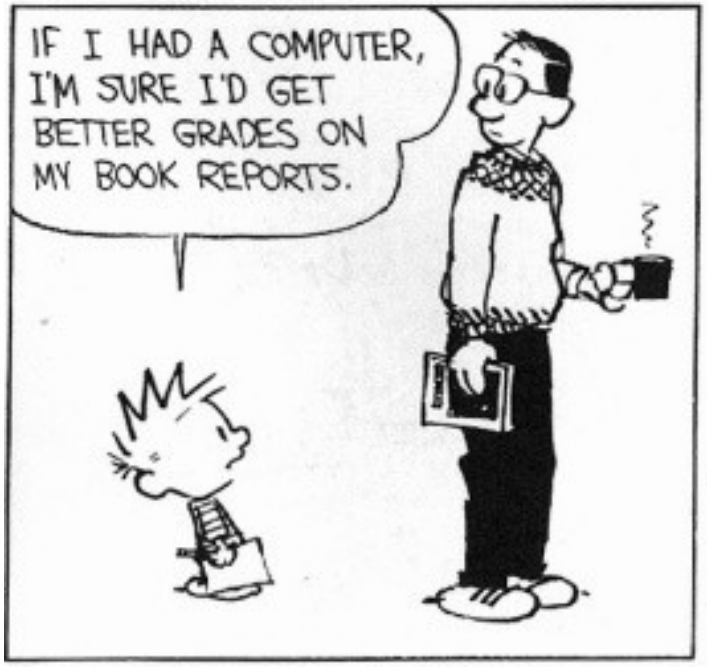
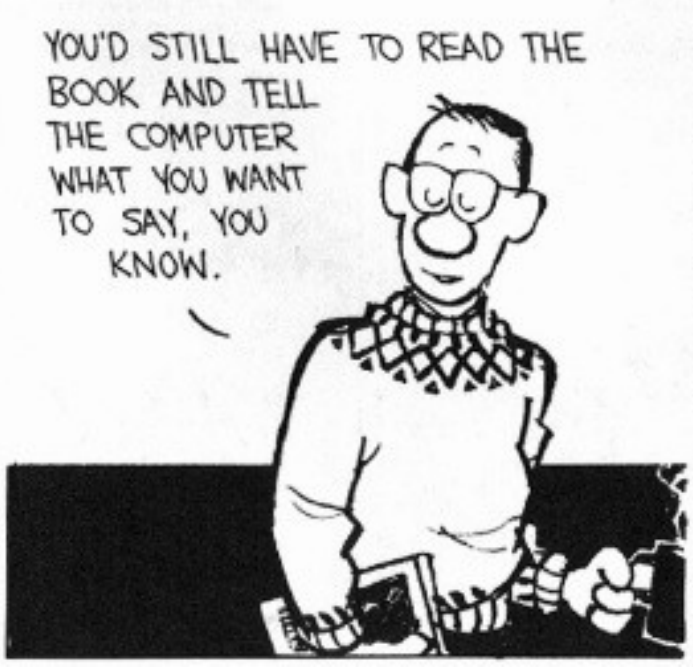
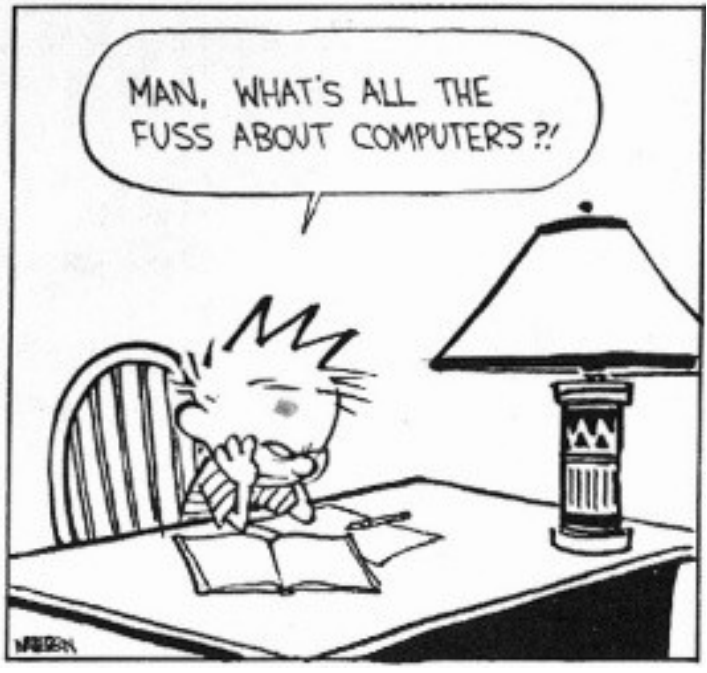
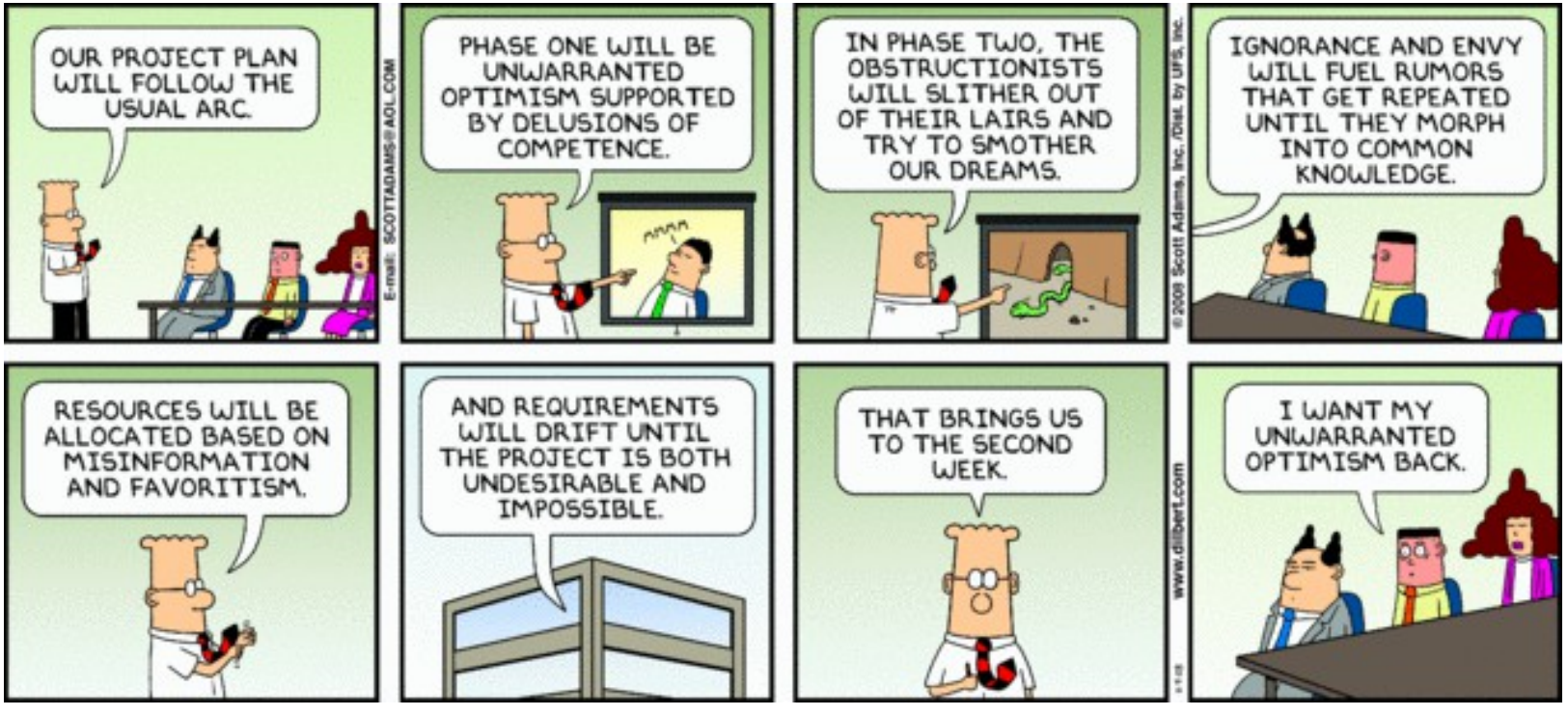






