Towards a History of Model-Modellee Conflations in Computer Science

Edgar G. Daylight — Siegen University — 8 Feb. 2018

Lille Workshop:

Models betw. structures and meanings of programs

"Engineers who acknowledge limits to technology are all too easily cast as pessimists or failures."

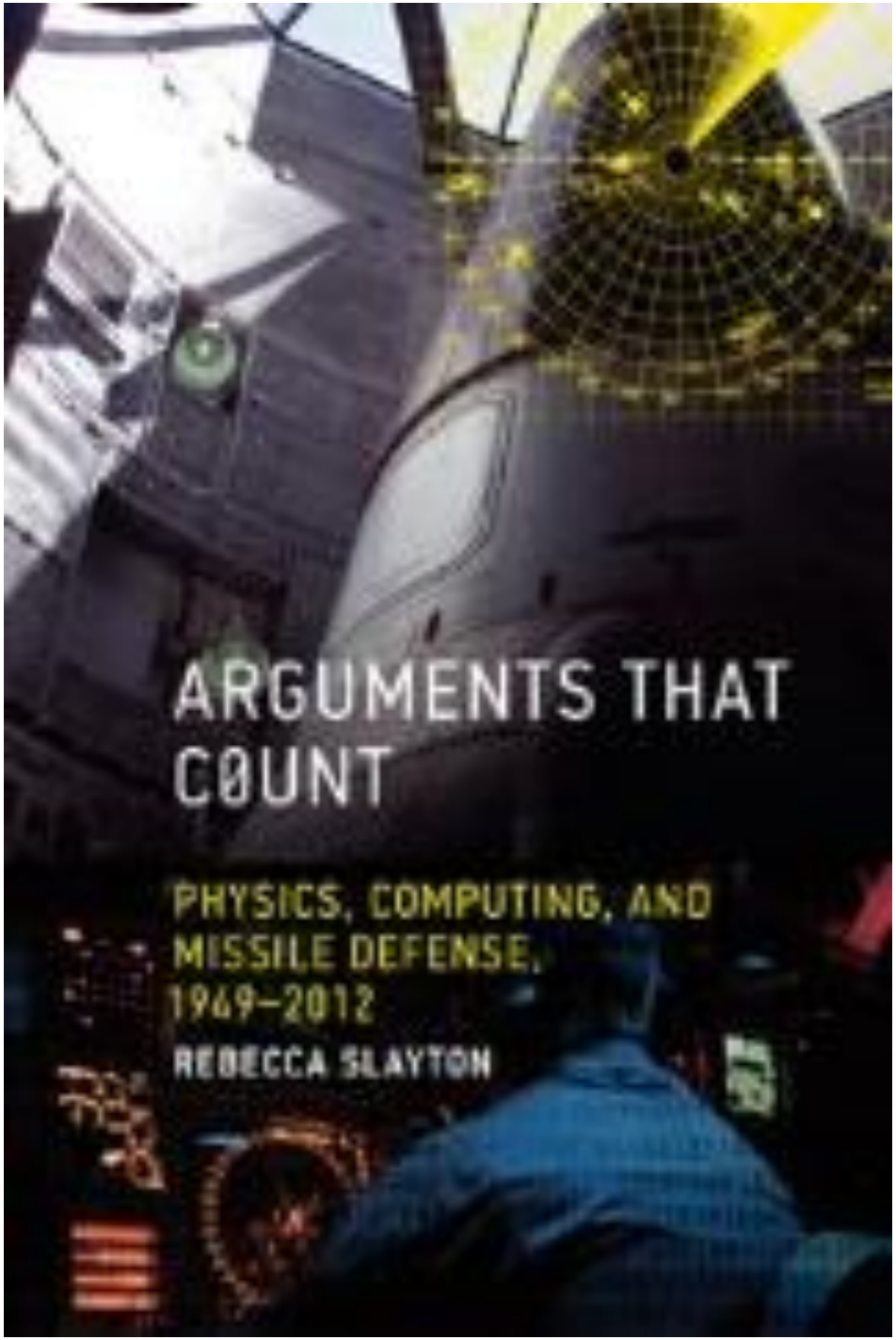
[Slayton, 2013, p.11]

Outline

1. A Bird’s Eye View
2. A Worm’s Eye View
3. Intermezzo
4. My Personal Contribution
5. Terminology & Methodology

A Bird’s Eye View

What Can We Learn from History?


