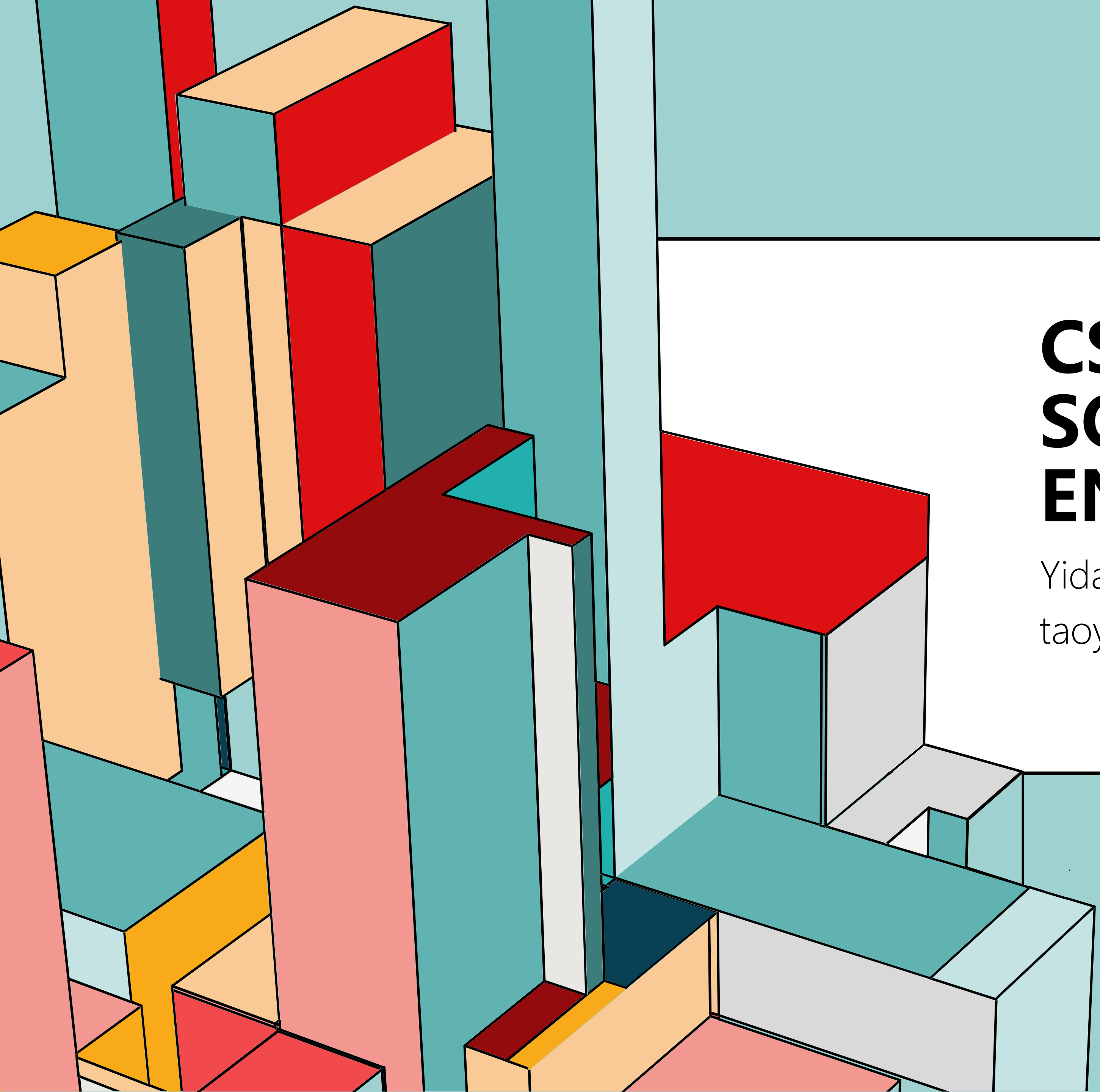


CS304 SOFTWARE ENGINEERING

Yida Tao

taoyd@sustech.edu.cn

Q1. WHAT IS SOFTWARE?

Is programming assignment a software?

WHAT'S THE EXPECTED LIFE SPAN OF YOUR CODE?

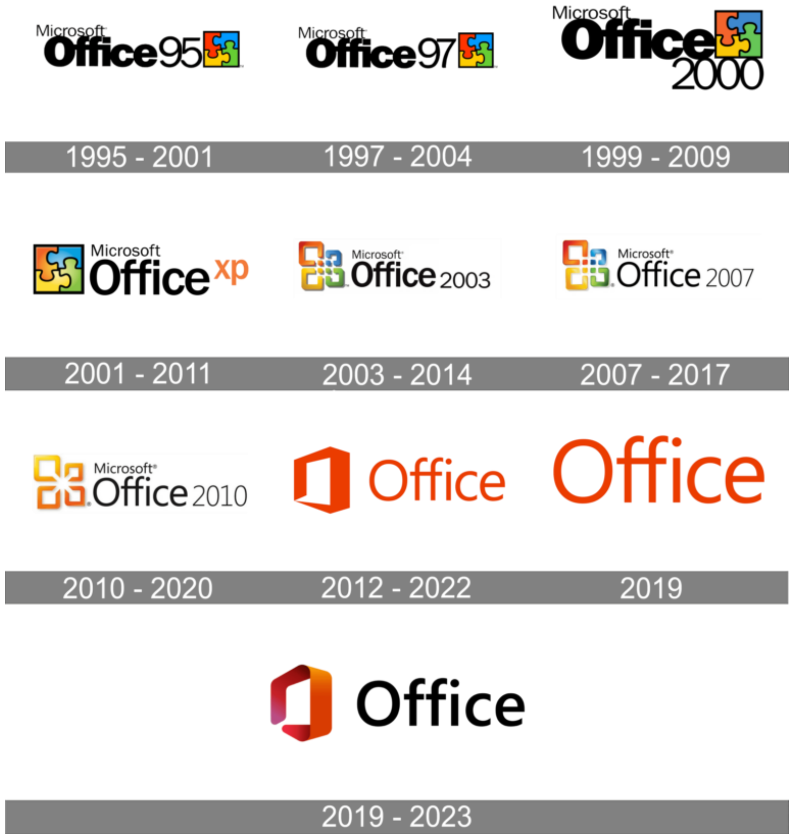
Programming Assignment

- **Short-term**: your programming code is likely to last for only hours, days, or weeks, not any longer (i.e., decades)
- **No-change**: You probably won't upgrade and maintain your programming code after the assignment deadline ☺

Software

- **Long-term**: large software (e.g., Microsoft Office, Google Search) tend to live for decades
- **Adapt-to-change**: to allow for longer life spans, software needs to adapt to new versions of underlying dependencies, OS, hardware, programming language versions, etc.

WHAT'S THE EXPECTED LIFE SPAN OF YOUR CODE?