Software Engineering

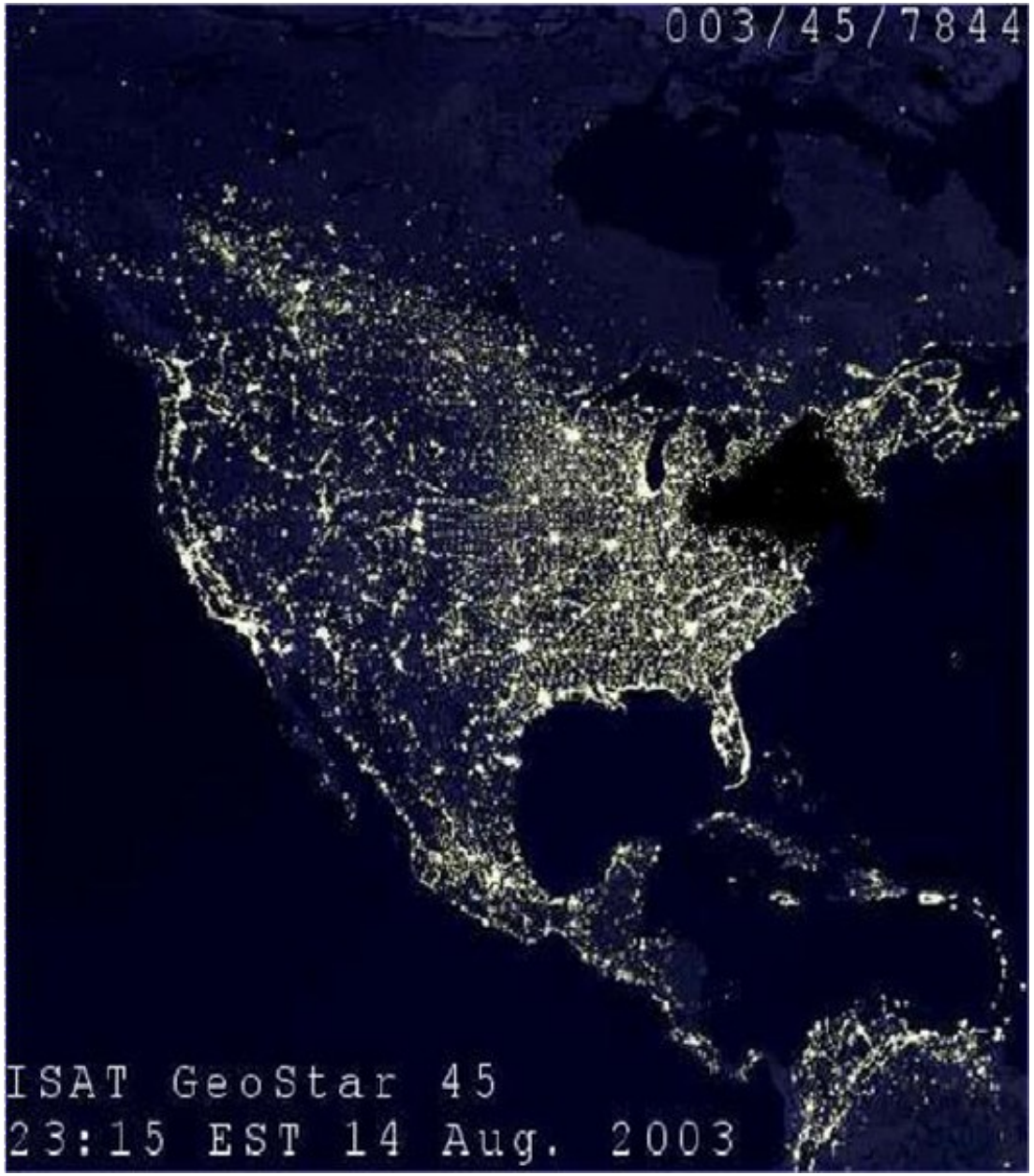


List of public corporations by market capitalization

This list is up to date as of December 31, 2018. Indicated changes in market value are relative to the previous quarter.

Rank	First Quarter	Second Quarter	Third Quarter	Fourth Quarter
1	Apple Inc. ▼851,317	Apple Inc. ▲909,840[10]	Apple Inc. ▲1,091,000[10]	Microsoft ▼780,520[11]
2	Alphabet Inc. ▼715,404	Amazon.com ▲824,790[12]	Amazon.com ▲976,650[12]	Apple Inc. ▼748,680[10]
3	Microsoft ▲702,760[11]	Alphabet Inc. ▲774,840[13]	Microsoft ▲877,400[11]	Amazon.com ▼735,900[12]
4	Amazon.com ▲700,672[12]	Microsoft ▲757,640[11]	Alphabet Inc. ▲839,740[14]	Alphabet Inc. ▼728,360[14]
5	Tencent ▲507,990[15]	Facebook ▲562,480[16]	Berkshire Hathaway ▲523,520[17]	Berkshire Hathaway ▼499,590[17]
6	Berkshire Hathaway ▲492,019[17]	Tencent ▼478,580[15]	Facebook ▼473,850[16]	Facebook ▼375,890[16]
7	Alibaba Group ▲470,930[18]	Alibaba Group ▲476,040[18]	Alibaba Group ▼423,600[18]	Tencent ▼375,110[15]
8	Facebook ▼464,189[16]	Berkshire Hathaway ▼463,980[17]	Tencent ▼388,080[15]	Alibaba Group ▼355,130[18]
9	JPMorgan Chase ▲377,410[19]	JPMorgan Chase ▼354,780[19]	JPMorgan Chase ▲379,440[19]	Johnson & Johnson ▼346,110[20]
10	Johnson & Johnson ▼343,780[20]	ExxonMobil ▲350,270 [21]	Johnson & Johnson ▲370,650[20]	JPMorgan Chase ▼324,660[19]

Find The “Mitten” of Michigan



Software is Critical: Power

The **Northeast blackout of 2003** was a widespread power outage that occurred throughout parts of the Northeastern and Midwestern United States and the Canadian province of Ontario on Thursday, August 14, 2003, just after 4:10 p.m. EDT.[1]

Some power was restored by 11 p.m. Most did not get their power back until two days later. In other areas it took nearly a week or two for power to be restored.[2] At the time, it was the world's second most widespread blackout in history, after the 1999 Southern Brazil blackout.[3][4] The outage, which was much more widespread than the Northeast Blackout of 1965, affected an estimated 10 million people in Ontario and 45 million people in eight U.S. states.

The blackout's primary cause was a programming error or "bug" in the alarm system at the control room of FirstEnergy Corporation, an Akron, Ohio-based company. The lack of an alarm left operators unaware of the need to re-distribute power after overloaded transmission lines hit unpruned foliage, triggering a "race condition" in the energy management system software, a bug affecting the order of operations in the system. What would have been a manageable local blackout cascaded into massive widespread distress on the electric grid.

Software is Critical: Defense

- Quoting an Air Force lieutenant general, “The only thing you can do with an F- 22 that does not require software is take a picture of it.”

[Crouching Dragon, Hidden Software: Software in DOD Weapon Systems (Ferguson, IEEE Software, 2001)]

Software is Critical: Driving

Toyota Case: Single Bit Flip That Killed

Junko Yoshida

10/25/2013 03:35 PM EDT

During the trial, embedded systems experts who reviewed Toyota's electronic throttle source code testified that they found Toyota's source code defective, and that it contains bugs -- including bugs that can cause unintended acceleration.

"We did a few things that NASA apparently did not have time to do," Barr said. For one thing, by looking within the real-time operating system, the experts identified "unprotected critical variables." They obtained and reviewed the source code for the "sub-CPU," and they "uncovered gaps and defects in the throttle fail safes."

The experts demonstrated that "the defects we found were linked to unintended acceleration through vehicle testing," Barr said. "We also obtained and reviewed the source code for the black box and found that it can record false information about the driver's actions in the final seconds before a crash."

Stack overflow and software bugs led to memory corruption, he said. And it turns out that the crux of the issue was these memory corruptions, which acted "like ricocheting bullets."

Barr also said more than half the dozens of tasks' deaths studied by the experts in their experiments "were not detected by any fail safe."

Bookout Trial Reporting

http://www.eetimes.com/document.asp?doc_id=1319903&page_number=1 (excerpts)

"Task X death in combination with other task deaths"

Software is Critical: Privacy

- Equifax security breach impacts 145.5 million
  - Name, SSN, DOB, Address. Also DL# and CC#.
  - “I didn't have to do anything fancy,” the researcher told Motherboard, explaining that the site was vulnerable to a basic “forced browsing” bug. The researcher requested anonymity out of professional concerns. “**All you had to do was put in a search term** and get millions of results, just instantly—in cleartext, through a web app,” they said. In total, the researcher downloaded the data of hundreds of thousands of Americans in order to show Equifax the vulnerabilities within its systems. They said they could have downloaded the data of all of Equifax's customers in 10 minutes: “I've seen a lot of bad things, but not this bad.”

Software is Critical: Healthcare

Healthcare.gov: Government IT Project Failure at its Finest

Posted: 10/18/2013 6:33 pm

Read more > Project Management, Government, Healthcare, It Projects, Open Source, Business News

The *BusinessWeek* article on the Healthcare.gov failure is nothing if not instructive. From the piece:

> Healthcare.gov isn't just a website; it's more like a platform for building health-care marketplaces. Visiting the site is like visiting a restaurant. You sit in the dining room, read the menu, and tell the waiter what you want, and off he goes to the kitchen with your order. The dining room is the front end, with all the buttons to click and forms to fill out. The kitchen is the back end, with all the databases and services. The contractor most responsible for the back end is CGI Federal. Apparently it's this company's part of the system that's burning up under the load of thousands of simultaneous users.

The restaurant analogy is a good one. Projects with scopes like these fail for all sorts of reasons. *Why New Systems Fail* details a bunch of culprits, most of which are people-related.

As I read the article, a few other things jumped out at me, as they virtually guarantee failure:

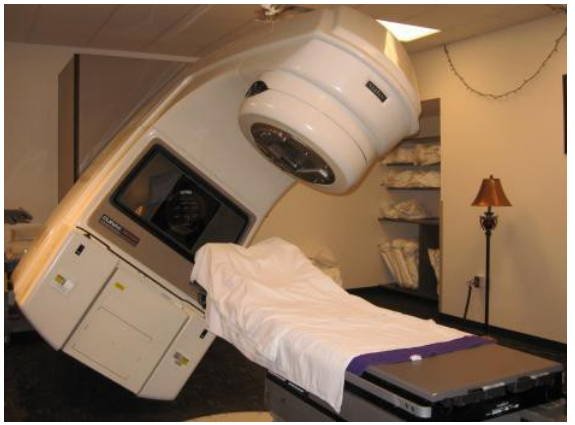
- The sheer number of vendors involved

Software is Critical: Space

- The European Space Agency's Ariane 5 Flight 501 was destroyed 40 seconds after takeoff (June 4, 1996). The US$1 billion prototype rocket self-destructed due to a bug in the on-board guidance software. (The bug? Bad conversion of `double` to `short`, leading to an overflow.)

Software is Critical: Healthcare (!)

