Université de Paris

Génie Logiciel
Towards Software Engineering

Sylvain Lobry

09/09/2022

A bit of history

Back to wooclap

https://www.wooclap.com/L3GL122

A bit of history

What is a computer?

- Can be argued that the first computer was built by Blaise Pascal in 1642 (he was 19): Addition, substraction, multiplication and division
- Go see it live (4 at CNAM), check how to operate

A pascaline

A bit of history

What is a computer (back in the 17th century)?

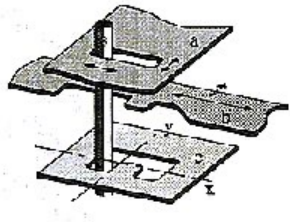
- One computer = one task
- What we call a program today is embeded in the computer's architecture
- New task? Need for a new computer

- Became desirable to have a separation between hardware and software:
  - Hardware: general, tangible
  - Software: specific, abstract

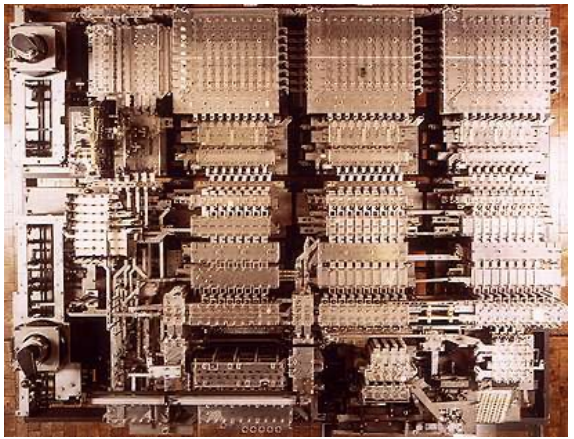
A bit of history

Zuse S1: the first freely programmable binary computer

- Built by Konrad Zuse from 1936 to 1938
- Mechanical (no electricity except for the clock)
- Clock speed: 1Hz (!!)
- Programmable through punch cards
- Destroyed in 1943
- Read more about it



1 bit of memory



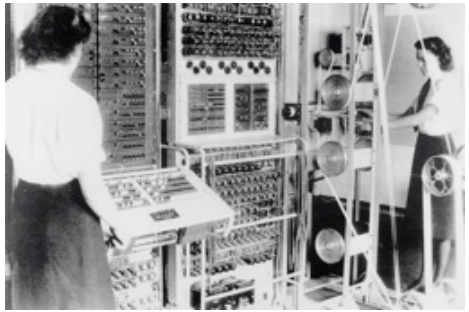
Bird-eye view of the Zuse S1
(Source: Deutsches Technik Museum Berlin)

A bit of history

Electronical computers

- Atanasoff-Berry Computer (1937-1942, US): not programmable
- Colossus (1943, UK): programmable (through cables): used to decrypt German's communications
- ENIAC (1945, US): turing complete, programmable, fast (100KHz!)

- By the way: at that time, "computer" is a job title given to women who operate calculcator



Colossus being operated

A bit of history

The software

- Computer can quickly run operations -> software becomes possible

- Ada Lovelace sees potential in calculator.
- 1842: writes an algorithm intended to be run by a machine to compute Bernouilli numbers (theoretical software!)

- 1937: Turing publish a paper establishing software theoretically

A bit of history

The software

- 21 June 1948: first software by Tom Kilburn run on the Manchester Baby -> look for the highest factor of 262,144

```
import time

number = 2**18
factor = number - 1
start = time.time()

while number % factor != 0:
    factor -= 1

end = time.time()
print(f"It took {(end - start) * 10**3}ms to find the answer {factor}")
```

It took 14.51ms to find the answer 131072

A bit of history

The software

- 21 June 1948: first software by Tom Kilburn run on the Manchester Baby -> look for the highest factor of 262,144
- 1951-1952: A-0 System by Grace Hopper, first “compiler”
- 1959: COBOL (COmmon Business Oriented Language) is created
- 1966: Apollo Guidance Computer (4th astronaut)

Margaret Hamilton next to the software listing produced for the Apollo mission

In 18 years, increasingly complex software is used for sensitive applications

A bit of history

The software

- 21 June 1948: first software by Tom Kilburn run on the Manchester Baby -> look for the highest factor of 262,144
- 1951-1952: A-0 System by Grace Hopper, first “compiler”
- 1959: COBOL (COmmon Business Oriented Language) is created
- 1966: Apollo Guidance Computer (4th astronaut)

Margaret Hamilton next to the software listing produced for the Apollo mission

In 18 years, increasingly complex software is used for sensitive applications

What could go wrong?

The need for software engineering

- Increasingly complex software for sensitive applications
- Problems of budget, deadlines, de-bugging, maintenance, …
- There is a need to address these issues:
  - 1968: first NATO (OTAN in French) conference on software engineering
  - Name coined by Margaret Hamilton

What could go wrong?

Increasingly large software

Software	Number of lines of code
First software (largest factor of 2**18)	17
Unix v1.0	10K
LibreOffice	9M
Android	~15M
Facebook	62M
Google	2B

Number of lines of codes in softwares
(Around 2015, various sources)

Infographic

What could go wrong?

Large softwares are more likely to be given up

Project's size (Lines of code)	Risk of giving up
100K	25%
500K	50%
1M	65%

Source: Casper Jones

What could go wrong?

Back to wooclap

https://www.wooclap.com/L3GL122

What could go wrong?

Different factors

	Defect Origins	Find Hours	Repair Hours	Total Hours
1	Security defects	11.00	24.00	35.00
2	Errors of omission	8.00	24.00	32.00
3	Hardware errors	3.50	28.00	31.50
4	Abeyant defects	5.00	23.00	28.00
5	Data errors	1.00	26.00	27.00
6	Architecture defects	6.00	18.00	24.00
7	Toxic requirements	2.00	20.00	22.00
8	Requirements defects	5.00	16.50	21.50
9	Supply chain defects	6.00	11.00	17.00
10	Design defects	4.50	12.00	16.50
11	Structural defects	2.00	13.00	15.00
12	Performance defects	3.50	10.00	13.50
13	Bad test cases	5.00	7.50	12.50
14	Bad fix defects	3.00	9.00	12.00
15	Poor test coverage	4.50	2.00	6.50
16	Invalid defects	3.00	3.00	6.00
17	Code defects	1.00	4.00	5.00
18	Document defects	1.00	3.00	4.00
19	User errors	0.40	2.00	2.40
20	Duplicate defects	0.25	1.00	1.25

Software Economics and Function Point Metrics: Thirty years of IFPUG Progress
Casper Jones

- Many sources of defects
- Programming errors are only one of them
- number of Lines Of Code (LOC) / time: bad productivity measure

- Strong need to study and improve the other parts of a software project

What could go wrong?

Back to wooclap

https://www.wooclap.com/L3GL122