Software

- **Long-term**: large software (e.g., Microsoft Office, Google Search) tend to live for decades
- **Adapt-to-change**: to allow for longer life spans, software needs to adapt to new versions of underlying dependencies, OS, hardware, programming language versions, etc.

WHAT'S THE EXPECTED LIFE SPAN OF YOUR CODE?

Elon Musk
@elonmusk

Just leaving Twitter HQ code review

4:28 AM · Nov 19, 2022

36.9K Retweets **16.1K** Quote Tweets **464K** Likes

Software

- **Long-term**: large software (e.g., Microsoft Office, Google Search) tend to live for decades
- **Adapt-to-change**: to allow for longer life spans, software needs to adapt to new versions of underlying dependencies, OS, hardware, programming language versions, etc.

HOW MANY RESOURCES ARE INVOLVED?

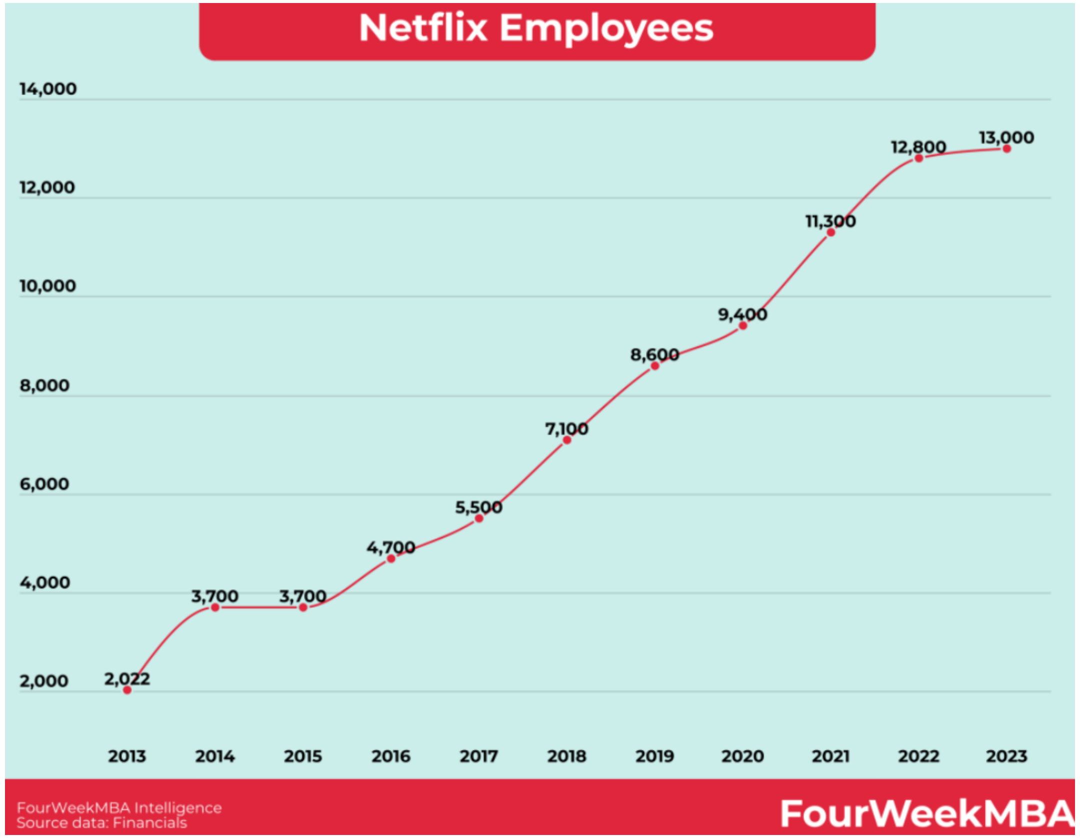
Programming Assignment

- **Human resources**: programming assignments are done by individuals or 2-3 sized small groups
- **Computing resources**: a single laptop is generally sufficient for programming assignments

Software

- **Human resources**: large software is developed and maintained by (large) teams
- **Computing resources**: as organization and users grow, large software needs to **scale** well with compute, memory, storage, bandwidth resources

HOW MANY RESOURCES ARE INVOLVED?



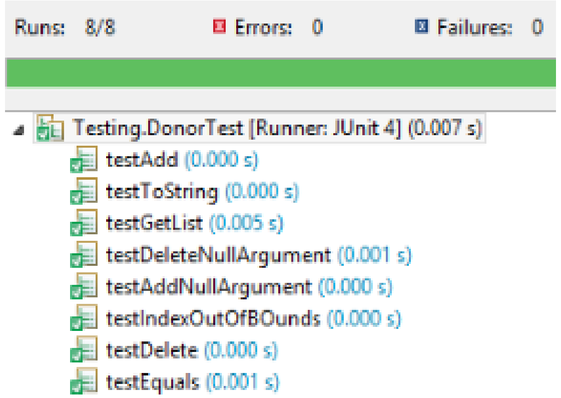
Software

- **Human resources**: large software is developed and maintained by (large) teams
- **Computing resources**: as organization and users grow, large software needs to **scale** well with compute, memory, storage, bandwidth resources

COMPLEXITY OF DECISIONS

Programming Assignment

- Correctness
- Time (e.g., deadline)



```
Runs: 8/8        Errors: 0        Failures: 0

Testing.DonorTest [Runner: JUnit 4] (0.007 s)
    testAdd (0.000 s)
    testToString (0.000 s)
    testGetList (0.005 s)
    testDeleteNullArgument (0.001 s)
    testAddNullArgument (0.000 s)
    testIndexOutOfBOunds (0.000 s)
    testDelete (0.000 s)
    testEquals (0.001 s)
```

Software

- Software quality
- Engineering efforts
- Financial costs
- Resource costs (e.g., CPU time)
- Social impact
- …

ARTIFACTS

Programming Assignment

- Code

Software

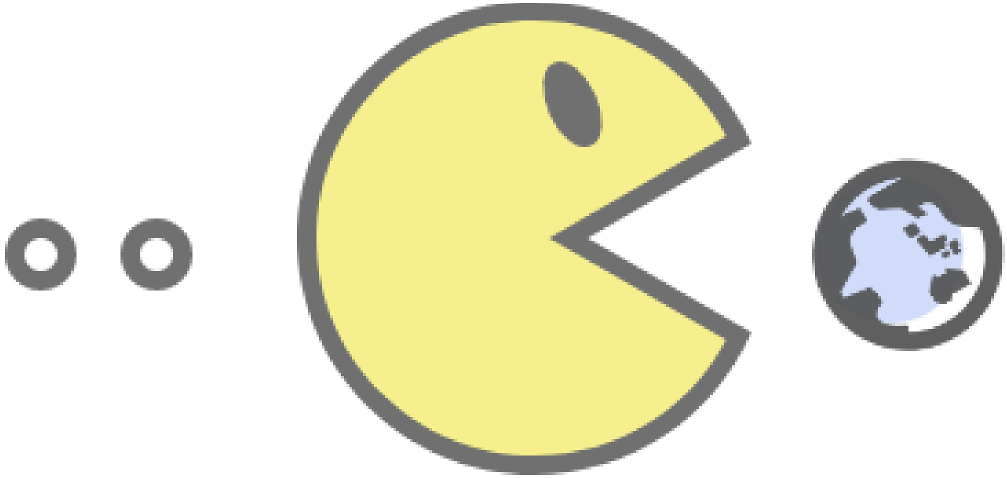
- Code
- Tests
- Documentation
- Build scripts
- Configurations
- Logs
- Executables (.exe etc.)
- ……

SOFTWARE IS EVERYWHERE

SOFTWARE IS EATING THE WORLD

In the future, every company will become a software company.