- Therac-25 radiation therapy machine
- At least six accidents in which patients were given massive overdoses of radiation
- Because of concurrent programming errors, it sometimes gave its patients radiation doses that were hundreds of times greater than normal, resulting in death or serious injury

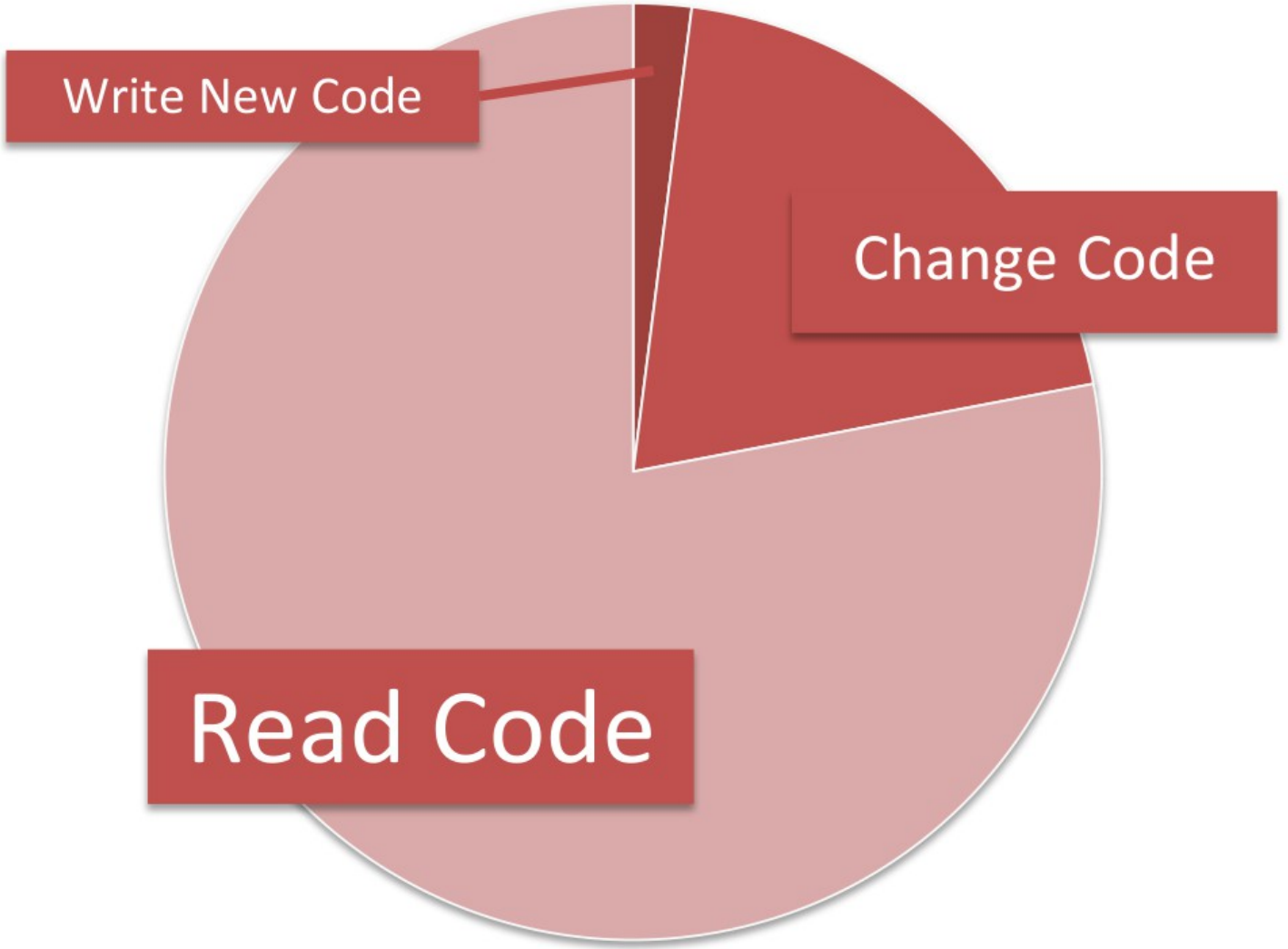


What Is Software Engineering?

What Is Software Engineering?

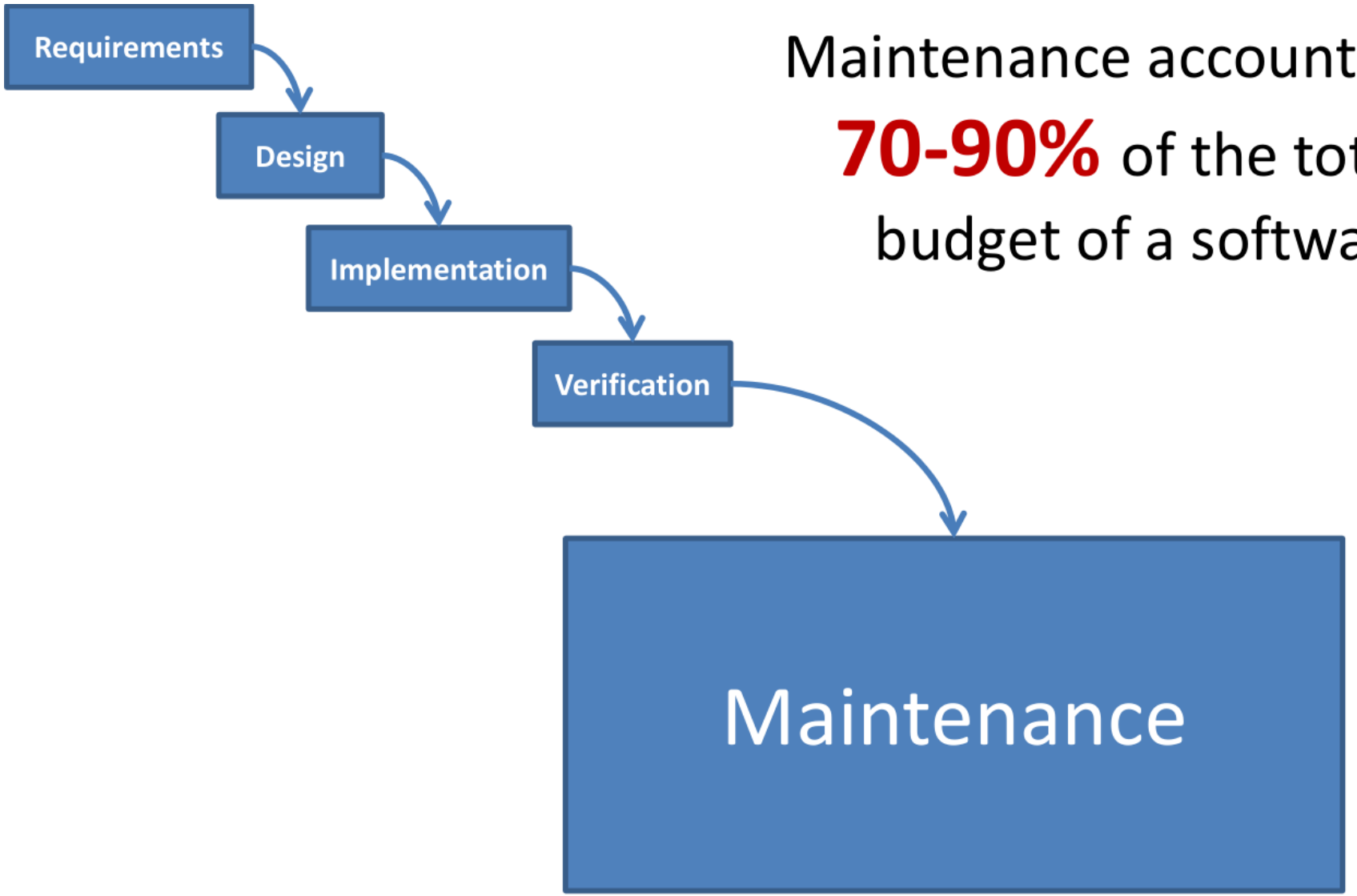
- The majority of industrial software engineering is *not* writing code.

- The dominant activities in software engineering are **comprehension** and **maintenance**.