ARGUMENTS THAT
C0UNT
PHYSICS, COMPUTING, AND
MISSILE DEFENSE,
1949–2012
REBECCA SLAYTON

“By the time physicists began to note the limitations of software, the missile defense program was moving forward with a momentum all its own.”

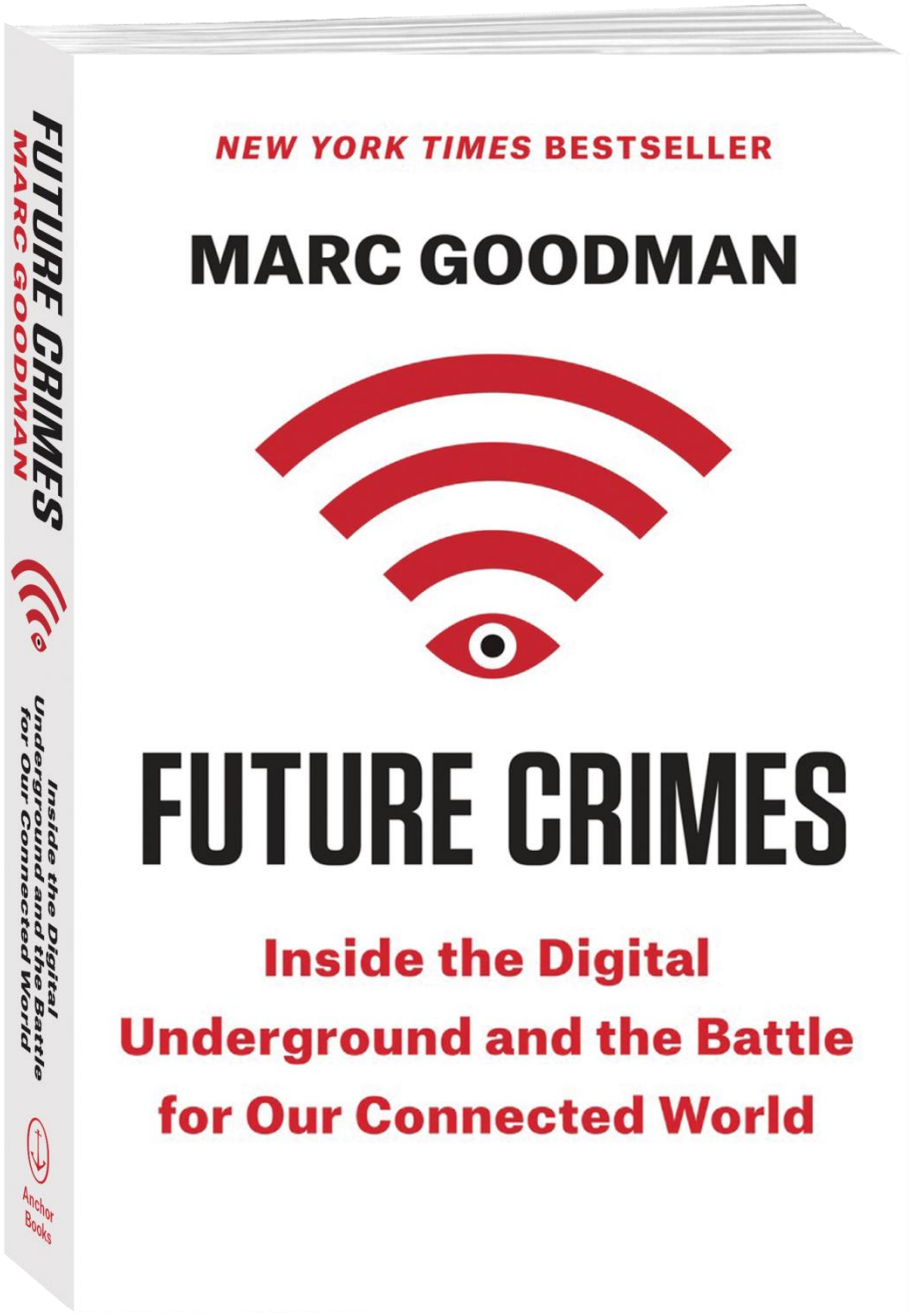
[Slayton, 2013, p.84]

< missing picture about human-machine symbiosis >

Does History Repeat Itself?