Today, every company is software intensive. That means, if they do not have a software strategy, then they will not be able to compete in the world and eventually kind of fade out.



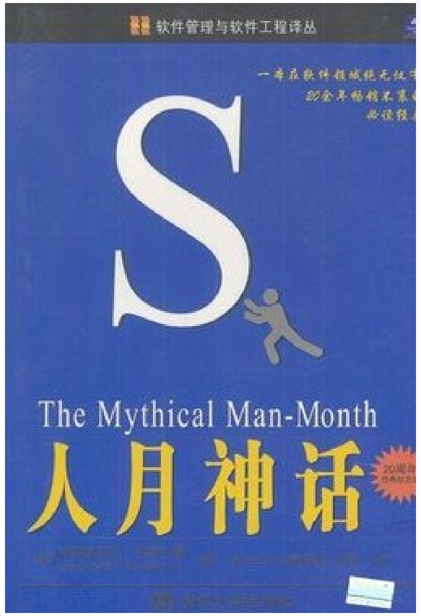
Marc Andreessen. Co-author of Mosaic, the first widely used web browser; co-founder of Netscape.
Why Software Is Eating the World. Wall Street Journal, 2011.

Q2. HOW TO BUILD A SOFTWARE?

Building a software == coding?

THE BUILDING OF IBM OS/360

- Time: 1963-1966
- Human involved: 5000 man-month (one person’s working time for a month)
- Codebase: 1M lines of code
- Cost: hundreds of millions $

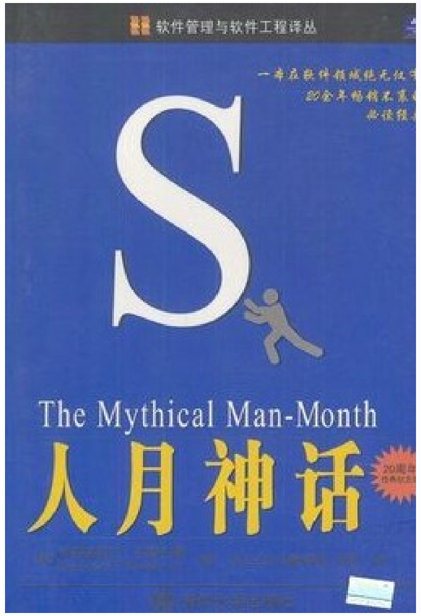


图灵奖得主、IBM 360系统之父
Frederick Brooks

THE BUILDING OF IBM OS/360

- Deferred releases
- Underestimated cost & memory resources
- Low-quality in first public release
- Thousands of bug fixes even after several releases

Adding manpower to a software project that is behind schedule delays it even longer

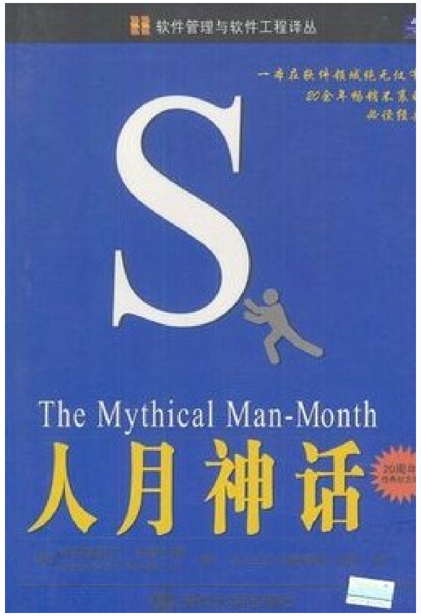


图灵奖得主、IBM 360系统之父
Frederick Brooks

THE BUILDING OF IBM OS/360

Software like a tar pit (焦油坑): The more you fight it, the deeper you sink!

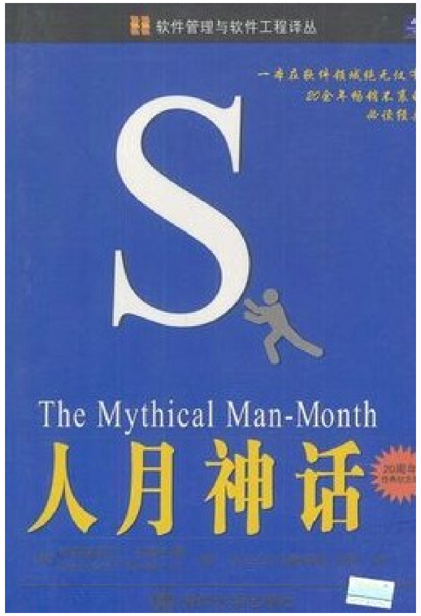
……正像一只逃亡的野兽落到泥潭中做垂死的挣扎，越是挣扎，陷得越深， 最后无法逃脱灭顶的灾难。……程序设计工作正像这样一个泥潭，……一批批程序员被迫在泥潭中拼命挣扎，……谁也没有料到问题竟会陷入这样的困境……



图灵奖得主、IBM 360系统之父
Frederick Brooks

All programmers say "Oh, I can easily beat the xx lines / day cited by industrial programmers."

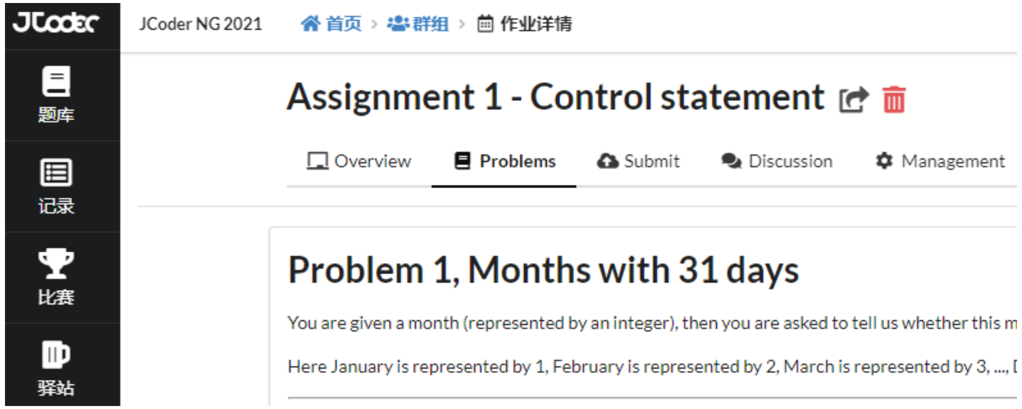
They are talking about just **coding something, not building a product**.