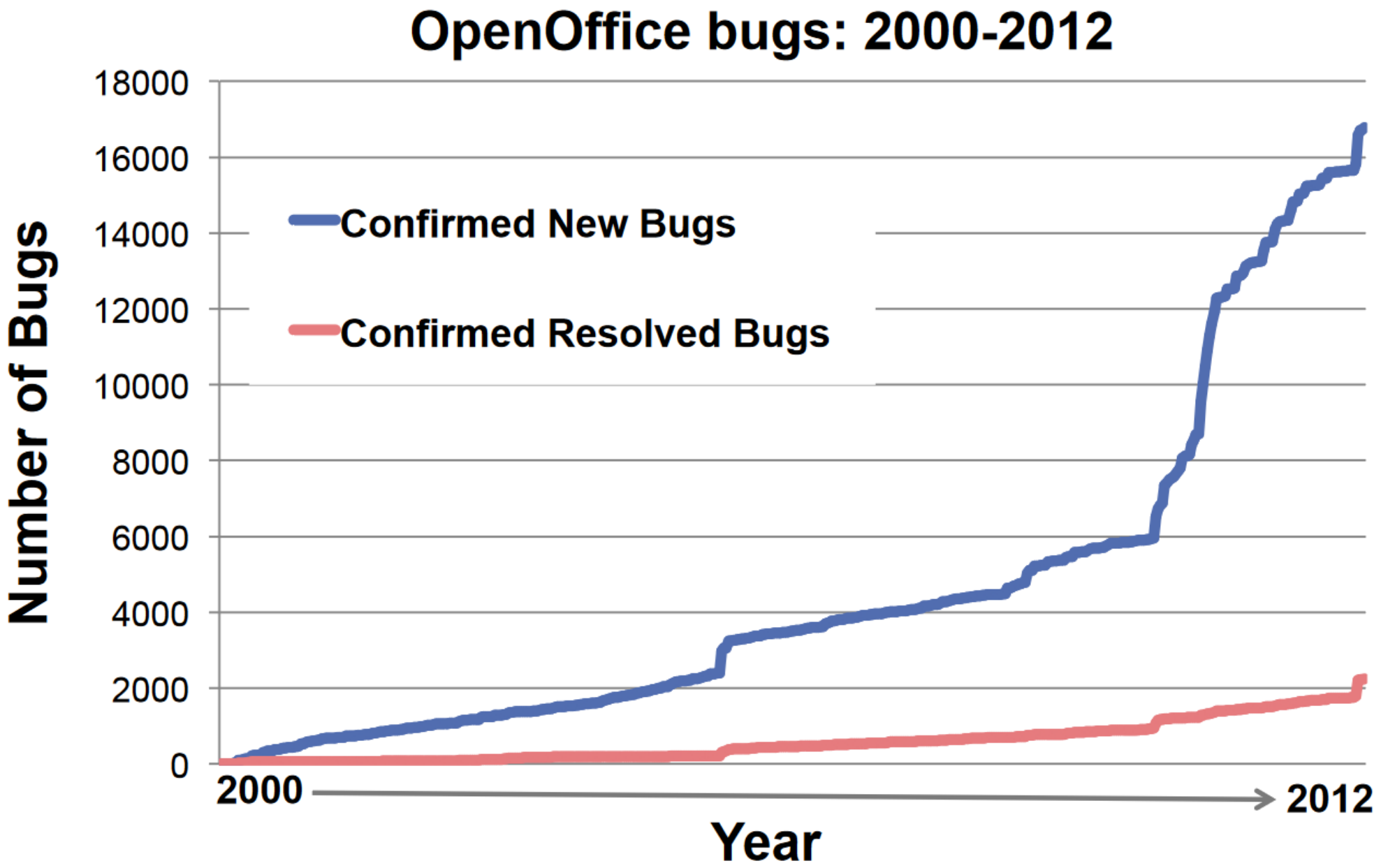
"**Understanding code** is **by far** the activity at which professional developers spend most of their time."

[Peter Hallam. *What Do Programmers Really Do Anyway?* Microsoft.]



Maintenance accounts for about **70-90%** of the total lifecycle budget of a software project.

[Pigoski. *Practical Software Maintenance: Best Practices for Managing Your Software Investment*. Seacord, Plakosh, and Lewis. *Modernizing Legacy Systems: Software Technologies*.]

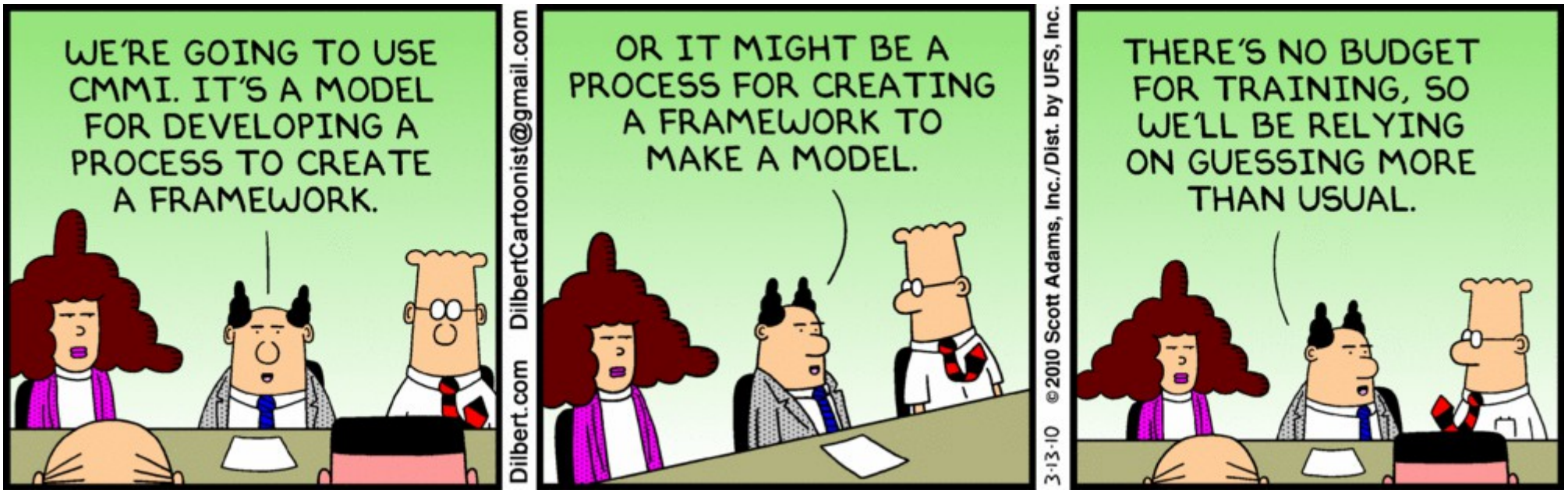


[Fry. *Leveraging Light-Weight Analyses to Aid Software Maintenance*. 2014]

A Key Issue

"Half of software engineering is crap."

– Your Instructor

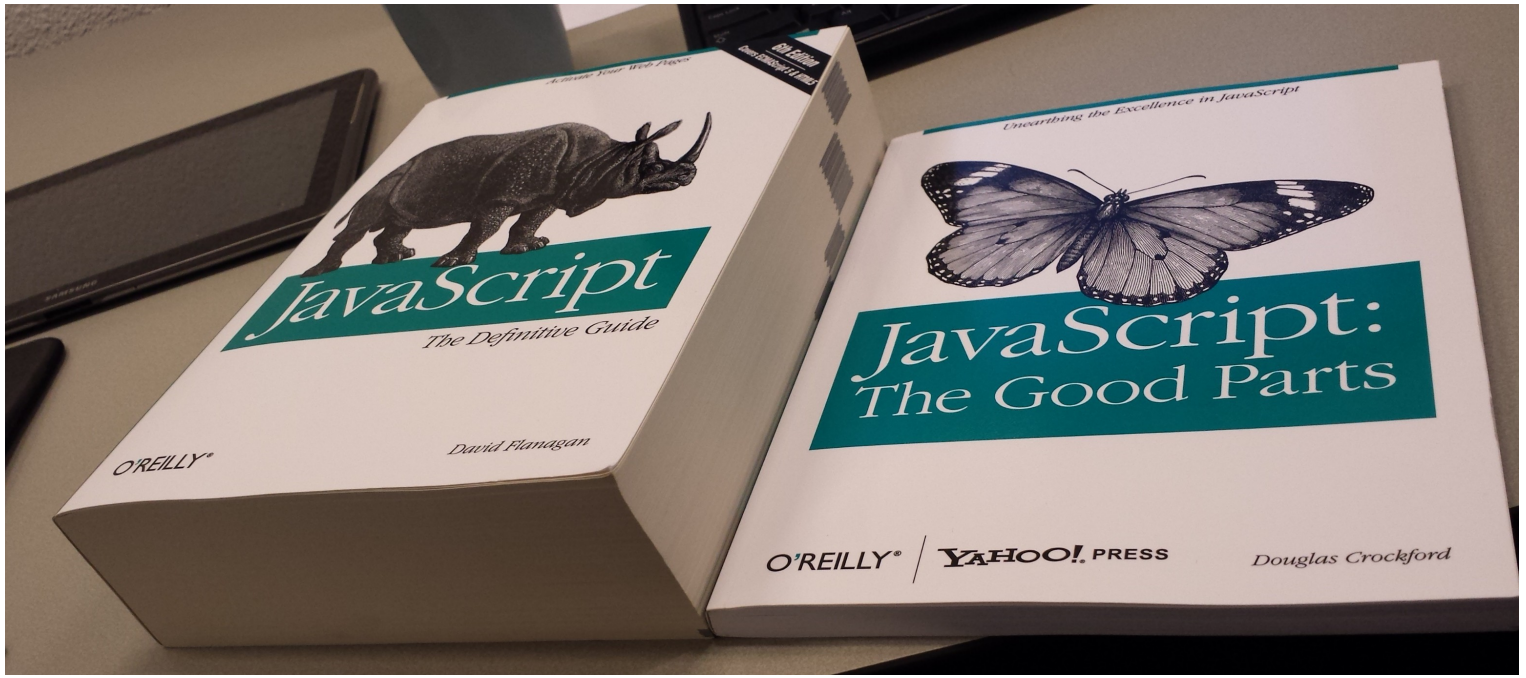


Revolutionary Solution

"So, by a vote of 8 to 2 we have decided to skip the Industrial Revolution completely, and go right into the Electronic Age."

