underpromise and overdeliver
  - Estimation is hard! But programmers tend to be better at it than managers.
  - Do not be afraid to “manage up”

Programming For People

- Write programs for people first, computers second
  - Engineering time is expensive, computer time (in general) is not (of course there are exceptions)
- Making code readable by humans means increased:
  - Comprehensibility
  - Reviewability
  - Fewer errors
  - Easier debugging
  - Less development time
  - Better external and internal quality
  - Easier to modify in the future

Don’t Fall For Gurus (Not Even Me)

- Read and understand others’ opinions, but…
- Make up your own mind.
- Someone offering you a silver bullet is offering you snake oil.

Welcome to a Dynamic Field

- Interested in working with mechanical watches?
- The last major change to how most mechanical watches work was in 1930, with the Rolex Oyster Perpetual's improvement on Harwood's auto-winder
- There have been incremental improvements (and of course electronic/quartz watches are a whole other story), but besides some of the materials, nothing in a modern mechanical watch would surprise a watchmaker from, say, 1935

Software Engineering != Watchmaking

- Would anything in a modern computer surprise our hypothetical counterpart from 1935?

Zuse Z1

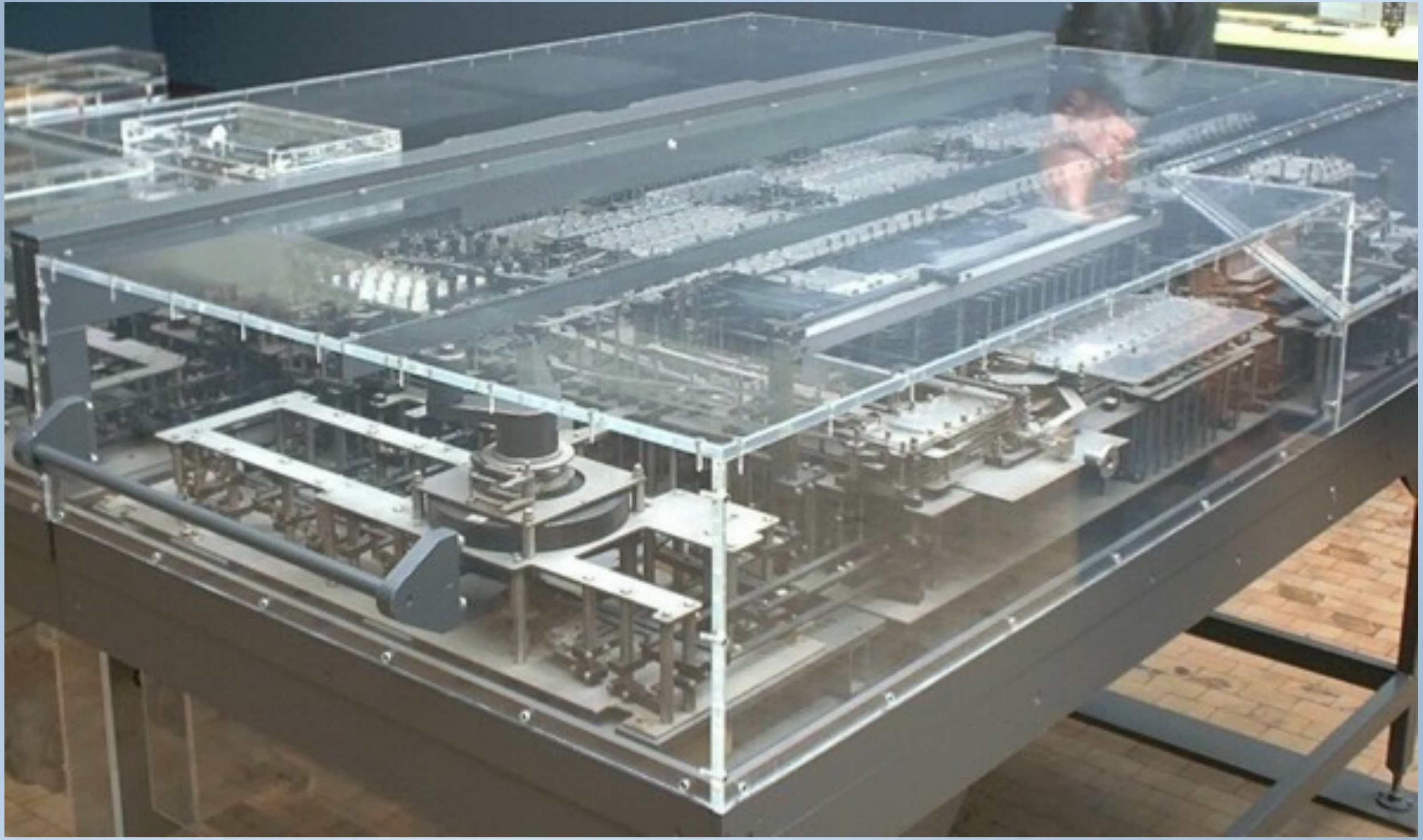

Image credits: https://en.wikipedia.org/wiki/Z1_(computer)#/media/File:Zuse_Z1.jpg

In some ways, yes, in others, no!

"Z1 was a machine of about 1000 kg weight, which consisted from some 20000 parts. It was a programmable computer, based on binary floating point numbers and a binary switching system. It consisted completely of thin metal sheets, which Kuno and his friends produced using a jigsaw."[6] "The [data] input device was a keyboard...The Z1's programs (Zuse called them Rechenplans) were stored on punch tapes by means of a 8-bit code."

– "**Konrad Zuse—the first relay computer**"
http://history-computer.com/ModernComputer/Relays/Zuse.html

Staying Up-To-Date

“Mr. Morgan, what will the stock market do next year?”

“It will fluctuate, my boy, it will fluctuate.”
-Attributed to J.P. Morgan in “The Intelligent Investor” by Benjamin Graham

Staying Up-To-Date

- Similarly, the only thing I can guarantee you in our field is that it will change
- Will functional programming take over the world? Will property-based testing become the new standard? Will Elixir be the web development language of choice?
- These are my opinions, but who knows… I have been wrong before and will be wrong again
- But I do know that those who don’t change will find themselves left behind.

What Can I Do to Stay Up To Date?

- Read books
- Read Hacker News
- Learn new languages, framework, and libraries that sound interesting! Especially ones that are different from what you normally use…
  - Haskell, Lisp, Prolog, J, Ruby, Julia, Elixir, Erlang, Rust, Go, Dart, Racket, etc.

What Can I Do to Stay Up To Date?

- Meet with other programmers! Especially ones not at your company or on your team. Build a network. If you are in Pittsburgh, check out Code & Supply - https://codeandsupply.co/, Slack: https://codeandsupply.slack.com
- Give talks/presentations (they are hard at first but get easier)
- Go to conferences
- Blog / write about your findings (writing a book is hard but very worthwhile, and there is nothing stopping you)
- Contribute to open source projects

As A Software Engineer, You Are A Professional

- Act as a professional
  - This doesn't mean a suit and tie, it means respect people and your work
- Professionals care about what they do more than whom they are working for
- Keep ethics in mind
  - Volkswagen Emission Scandal
  - THERAC-25
  - StuxNet