图灵奖得主、IBM 360系统之父
Frederick Brooks

BUILDING A SOFTWARE VS. PROGRAMMING

- **Programming** is a significant part of software engineering
  - Programming is how you generate a new software in the first place
- **Building a software** is programming integrated over **time, scale, and trade-offs**



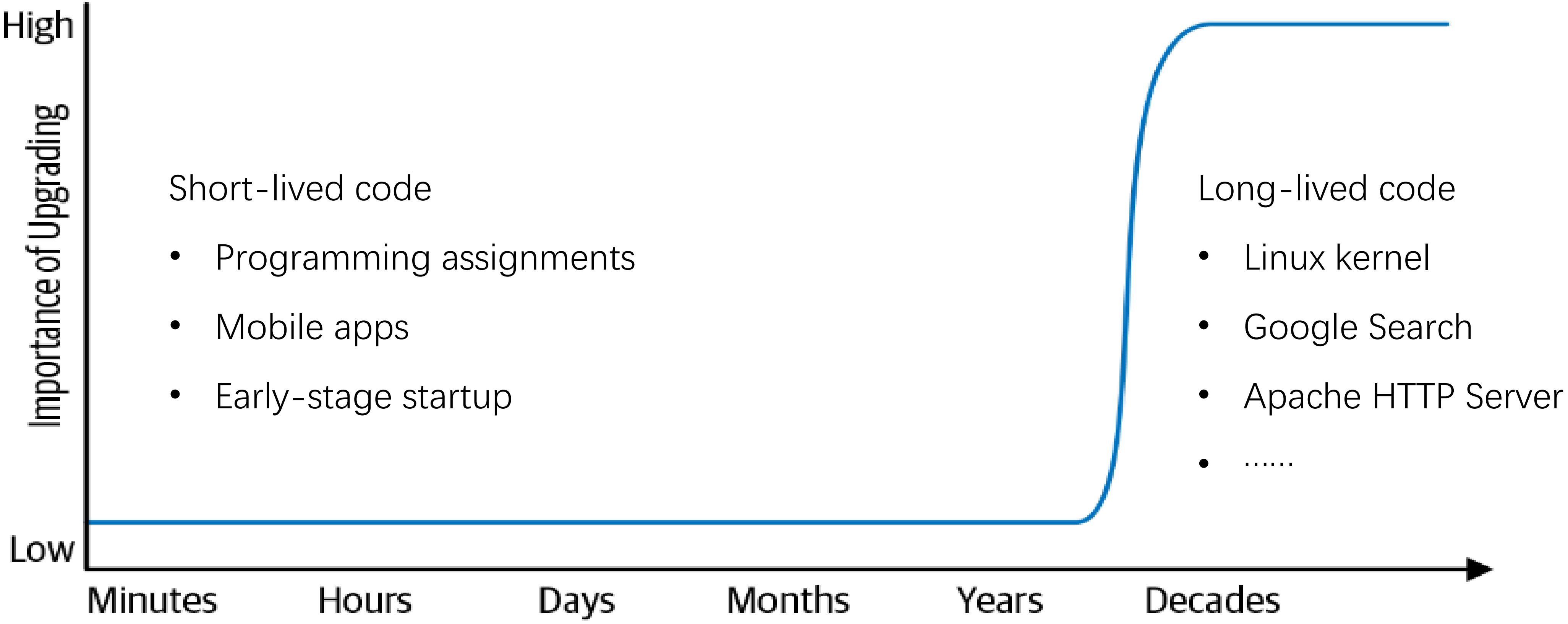
Assignment 1 - Control statement

Problem 1, Months with 31 days

You are given a month (represented by an integer), then you are asked to tell us whether this m

Here January is represented by 1, February is represented by 2, March is represented by 3, ..., I

LIFE SPAN OF SOFTWARE



Short-lived code

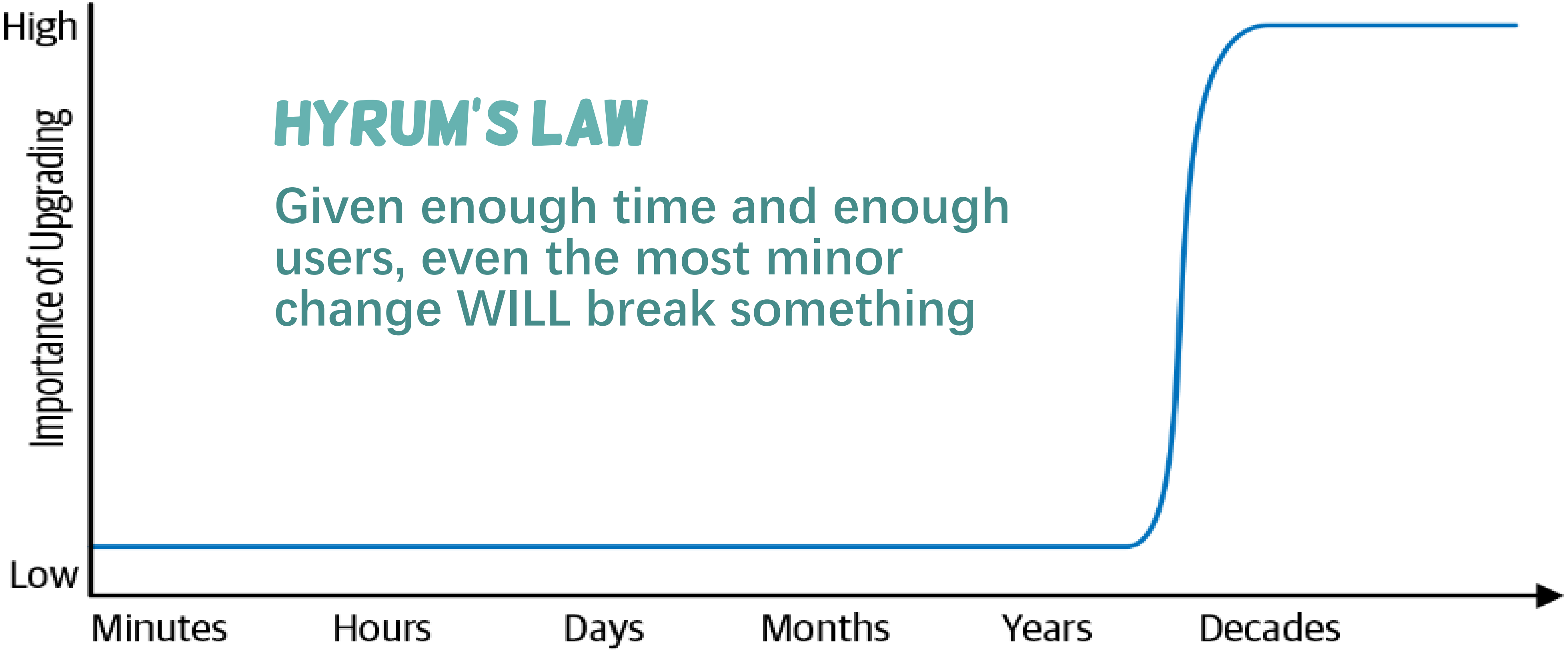
- Programming assignments
- Mobile apps
- Early-stage startup