Class Philosophy

"Anyway, here's the 'good parts' version. S. Morgenstern wrote it. And my father read it to me. And now I give it to you. What you do with it will be of more than passing interest to us all." – William Goldman, *The Princess Bride*



This Course

http://web.eecs.umich.edu/~weimerw/481/

- Administrivia
- Assignments and Grading
- Outline of Topics

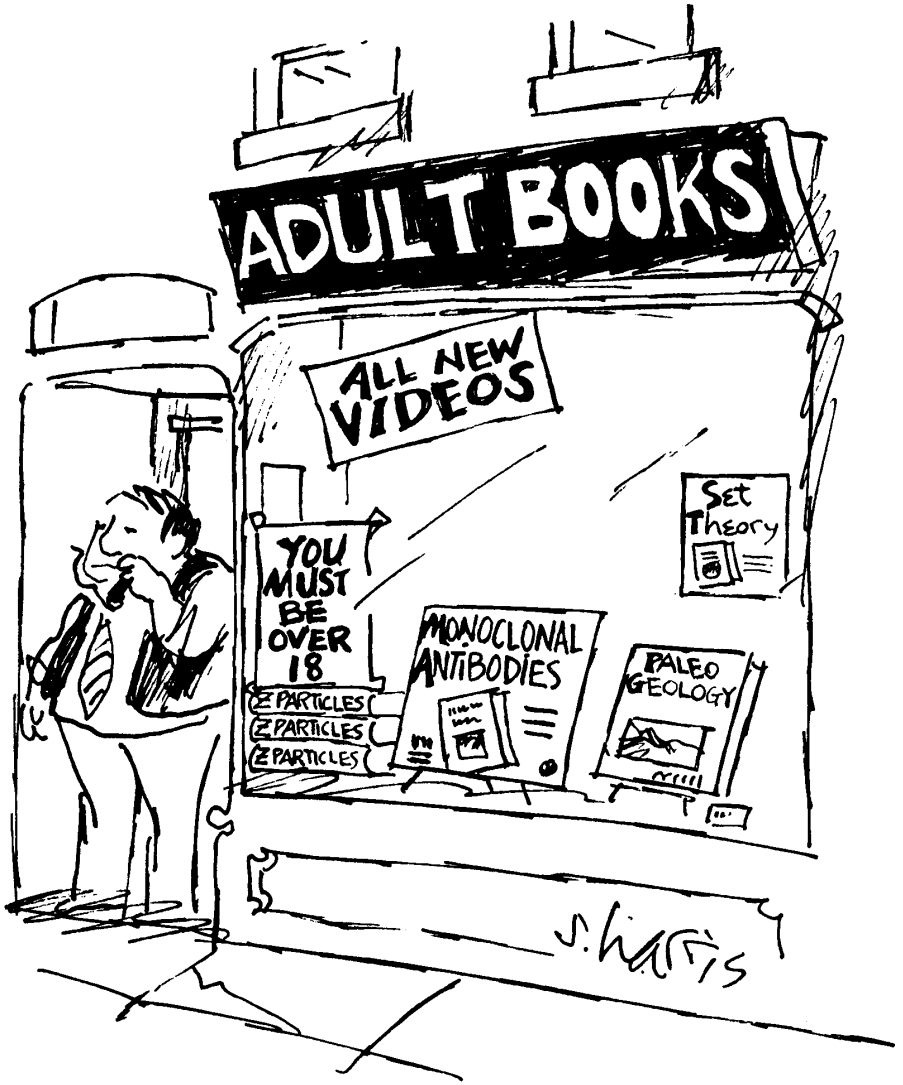
How will this help me graduate?

- Upper-Level CS/CE Technical Elective
- ENGR *and* LSA

- Major Design Experience
- Capstone

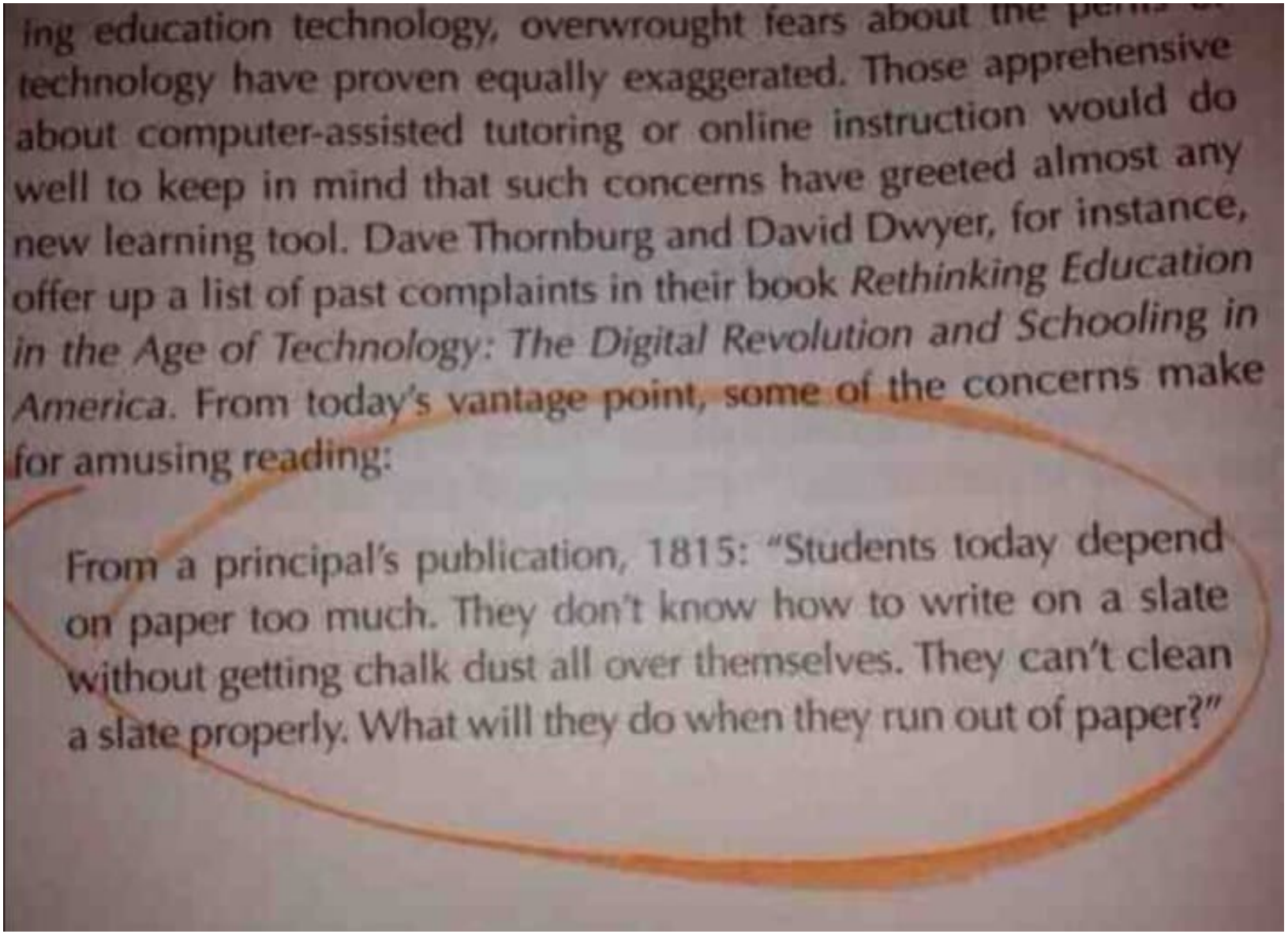
How Hard Is This Course?

- Workload Survey is misleading!
  - Easier than 281 (Data Structures) or 482 (OS)
  - Harder than 493 (UI)
- More "time consuming" than "difficult"
- See webpage quotes from former students



Blah blah laptops blah …

(up hill both ways ...)



ing education technology, overwrought fears about the pe
technology have proven equally exaggerated. Those apprehensive about computer-assisted tutoring or online instruction would do well to keep in mind that such concerns have greeted almost any new learning tool. Dave Thornburg and David Dwyer, for instance, offer up a list of past complaints in their book *Rethinking Education in the Age of Technology: The Digital Revolution and Schooling in America*. From today's vantage point, some of the concerns make for amusing reading:

From a principal's publication, 1815: "Students today depend on paper too much. They don't know how to write on a slate without getting chalk dust all over themselves. They can't clean a slate properly. What will they do when they run out of paper?"

Laptops and Cell Phones

"…participants who multitasked on a laptop during a lecture scored lower on a test compared to those who did not multitask, and participants who were in direct view of a multitasking peer scored lower on a test compared to those who were not. The results demonstrate that multitasking on a laptop poses a significant distraction to both users and **fellow students** and can be detrimental to comprehension of lecture content."

[Faria Sana, Tina Weston, and Nicholas J. Cepeda. 2013. *Laptop multitasking hinders classroom learning for both users and nearby peers.* Comput. Educ. 62 (March 2013), 24-31.]

Laptops and Cell Phones

"…students who took notes on laptops performed worse on conceptual questions than students who took notes longhand. We show that whereas taking more notes can be beneficial, laptop note takers' tendency to transcribe lectures verbatim rather than processing information and reframing it in their own words is detrimental to learning."

[Mueller PA1, Oppenheimer DM2. *The pen is mightier than the keyboard: advantages of longhand over laptop note taking.* Psychol Sci. 2014 Jun; 25(6):Epub 2014 Apr 23.]

Assignments and Grading

- Assigned reading due before each lecture
- Attend lecture, take notes, visit forum

- Six homework assignments (~60%)
- Reading quizzes (~ 5%)
- Two examinations (~35%)

- See webpage for regrade and makeup policy

Readings

- No expensive, outdated textbook
- Assigned reading to be done before lectures
  - High-level summaries (e.g., Wikipedia)
  - Industrial tech reports and academic research
  - Homework assignment instructions
  - Optional readings for further exploration
- Higher standard than the EECS usual

Mon Feb 05	Static Analysis Tools [qa]	• Wikipedia's *Static Program Analysis* • Ayewah et al.'s *Experiences Using Static Analysis to Find Bugs* [Google] • Optional: "How does Microsoft find bugs in critical device drivers?" Find out in: Ball et al.'s *A Decade of Software Model Checking with SLAM* [Microsoft] • Optional: "What are deployment challenges for static analysis?" Find out in: Bessey et al.'s *A Few Billion Lines of Code Later: Using Static Analysis to Find Bugs in the Real World* [Coverity]

Assignments

- Six Assignments
  - Test Coverage, Test Automation, Mutation Testing, Defect Detection, Debugging Automation, Open Source GitHub Contribution
- Coding: **`autograder.io`** (as in 280 and 490)
  - Multiple object languages (C, Java, Python, etc.)
- Writing: **`canvas`** or **`gradescope`**
- Due dates posted in advance (now!)
- Materials available in advance (now!)