“Historians assess past examples of change to better understand change in society today.”

[Stearns, 1998]


NEW YORK TIMES BESTSELLER
MARC GOODMAN
FUTURE CRIMES
Inside the Digital Underground and the Battle for Our Connected World
FUTURE CRIMES
MARC GOODMAN
Inside the Digital Underground and the Battle for Our Connected World
Anchor Books

“Dozens of demonstrations by hackers and security researchers have proven it is entirely possible for criminals 1500 miles away to seize control of your car when you are driving 65 miles per hour down the highway.”

[Goodman, 2016, p.363]

Governments and industries simply do not want to "miss the bandwagon of tests, first deployments, and perhaps manufacturing too."

[De Grave, 2016]

Perhaps There is No Technological Fix

In the *Communications of the ACM:*

"A steady march toward the automated car is clearly under way"

[Casner et al., 2016]

Can History Help to Prevent Mistakes?

A historical approach is needed that renders errors visible and gives us a broader, more contextual view of software and its actual and potential development, and the implications this has for science and society.

Without knowledge of the past, there is no future.

Can Historians Help to Prevent Mistakes?

— Yes we can!

A Worm’s Eye View

< missing pictures of Ronald Reagan and the Star Wars project, available on the Internet >

Slayton's Narrative Suggests that There is No Technological Fix for Missile Defense

“Most of the many published proofs of programs [in computer science] are actually proofs about models of programs, models that ignore the very properties of digital computers that cause many of the `bugs’ we are trying to eliminate.”

[Parnas, 2012]

Begin of Intermezzo

Computer scientists,
have an “inclination to slide, in their discourse, between programs, models, and theories as if there was no distinction between them.”

[Moor, 1978]

Computer scientists — and relatively few software engineers and security experts — believe in a **Turing Fix** in that they, often heedlessly, treat a laptop *as* a Turing machine or a computer program *as* a Turing machine program in their research, thereby downplaying the modelling activity at hand.

Conflations in various fields:

Linguistics: mistaking sentences deduced from a formal grammar for a natural language [Schüttpelz, 1996, Ch.3].

Cognitive studies: equating the computation of functions with actual cognition [Fetzer, 1998, p.378].

Conflations in various fields:

Computability: confusing a Turing machine with a stored-program computer [Daylight, 2014].

Software engineering: mistaking a C computer program for a system realization [Lee, 2015, p.4839].

Conflations in various fields:

Database management: “representations and reality are conflated” by computer scientists, “it is difficult to comprehend privacy in any terms other than control over access to *individually identified* data records” [Agre, 1997].

Conflations in various fields:

Database management: “representations and reality are conflated” by computer scientists, “it is difficult to comprehend privacy in any terms other than control over access to *individually identified* data records” [Agre, 1997].

Formal verification:
confusing verification of mathematical models with truth [De Millo et al, 1979].

"Models often have properties real mechanisms do not have, and it is possible to verify the correctness of a model of a program even if the actual program will fail."

[Parnas, 2012]

End of Intermezzo

Is There a Technological Fix for Internet-Connected Pacemakers?

< missing cartoon about pacemakers, available on the Internet >

Stakeholders who (implicitly) believe in a foolproof mapping — that is, who *conflate* the security model of the pacemaker's software with the deployed software itself (= modellee) — will put too much weight on the "security proofs" obtained by researchers.

What is missing in the current state of the art is …

… an in-depth history of software and its mathematical underpinnings

Too Much Technical Jargon for Policymakers

Error-free software is a necessary (but not a sufficient) condition for having a secure — and, hence, a humanly safe — Internet-connected, cyber-physical system.

www.dijkstrascry.com/Lee1

OVERARCHING RESEARCH QUESTION

How did software specialists technically cope with the problem of having software errors (in deployed systems) during the course of history, from the 1950s till today?