Long-lived code

- Linux kernel
- Google Search
- Apache HTTP Server
- ……

Figure 1-1. Life span and the importance of upgrades

LIFE SPAN OF SOFTWARE

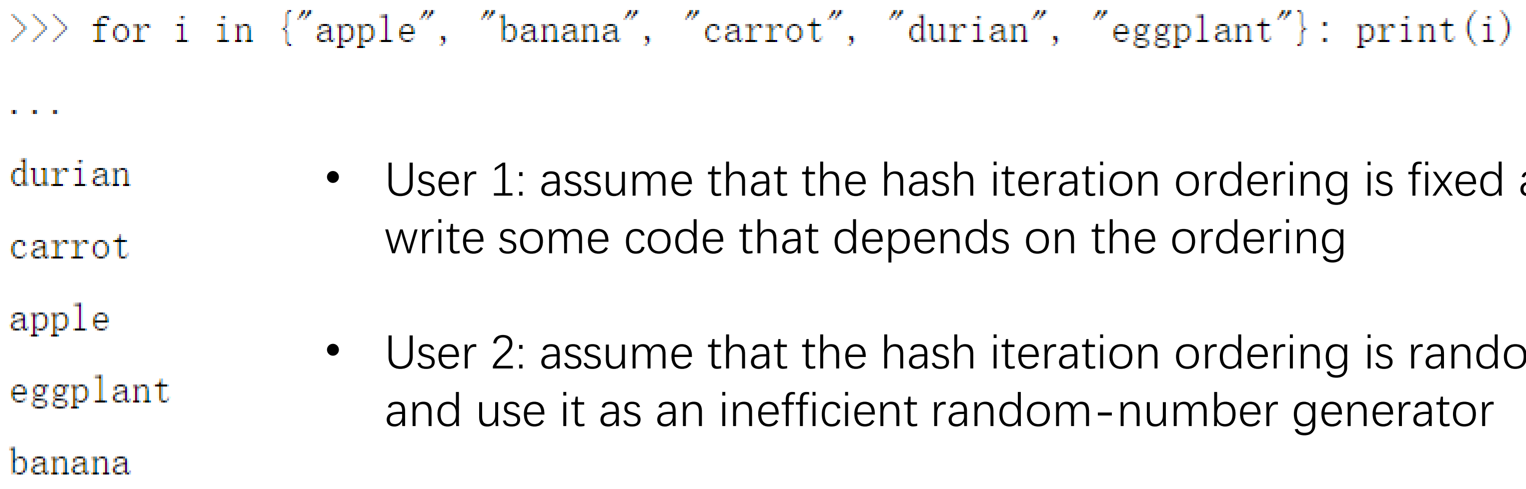


HYRUM'S LAW

Given enough time and enough users, even the most minor change WILL break something

Figure 1-1. Life span and the importance of upgrades

HYRUM'S LAW IN PRACTICE



```
>>> for i in {"apple", "banana", "carrot", "durian", "eggplant"}: print(i)
...
durian
carrot
apple
eggplant
banana
```

- User 1: assume that the hash iteration ordering is fixed and write some code that depends on the ordering
- User 2: assume that the hash iteration ordering is random, and use it as an inefficient random-number generator
- User 3: assume that the hash iteration ordering is random and never write any code that depend on the ordering.

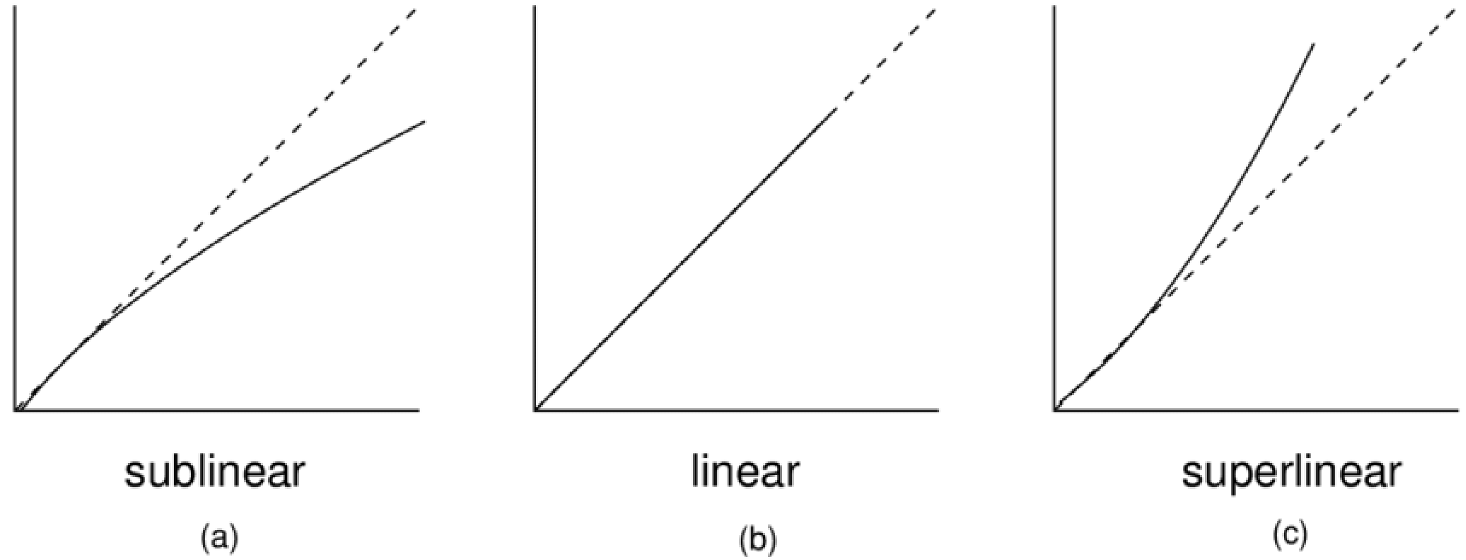
User 1 breaks if the hash ordering becomes randomized

User 2 breaks if the hash ordering becomes fixed

User 4 breaks if the results produced by user 3 are consumed by user 4, whose code depends on the order of the results