Optional Teams

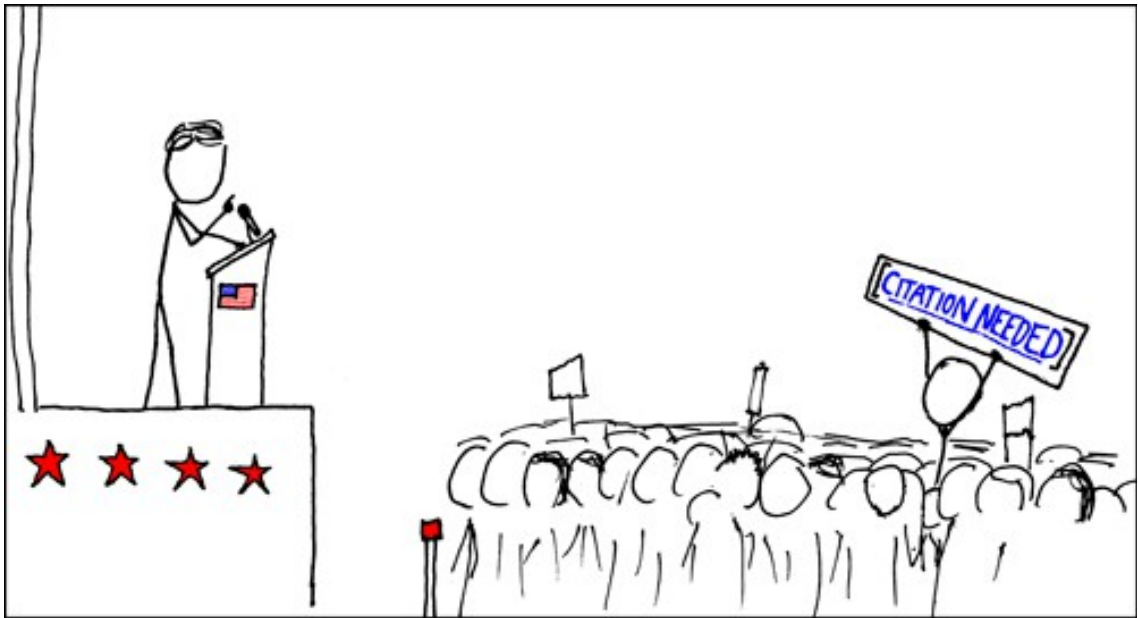
- Modern industrial software engineering is almost exclusively team-based
- But this is an ULCS, not a Capstone/MDE
  - You will be exposed to building a large project in a team elsewhere in the curriculum
- For most of the assignments, you may work **alone or in pairs** of your own choosing
  - We are not responsible if your partner disappears
  - Use the forum to find partners, etc.

Discussion Sections

- Homework help (!), exam preparation, explain difficult material, answer questions
  - Discussion 1 — F 9:30-10:30am 1005 DOW, starting January 18th (led by Sabrina Benge)
  - Discussion 2 — F 10:30-11:30am 1018 DOW, starting January 18th (led by Sabrina Benge)
  - Discussion 3 — F 1:30-2:30pm 1014 DOW, starting January 18th (led by James Perretta)
- I don't care which one you attend
  - … assuming there is space in the room, etc.

Software Engineering You Can Believe In

- Citations for strong claims (or ask on forum)
- Guest Lectures
  - Large companies, startups, researchers, etc.
- Readings from Industry
- Material from
  - Prem Devanbu
  - Christian Kästner
  - Marouane Kessentini
  - Claire Le Goues



Changeups and Trivia

- “[Professors who] deliberately and consistently interspersed their lectures with … some other form of deliberate break … usually commanded a better attention span from the class, and these deliberate variations had the effect of postponing or even eliminating the occurrence of an attention break”

[Johnstone and Percival. Attention breaks in lectures. Education in Chemistry, 13. 49-50, 1976.]

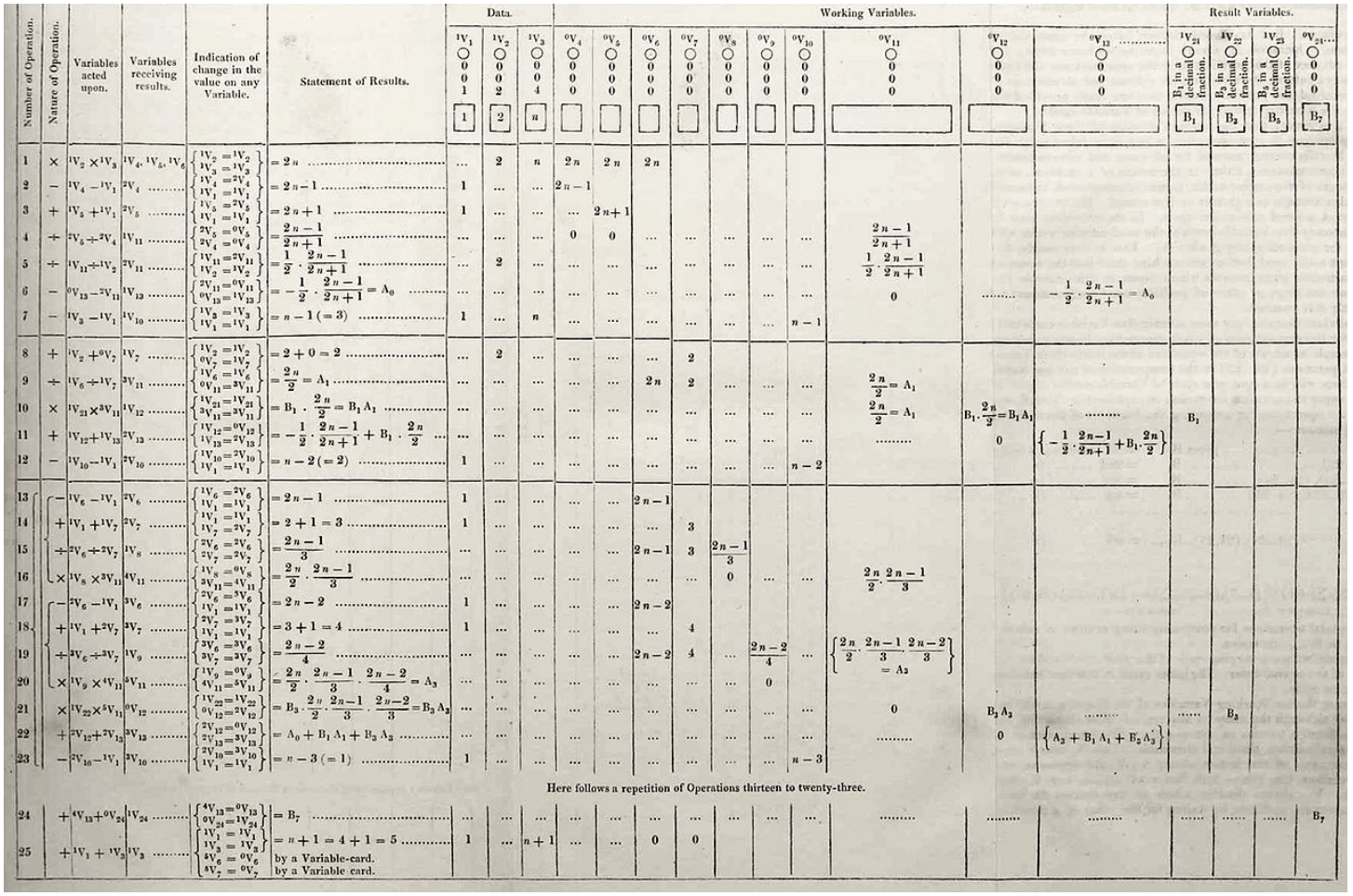
[Middendorf and Kalish. The “Change-up” in Lectures. TRC Newsletter, 8:1 (Fall 1996).]

Computer Science

- *This* English mathematician and writer published the first algorithm (~1842) to be carried out by a general-purpose computer and is often called the first computer programmer.

Computer Science

- *What* did that first program do?