Answering this Research Question is Urgent

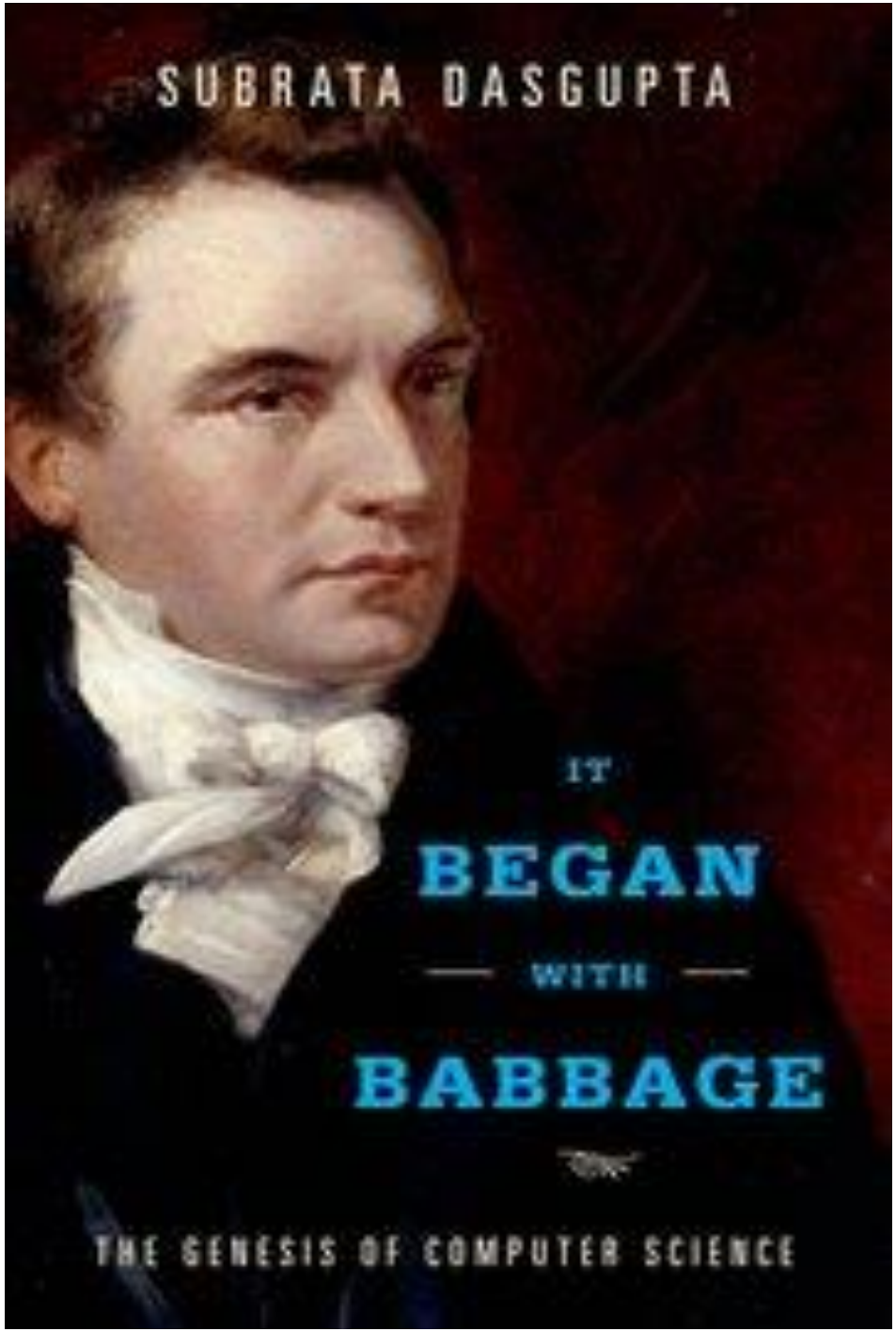
My Personal Contribution

(forthcoming)

Fill two research gaps:

1. Computer science history *pur sang*

2. The socio-technical history of software engineering


SUBRATA DASGUPTA
IT
BEGAN
WITH
BABBAGE
THE GENESIS OF COMPUTER SCIENCE

For Computer Scientists, the Role of Mathematical Logic is that of a Queen

Courtesy of Michael A. Jackson

For Software Engineers, the Role of Mathematical Logic is that of a Humble Servant

Courtesy of Michael A. Jackson

Understanding technical developments of software engineering for drones, cars, and pacemakers requires comprehension of computer science's decades-old appropriation of ideas from mathematical logic in their continual move away from the real, physical world.