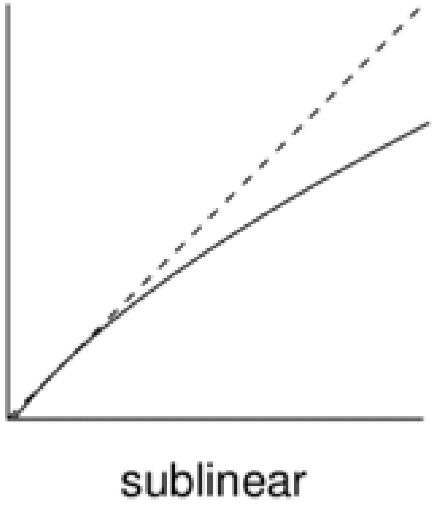
SCALE

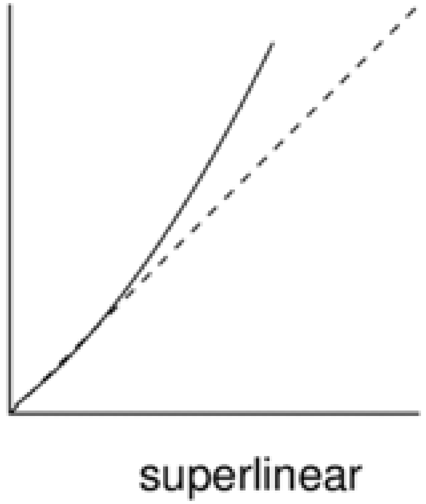
- Your organization's codebase is sustainable when you are able to change all of the things that you ought to change, safely, and can do so for the life of your codebase.
- If costs grow **superlinearly** over time, the operation clearly is **not scalable**.



SCALE

- X axis: the demand (e.g., codebase)
- Y axis: resources costs
    - How many additional computing resources (e.g., memory, storage) are needed?
    - How many additional human involvement is needed?
    - How long does it take to do a full build?
    - How long does it take to pull a fresh copy of the repository?
    - How much will it cost to upgrade to a new language version?
    - How long does it take to respond to users?
    - ….





Q3. WHAT IS SOFTWARE ENGINEERING?

WHAT IS ENGINEERING?

Engineering is the application of an empirical, scientific approach to finding efficient solutions to practical problems.

THE ORIGIN OF SOFTWARE ENGINEERING

Software Engineering was first formally used by Professor Friedrich L. Bauer, at NATO conference, the first conference on software engineering, in 1968:

“The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines.”

Software engineering is now considered one of major computing disciplines.

NATO conference, 1968

WHAT IS SOFTWARE ENGINEERING?

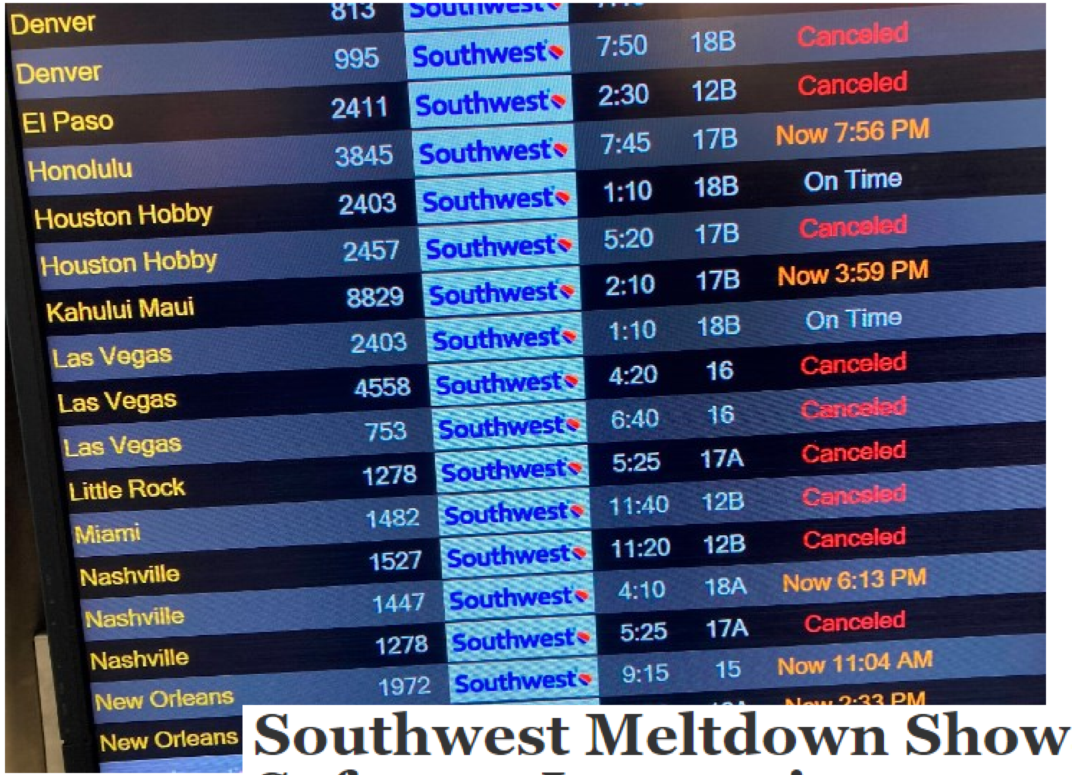
Software engineering: the branch of engineering that deals with the design, development, testing, and maintenance of software applications, while ensuring that the software to be built is:

- Correct
- Consistent
- On budget
- On time
- Within the required requirements.

Q4. OBJECTIVE OF SOFTWARE ENGINEERING?

1. **Quality**: improve the quality of software
2. **Efficiency**: improve the efficiency of building software

SOFTWARE IS EATING THE WORLD



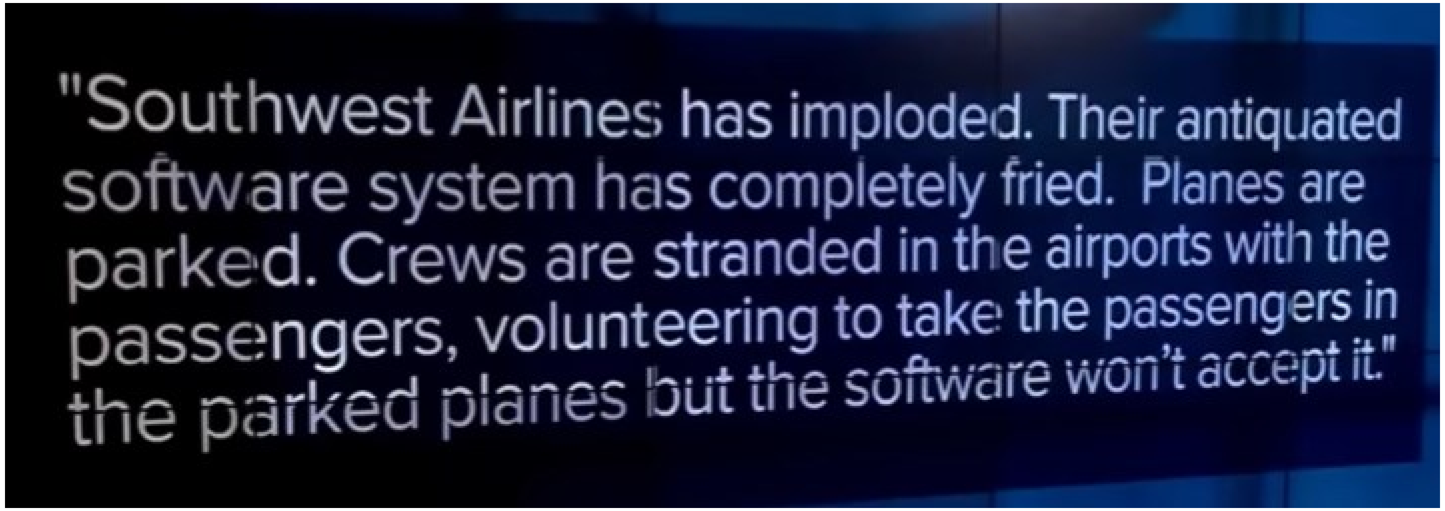
Denver	813	Southwest			
Denver	995	Southwest	7:50	18B	Canceled
El Paso	2411	Southwest	2:30	12B	Canceled
Honolulu	3845	Southwest	7:45	17B	Now 7:56 PM
Houston Hobby	2403	Southwest	1:10	18B	On Time
Houston Hobby	2457	Southwest	5:20	17B	Canceled
Kahului Maui	8829	Southwest	2:10	17B	Now 3:59 PM
Las Vegas	2403	Southwest	1:10	18B	On Time
Las Vegas	4558	Southwest	4:20	16	Canceled
Las Vegas	753	Southwest	6:40	16	Canceled
Little Rock	1278	Southwest	5:25	17A	Canceled
Miami	1482	Southwest	11:40	12B	Canceled
Nashville	1527	Southwest	11:20	12B	Canceled
Nashville	1447	Southwest	4:10	18A	Now 6:13 PM
Nashville	1278	Southwest	5:25	17A	Canceled
New Orleans	1972	Southwest	9:15	15	Now 11:04 AM
New Orleans					Now 2:33 PM