Number of Operation.	Nature of Operation.	Variables acted upon.	Variables receiving results.	Indication of change in the value on any Variable.	Statement of Results.	Data. $^1V_1$ 1	$^1V_2$ 2	$^1V_3$ n	Working Variables. $^0V_4$	$^0V_5$	$^0V_6$	$^0V_7$	$^0V_8$	$^0V_9$	$^0V_{10}$	$^0V_{11}$	$^0V_{12}$	$^0V_{13}$	Result Variables. $^1V_{21}$ B_1	$^1V_{22}$ B_2	$^1V_{23}$ B_5	$^0V_{24}$... B_7
1	×	$^1V_2 \times {}^1V_3$	$^1V_4, {}^1V_5, {}^1V_6$	$\begin{cases} {}^1V_2 = {}^1V_2 \\ {}^1V_3 = {}^1V_3 \end{cases}$	$= 2n$	...	2	n	$2n$	$2n$	$2n$											
2	−	$^1V_4 - {}^1V_1$	$^2V_4$	$\begin{cases} {}^1V_4 = {}^2V_4 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= 2n-1$	1	...	...	$2n-1$													
3	+	$^1V_5 + {}^1V_1$	$^2V_5$	$\begin{cases} {}^1V_5 = {}^2V_5 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= 2n+1$	1	...	...	...	$2n+1$												
4	÷	$^2V_5 \div {}^2V_4$	$^1V_{11}$	$\begin{cases} {}^2V_5 = {}^0V_5 \\ {}^2V_4 = {}^0V_4 \end{cases}$	$= \dfrac{2n-1}{2n+1}$	...	...	...	0	0	...	...	...	...	...	$\dfrac{2n-1}{2n+1}$						
5	÷	$^1V_{11} \div {}^1V_2$	$^2V_{11}$	$\begin{cases} {}^1V_{11} = {}^2V_{11} \\ {}^1V_2 = {}^1V_2 \end{cases}$	$= \dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1}$	...	2	...	...	...	...	...	...	...	...	$\dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1}$						
6	−	$^0V_{13} - {}^2V_{11}$	$^1V_{13}$	$\begin{cases} {}^2V_{11} = {}^0V_{11} \\ {}^0V_{13} = {}^1V_{13} \end{cases}$	$= -\dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1} = A_0$	...	...	...	...	...	...	...	...	...	...	0	.....	$-\dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1} = A_0$				
7	−	$^1V_3 - {}^1V_1$	$^1V_{10}$	$\begin{cases} {}^1V_3 = {}^1V_3 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= n-1\,(=3)$	1	...	n	...	...	...	...	...	...	$n-1$							
8	+	$^1V_2 + {}^0V_7$	$^1V_7$	$\begin{cases} {}^1V_2 = {}^1V_2 \\ {}^0V_7 = {}^1V_7 \end{cases}$	$= 2+0=2$	...	2	...	...	...	...	2										
9	÷	$^1V_6 \div {}^1V_7$	$^3V_{11}$	$\begin{cases} {}^1V_6 = {}^1V_6 \\ {}^0V_{11} = {}^3V_{11} \end{cases}$	$= \dfrac{2n}{2} = A_1$	...	...	...	...	...	$2n$	2	...	...	...	$\dfrac{2n}{2} = A_1$						
10	×	$^1V_{21} \times {}^3V_{11}$	$^1V_{12}$	$\begin{cases} {}^1V_{21} = {}^1V_{21} \\ {}^3V_{11} = {}^3V_{11} \end{cases}$	$= B_1 \cdot \dfrac{2n}{2} = B_1 A_1$	...	...	...	...	...	...	...	...	...	...	$\dfrac{2n}{2} = A_1$	$B_1 \cdot \dfrac{2n}{2} = B_1 A_1$	.....	B_1			
11	+	$^1V_{12} + {}^1V_{13}$	$^2V_{13}$	$\begin{cases} {}^1V_{12} = {}^0V_{12} \\ {}^1V_{13} = {}^2V_{13} \end{cases}$	$= -\dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1} + B_1 \cdot \dfrac{2n}{2}$ ...	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{-\dfrac{1}{2} \cdot \dfrac{2n-1}{2n+1} + B_1 \cdot \dfrac{2n}{2}\right\}$				
12	−	$^1V_{10} - {}^1V_1$	$^2V_{10}$	$\begin{cases} {}^1V_{10} = {}^2V_{10} \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= n-2\,(=2)$	1	...	...	...	...	...	...	...	...	$n-2$							
13	−	$^1V_6 - {}^1V_1$	$^2V_6$	$\begin{cases} {}^1V_6 = {}^2V_6 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= 2n-1$	1	...	...	...	...	$2n-1$											
14	+	$^1V_1 + {}^1V_7$	$^2V_7$	$\begin{cases} {}^1V_1 = {}^1V_1 \\ {}^1V_7 = {}^2V_7 \end{cases}$	$= 2+1=3$	1	...	...	...	...	...	3										
15	÷	$^2V_6 \div {}^2V_7$	$^1V_8$	$\begin{cases} {}^2V_6 = {}^2V_6 \\ {}^2V_7 = {}^2V_7 \end{cases}$	$= \dfrac{2n-1}{3}$	...	...	...	...	...	$2n-1$	3	$\dfrac{2n-1}{3}$									
16	×	$^1V_8 \times {}^3V_{11}$	$^4V_{11}$	$\begin{cases} {}^1V_8 = {}^0V_8 \\ {}^3V_{11} = {}^4V_{11} \end{cases}$	$= \dfrac{2n}{2} \cdot \dfrac{2n-1}{3}$	...	...	...	...	...	...	...	0	...	...	$\dfrac{2n}{2} \cdot \dfrac{2n-1}{3}$						
17	−	$^2V_6 - {}^1V_1$	$^3V_6$	$\begin{cases} {}^2V_6 = {}^3V_6 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= 2n-2$	1	...	...	...	...	$2n-2$											
18	+	$^1V_1 + {}^2V_7$	$^3V_7$	$\begin{cases} {}^2V_7 = {}^3V_7 \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= 3+1=4$	1	...	...	...	...	...	4										
19	÷	$^3V_6 \div {}^3V_7$	$^1V_9$	$\begin{cases} {}^3V_6 = {}^3V_6 \\ {}^3V_7 = {}^3V_7 \end{cases}$	$= \dfrac{2n-2}{4}$	...	...	...	...	...	$2n-2$	4	...	$\dfrac{2n-2}{4}$	...	$\left\{\dfrac{2n}{2} \cdot \dfrac{2n-1}{3} \cdot \dfrac{2n-2}{3} = A_3\right\}$						
20	×	$^1V_9 \times {}^4V_{11}$	$^5V_{11}$	$\begin{cases} {}^1V_9 = {}^0V_9 \\ {}^4V_{11} = {}^5V_{11} \end{cases}$	$= \dfrac{2n}{2} \cdot \dfrac{2n-1}{3} \cdot \dfrac{2n-2}{4} = A_3$ ...	...	...	...	...	...	...	...	...	0	...							
21	×	$^1V_{22} \times {}^5V_{11}$	$^0V_{12}$	$\begin{cases} {}^1V_{22} = {}^1V_{22} \\ {}^0V_{12} = {}^2V_{12} \end{cases}$	$= B_3 \cdot \dfrac{2n}{2} \cdot \dfrac{2n-1}{3} \cdot \dfrac{2n-2}{3} = B_3 A_3$	...	...	...	...	...	...	...	...	...	...	0	$B_3 A_3$	.....	.....	B_3		
22	+	$^2V_{12} + {}^2V_{13}$	$^3V_{13}$	$\begin{cases} {}^2V_{12} = {}^0V_{12} \\ {}^2V_{13} = {}^3V_{13} \end{cases}$	$= A_0 + B_1 A_1 + B_3 A_3$	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{A_3 + B_1 A_1 + B_3 A_3\right\}$				
23	−	$^2V_{10} - {}^1V_1$	$^3V_{10}$	$\begin{cases} {}^2V_{10} = {}^3V_{10} \\ {}^1V_1 = {}^1V_1 \end{cases}$	$= n-3\,(=1)$	1	...	...	...	...	...	...	...	...	$n-3$							
Here follows a repetition of Operations thirteen to twenty-three.																						
24	+	$^4V_{13} + {}^0V_{24}$	$^1V_{24}$	$\begin{cases} {}^4V_{13} = {}^0V_{13} \\ {}^0V_{24} = {}^1V_{24} \end{cases}$	$= B_7$	...	...	...	...	...	...	...	...	...	...	.....	.....	.....	.....	.....	.....	B_7
25	+	$^1V_1 + {}^1V_2$	$^1V_3$	$\begin{cases} {}^1V_1 = {}^1V_1 \\ {}^1V_2 = {}^1V_2 \\ {}^4V_5 = {}^0V_5 \\ {}^4V_7 = {}^0V_7 \end{cases}$ by a Variable-card. by a Variable-card.	$= n+1 = 4+1 = 5$	1	...	$n+1$	...	...	0	0										

Who Cared?
What was one “killer app”?

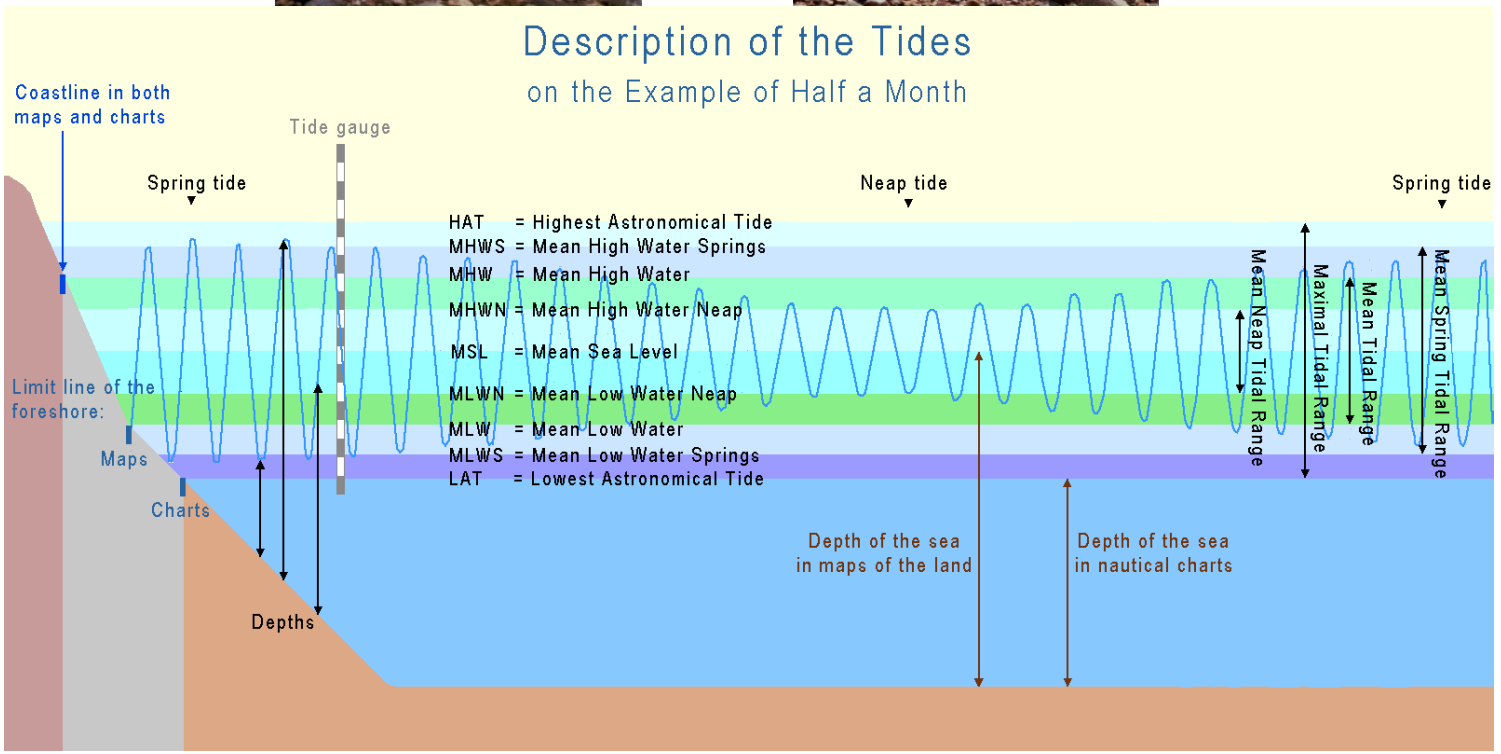
A hundred years later, the early Victorians used not Newton's theory of the tides but the variant that Bernoulli worked out for this competition, which they referred to as the equilibrium theory. The equilibrium theory outlined the essential mathematical formulas needed to predict the tides for each day of the year based on the positions of the sun and moon. The most important variable for each port was the "vulgar establ-