After all, logico-mathematical rigor in computer science has led to the development of software analysis tools that engineers use daily.

The Practice of Conflating is Both Powerful & Troubling

1) Mathematical rigor in the science of computer programming

…

2) Ignorance of the model's *limited regime of applicability* …

Courtesy of Edward A. Lee

< missing picture about human-machine symbiosis >

Terminology & Methodology

Computing vs. Computer Science

Computing vs. Computer Science

The **history of computing** includes the history of the abacus, punched card machines, personal computers, computer games, the computing industry, and so forth.

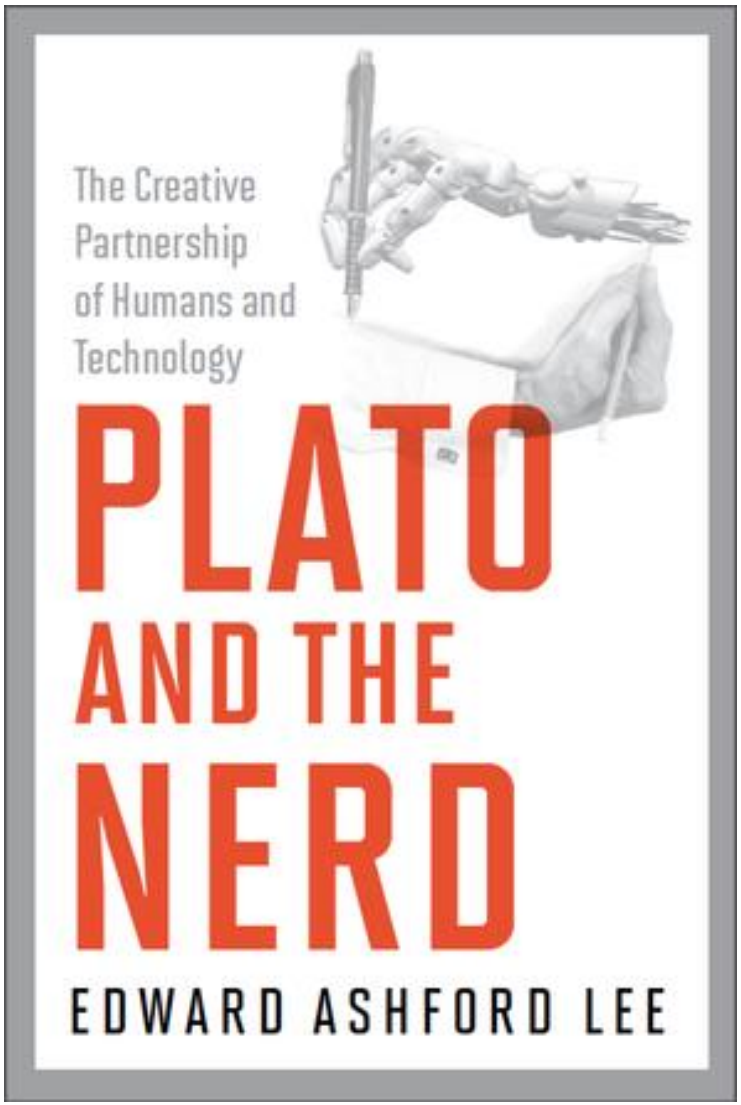
Computing vs. Computer Science

The **history of computing** includes the history of the abacus, punched card machines, personal computers, computer games, the computing industry, and so forth.

While these kinds of histories have been published, computing history's modern chapter on **computer science** has yet to be drafted, let alone written.

Computer Science vs. Software Engineering

Computer Science vs. Software Engineering



Computer Science vs. Software Engineering

- The ***Scientific Mechanism*** amounts to choosing a model that is faithful to the modellee.

Computer Science vs. Software Engineering

- The ***Scientific Mechanism*** amounts to choosing a model that is faithful to the modellee. A scientist studies nature — and a computer scientist studies the nature of the artificial (e.g. computers) — and subsequently chooses or invents a model that s/he deems is useful.