Southwest Meltdown Shows Airlines Need Tighter Software Integration

The airline industry is long overdue for a tech overhaul that takes full advantage of the cloud and data integration, analysts say

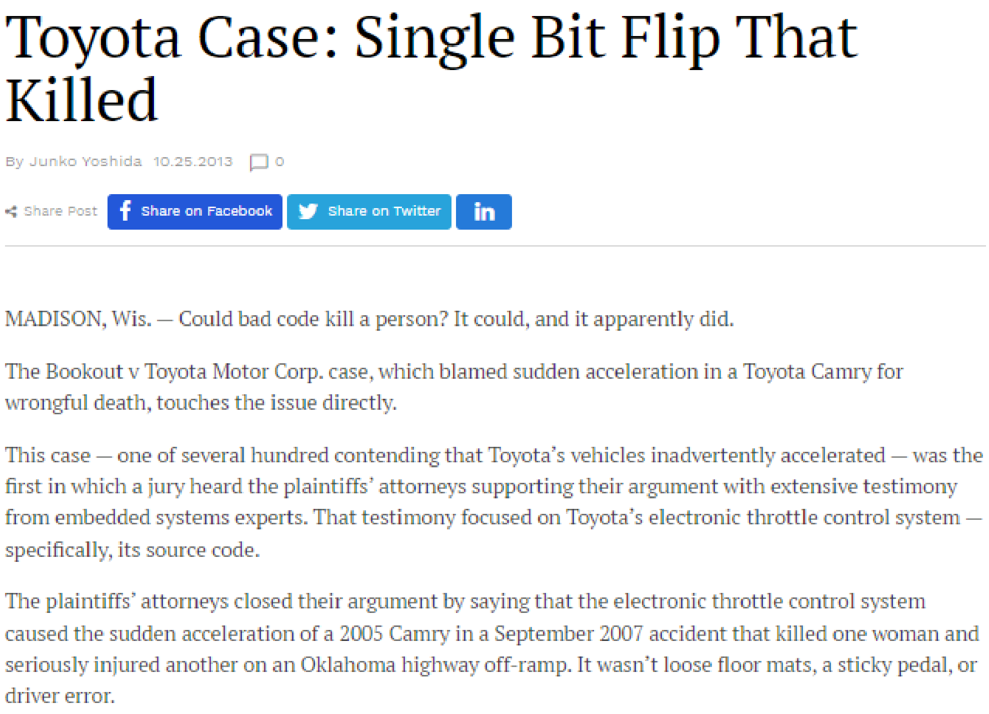
SOFTWARE IS EATING THE WORLD

Southwest relies on crew-assignment software called SkySolver, an old application developed decades ago



"Southwest Airlines has imploded. Their antiquated software system has completely fried. Planes are parked. Crews are stranded in the airports with the passengers, volunteering to take the passengers in the parked planes but the software won't accept it."

SOFTWARE IS EATING THE WORLD



Toyota Case: Single Bit Flip That Killed

By Junko Yoshida 10.25.2013 0

Share Post Share on Facebook Share on Twitter in

MADISON, Wis. — Could bad code kill a person? It could, and it apparently did.

The Bookout v Toyota Motor Corp. case, which blamed sudden acceleration in a Toyota Camry for wrongful death, touches the issue directly.

This case — one of several hundred contending that Toyota's vehicles inadvertently accelerated — was the first in which a jury heard the plaintiffs' attorneys supporting their argument with extensive testimony from embedded systems experts. That testimony focused on Toyota's electronic throttle control system — specifically, its source code.

The plaintiffs' attorneys closed their argument by saying that the electronic throttle control system caused the sudden acceleration of a 2005 Camry in a September 2007 accident that killed one woman and seriously injured another on an Oklahoma highway off-ramp. It wasn't loose floor mats, a sticky pedal, or driver error.

Hard questions raised when a software 'glitch' takes down an airliner

Posted by Taylor Armerding on Thursday, November 29, 2018

The parts and systems on an airplane don't have to fail in a big way to have big consequences. A flaw in airline software could be a matter of life or death.

The original version of this post was published in Forbes.

It doesn't take a failure of anything big to cause big trouble—big as in massive, catastrophic, and lethal damage to a sophisticated transportation system.

THE IMPACT OF ENGINEERING IN SOFTWARE

DevOps, a software engineering practice, **improves efficiency dramatically**:

- 8000x faster deployment time
- 21% less time spent on unplanned work and rework
- 44% more time on new work
- 50% lower change-failure rates
- 50% less time spent fixing security issues
- 50% higher market cap growth over 3 years

https://services.google.com/fh/files/misc/state-of-devops-2014.pdf

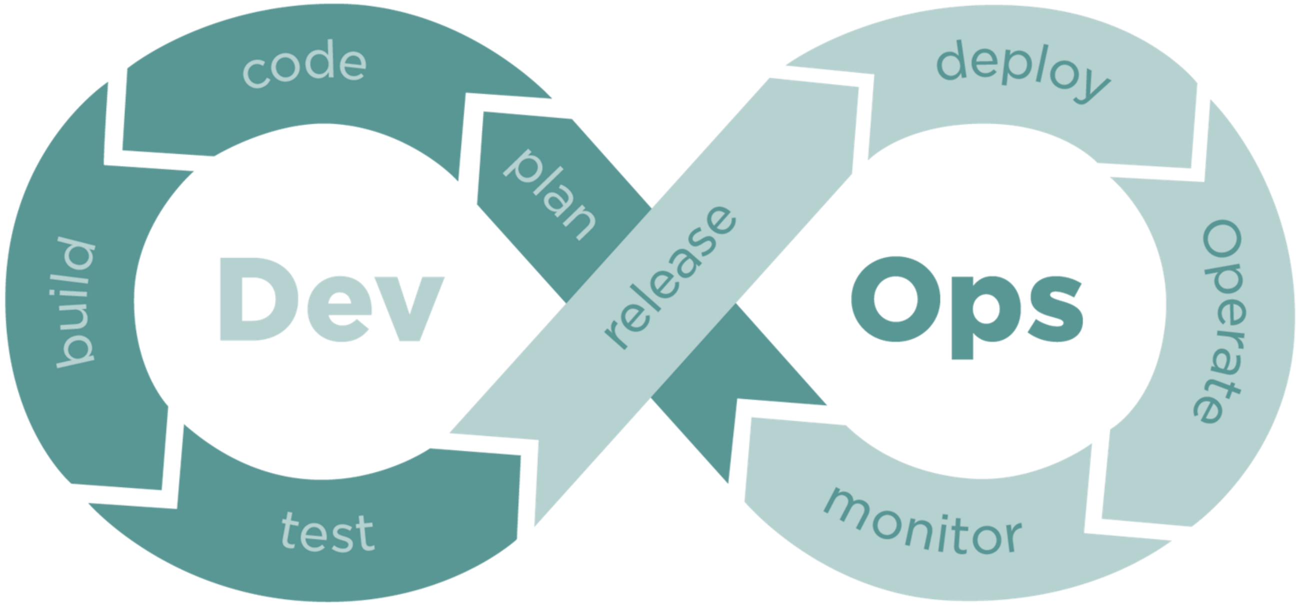
COURSE THEMES

The themes may change as the course progresses.

1. Intro to Software Engineering
2. Software Process & DevOps
3. Version Control Systems
4. Software Requirements
5. Software Design
6. Software Architecture
7. Build & Dependency Management
8. Software Documentation
9. Software Testing
10. Continuous Integration
11. Continuous Deployment
12. Software Quality & Metrics
13. Software Evolution & Maintenance
14. AI in Software Engineering

GETTING LOST IN THE COURSE?

1. Intro to Software Engineering
2. Software Process & DevOps
3. Version Control Systems
4. Software Requirements
5. Software Design
6. Software Architecture
7. Build & Dependency Management
8. Software Documentation
9. Software Testing
10. Continuous Integration
11. Continuous Deployment
12. Software Quality & Metrics
13. Software Evolution & Maintenance
14. AI in Software Engineering



GETTING LOST IN THE COURSE?

1. Intro to Software Engineering
2. Software Process & DevOps
3. Version Control Systems
4. Software Requirements
5. Software Design
6. Software Architecture
7. Build & Dependency Management
8. Software Documentation
9. Software Testing
10. Continuous Integration
11. Continuous Deployment
12. Software Quality & Metrics
13. Software Evolution & Maintenance
14. AI in Software Engineering

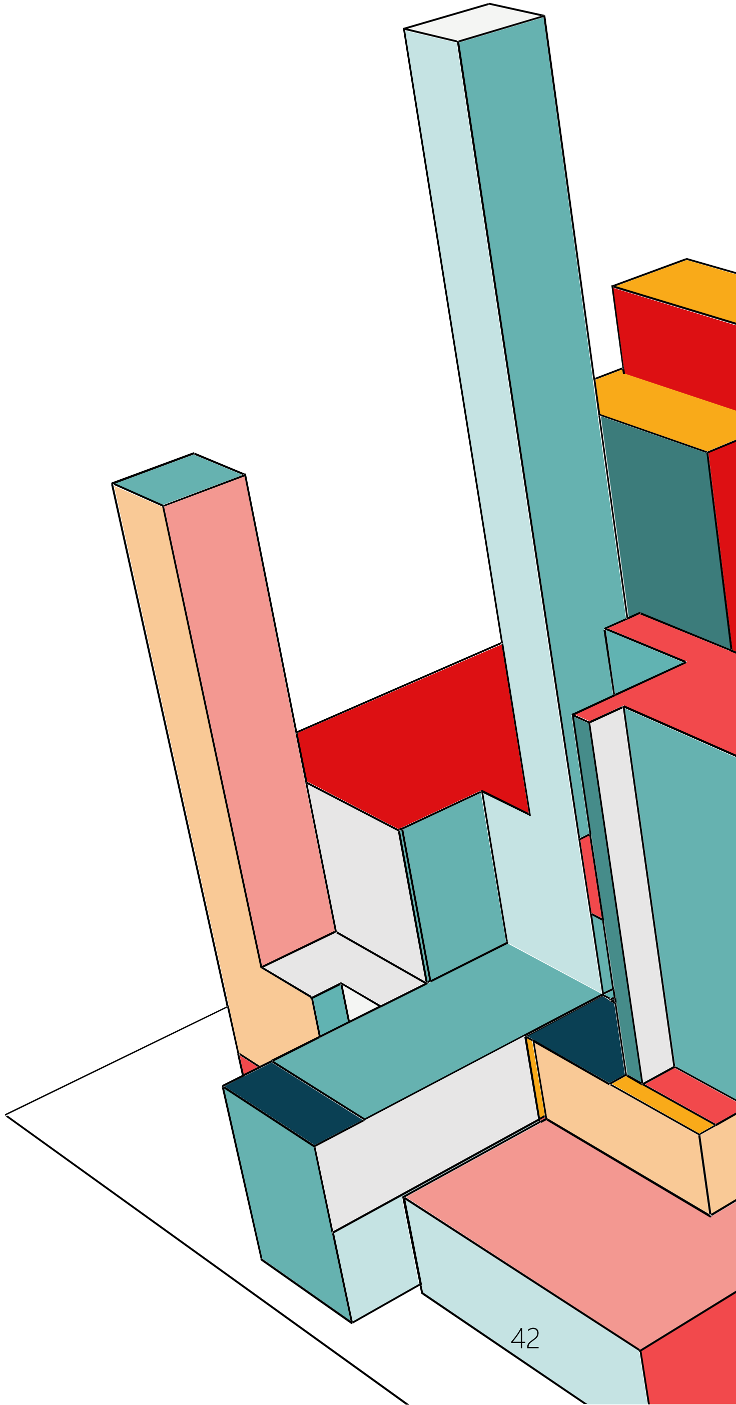
GETTING LOST IN THE COURSE?

- The principles, concepts, practices and methodology introduced in this course all emerge from growing experiences of building real software

- To acquire a better understanding of the lectures, take labs and the team project seriously.

READINGS

- Chapter 4. What Is Software Engineering? Software Engineering at Google by Titus Winters, et al.



NEXT

- Software process
- DevOps