to calculate the important variables specific to each port. Observations of the tides were a valued commodity, however, and once taken they were guarded as private property. The inaccessibility of Bernoulli's methods

mer,'" published more than ninety years later. Bernoulli's treatise significantly advanced the methods of tidal prediction, but using those methods still demanded both theoretical refinement and intense and laborious calculations. Moreover, to find the corrected establishment, one also

[Michael S. Reidy. *Tides of History: Ocean Science and Her Majesty's Navy.* From Chapter 1, “Tidal Prediction After Newton and Halley”]

“Amazon Prime” circa 1842



Psychology: The Fundamental Attribution Error

- The **fundamental attribution error** is that people emphasize internal characteristics when explaining the behavior of others but external factors when explaining their own behavior.
  - Example: cutting someone off in traffic.
- In an experiment, subjects read essays for and against Fidel Castro and were asked to rate the pro-Castro attributes of the writers. Conditions:
  - When subjects believed the writers choose freely:
    - Expect "pro-Castro" → positive attitude
  - When subjects believed the positions were determined by a coin toss:
    - Expect neutral attitude on average

Psychology: The Fundamental Attribution Error

- Experimental findings:
  - Even when they knew the position came from a coin toss, subjects rated pro-Castro essay writers as having a positive Castro attitude.
  - “The subjects were unable to properly see the influence of the situational constraints placed upon the writers; they could not refrain from attributing sincere belief to the writers.”

[Jones, E. E.; Harris, V. A. (1967). "The attribution of attitudes". Journal of Experimental Social Psychology. 3 (1): 1–24.]

- SE Implication: Teamwork. Be careful when you see defects (mine just mean I made a typo, others mean they are stupid).

Core Course Topics

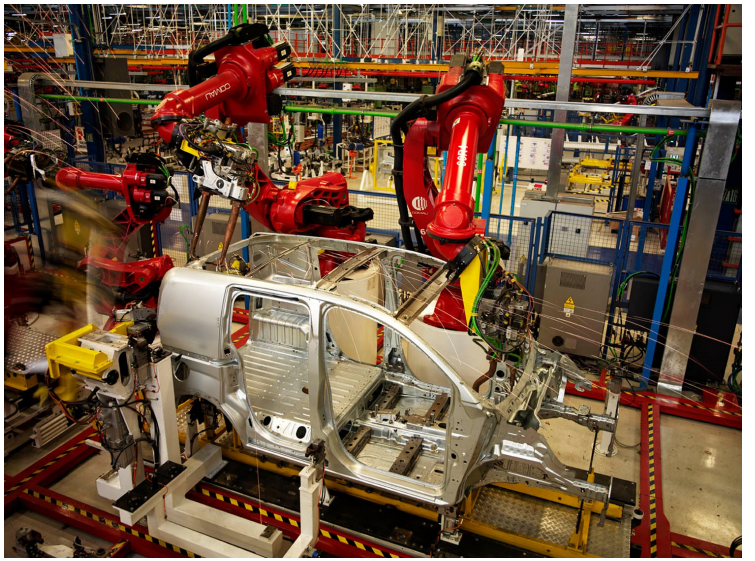
- Measurement and Risk
  - Process, scheduling, and information
- **Quality Assurance**
  - Code review, **testing**, and analysis
- Software Defects
  - Reporting and localizing
- Software Design
  - Requirements, patterns, and maintainability
- Productivity at Scale
  - People, teams, interviews, and brains

Course Themes

- Software engineering is a human process
- Software engineering deals with large scales
- Software engineering requires strategic thinking
- Software engineering is constrained by reality

Analogy: Engineering Envy

- Producing a car
  - Estimate costs, risks
  - Expected results
  - High quality
- Separate plan and production
- Simulate before constructing
- Quality assurance through measurement
- Potential for automation



Dangerous Analogy

- Producing a car
  - Estimate costs, risks
  - Expected results
  - High quality
- Separate plan and production
- Simulate before constructing
- Quality assurance through measurement
- Potential for automation

- Software = Design = Plan
- Programming is design, not production
  - Production (copying/loading a program) is automated
  - Simulation is not necessary
- Quality measurement?

Software Engineering

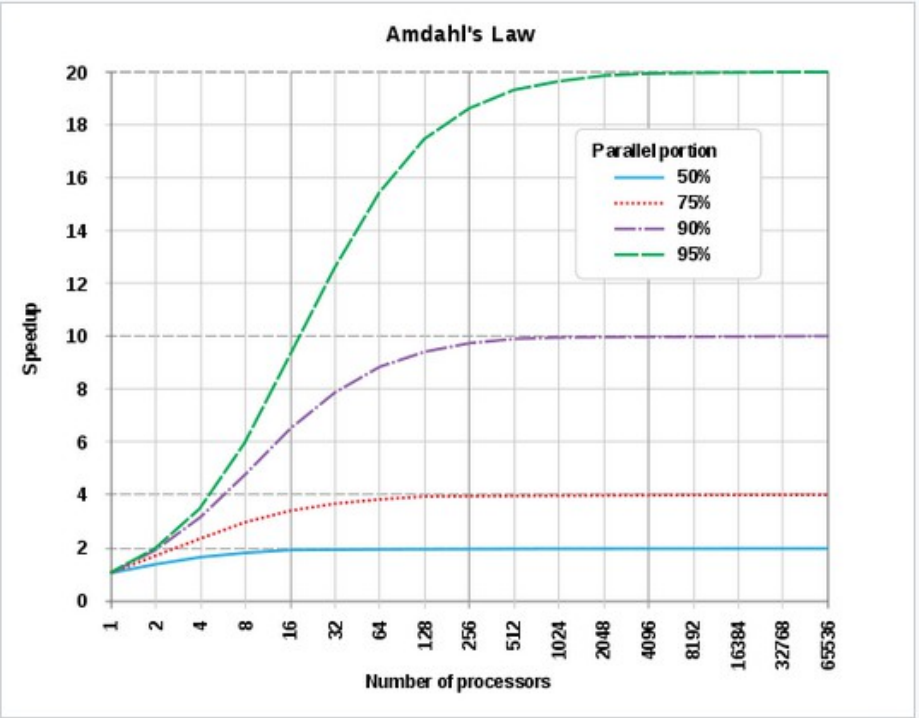
“My favorite operational definition of engineering is '**design under constraint**.' Engineering is creating, designing what can be, but it is constrained by nature, by cost, by concerns of safety, reliability, environmental impact, manufacturability, maintainability, and many other such 'ilities.'”

[Bill Wulf, NAE President, The Urgency of Engineering Education Reform, 2008]

“[Software Engineering is] The Establishment and use of sound **engineering principles** in order to obtain **economically** software that is **reliable** and works **efficiently** on **real** machines.”

[Bauer 1975, S. 524]

Measurement Teaser



Evolution according to Amdahl's law of the theoretical speedup in latency of the execution of a program in function of the number of processors executing it, for different values of p. The speedup is limited by the serial part of the program. For example, if 95% of the program can be parallelized, the theoretical maximum speedup using parallel computing would be 20 times.

- What is Amdahl's Law?
- Suppose you want a program to run faster
- Suppose you want software to be created-and-sold faster

Quality Assurance Teaser

- To assess quality, we can look at the source code or run the program
- Testing is the dominant approach here
- But not all test suites are created equal!
- Statement coverage, branch coverage

- Mutation testing
- Automated test generation

Defect Teaser

- Just put in print statements
- Find the line with the bug
- Flail around, resubmit until it passes

- Automatic fault localization
- Debugging as Hypothesis Testing

Design Teaser

- Requirements and Specifications
  - How can we elicit what people actually want?
- Validation and Risk
- Design for Maintainability

Productivity Teaser

- The ratio of programming time and program performance between novices and experts has been published at up to 28:1
  - Why?

- Pair Programming, Agile, etc.
- How do experts and novices think?
- Medical Imaging Studies

Questions?

- No discussion sections or office hours this week!