Computer Science vs. Software Engineering

- The ***Scientific Mechanism*** amounts to choosing a model that is faithful to the modellee. A scientist studies nature — and a computer scientist studies the nature of the artificial (e.g. computers) — and subsequently chooses or invents a model that s/he deems is useful.
  - E.g., the laws of classical mechanics (= model) are an attempt to capture and predict motions in our physical world (= modellee).

Computer Science vs. Software Engineering

- The ***Scientific Mechanism*** amounts to choosing a model that is faithful to the modellee. A scientist studies nature — and a computer scientist studies the nature of the artificial (e.g. computers) — and subsequently chooses or invents a model that s/he deems is useful.
  - E.g., the laws of classical mechanics (= model) are an attempt to capture and predict motions in our physical world (= modellee).

- The ***Engineering Mechanism*** amounts to producing a modellee that is faithful to the model.
  - E.g.: a software engineer uses a flowchart (= model) to produce a program (= modellee).

The Symmetry Principle

The Symmetry Principle

"I am simply attending with equal curiosity to both sides of the dispute and reporting how the historical actors saw one another."

[Bloor, 2014, p.404]

The Symmetry Principle (Bycroft,2016)

1. Should not assume in advance of empirical inquiry that:
   a. true beliefs are best explained rationally
   b. false beliefs are best explained irrationally

2. Says nothing against the practice of internal history of science

Thank You!

Thank You!

"Engineers who acknowledge limits to technology are all too easily cast as pessimists or failures."

[Slayton, 2013, p.11]

Sources

- **[Agar, 2013]** Agar, `review of Michael Sean Mahoney; Thomas Haigh, eds. *Histories of Computing*,' Isis, 104:4, 868–869, 2013.
- **[Agre, 1997]** Agre, `Beyond the Mirror World: Privacy and the Representational Practices of Computing,' Chapter in: `Technology and Privacy: The New Landscape,' MIT Press, 1997.
- **[Bloor, 2014]** Bloor, `Reply to Christopher Norris,' Journal of Critical Realism, 13:4, 399–410, 2014.
- **[Bullynck et al., 2015]** Bullynck, Daylight, De Mol. 'Why did Computer Science Make a Hero out of Turing?', CACM, 58:3, 37–39, 2015.
- **[Bycroft, 2016]** Bycroft, `How to Save the Symmetry Principle,' in: Sauer and Scholl (Eds.), `The Philosophy of Historical Case Studies,' Springer, 2016.
- **[Carrier & Nordman, 2011]** Carrier, Nordman (Eds.), `Science in the Context of Application,' Springer, 2011.
- **[Casner et al., 2016]** Casner, Hutchins, Norman, `The Challenges of Partially Automated Driving,' CACM, 59:5, 70–77, 2016.
- **[Colburn, 2000]** Colburn, `Philosophy and Computer Science,' M.E. Sharpe, 2000.
- **[Cook et al., 2011]** Cook, Podelski, Rybalchenko, 'Proving Program Termination,' CACM, 54:5, 88–98, 2011.
- **[Dasgupta, 2014]** Dasgupta, `It Began with Babbage: The Genesis of Computer Science,' Oxford University Press, 2014.
- **[Daylight, 2014]** Daylight, 'A Turing Tale,' CACM, 57:10, 36–38, 2014.
- **[Daylight, 2015]** Daylight, 'Towards a Historical Notion of "Turing — the Father of Computer Science,"' History & Philosophy of Logic, 36:3, 205–228, 2015.
- **[Daylight, 2016]** Daylight, `Turing Tales,' Lonely Scholar, 2016.
- **[De Millo et al, 1979]** De Millo, Lipton, Perlis, `Social Processes and Proofs of Theorems and Programs,' CACM, 22:5, 271-280, 1979.
- **[De Grave, 2016]** De Grave, `Networking self-driving cars,' comment on the PI's blog from a specialist in the automotive industry — www.dijkstrascry.com/comment/2232#comment-2232 — 2016.
- **[Dear, 2006]** Dear, `The Intelligibility of Nature: How Science Makes Sense of the World,' The University of Chicago Press, 2006.
- **[Dijkstra, 1973]** Dijkstra, 'On the Axiomatic Definition of Semantics,' EWD 367 — www.cs.utexas.edu/users/EWD/index03xx.html — 1973.
- **[Fetzer, 1998]** Fetzer, `People are not computers: (most) thought processes are not computational procedures,' Journal of Experimental and Theoretical Artificial Intelligence, 10:4, 371–391, 1998.
- **[Golomb, 1971]** Golomb, `Mathematical models: Uses and limitations," IEEE Transactions on Reliability, 20:3, 130–131, 1971.
- **[Goodman, 1960]** Goodman (ed.), 'Annual Review in Automatic Programming I,' Pergamon Press, 1960.
- **[Goodman, 2016]** Goodman, `Future Crimes: Inside the Digital Underground and the Battle for our Connected World,' Corgi Books, 2016.
- **[Haigh, 2015]** Haigh, 'The Tears of Donald Knuth: Has the History of Computing Taken a Tragic Turn?', CACM, 58:1, 40–44, 2015.
- **[Harari, 2017]** Harari, `Reboot for the AI revolution,' Nature, 550, 324–327, 2017.
- **[HoC, 2010s]** Reference to the following sources on the History of Computing:
  - **[Campbell-Kelly et al., 2014]** Campbell-Kelly, Aspray, Ensmenger, Yost, 'Computer: A History of the Information Age,' 3rd ed., Westview Press, 2014.
  - **[Ceruzzi, 2012]** Ceruzzi, `Computing: A Concise History,' MIT Press, 2012.
  - **[Ensmenger, 2010]** Ensmenger, 'The Computer Boys Take Over: Computers, Programmers, and the Politics of Technical Expertise,' MIT Press, 2010.
  - **[Haigh et al., 2016**] Haigh, Priestley, Rope, `ENIAC in Action: Making and Remaking the Modern Computer,' MIT Press, 2016.
- **[Huisman et al., 2016]** Huisman, Bos, Brinkkemper, Van Deursen, Groote, Lago, Van de Pol, Visser, `Software that meets its intent,' In: Margaria & Steffen (Eds.), *Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications*, LNCS, Vol 9953. Springer, 2016.

Sources

- **[Jackson, 2015]** Jackson, Daylight, `Formalism and Intuition in Software Development,' Lonely Scholar, 2015.
- **[J-B, 2008]** Jasanoff's review of Bakale's book `Designs on Nature,' Political and Legal Anthropology Review, 31:1, 157–160, 2008.
- **[Johnson, 2011]** Johnson, `Everything New Is Old Again: What Place Should Applied Science Have in the History of Science?,' in [Carrier & Nordmann, p.455–466].
- **[Keen, 2017]** Keen Security Lab of Tencent, `New Car Hacking Research: 2017, Remote Attack Tesla Motors Again,' — video available at *https://keenlab.tencent.com/en/* — 2017.
- **[Knuth, Daylight, 2014]** Knuth, Daylight, 'Algorithmic Barriers Falling: P = NP?', Lonely Scholar, 2014.
- **[Kroening, Strichman, 2008]** Kroening, Strichman, 'Decision Procedures: An Algorithmic Point of View,' Springer, 2008.
- **[Kurzweil, 2005]** Kurzweil, `The Singularity is Near,' Viking Books, 2005.
- **[Lagesen, 2007]** Lagesen, `The Strength of Numbers: Strategies to Include Women into Computer Science,' Social Studies of Science, 37, 67–92, 2007.
- **[Lee, 2015]** Lee, `The past, present and future of cyber-physical systems: A focus on models,' Sensors, 15:3, 4837-4869, 2015.
- **[Lee, 2017]** Lee, `Plato and the Nerd: The Creative Partnership of Humans & Technology,' MIT Press, 2017.
- **[Leveson, 2012]** Leveson, `Engineering a Safer World: Systems Thinking Applied to Safety,' MIT Press, 2012.
- **[Lindqvist et al., 2017]** Lindqvist, Neumann, `The Future of the Internet of Things,' CACM, 60:2, 26-30, 2017.
- **[MacKenzie, 2004]** MacKenzie, `Mechanizing Proof: Computing, Risk, and Trust,' MIT Press, 2004.
- **[Mahoney, 2011]** Mahoney, (Haigh, Ed.), 'Histories of Computing,' Harvard University Press, 2011.
- **[Mearian, 2018]** Mearian, `Here's Why Self-Driving Cars May Never Really Be Self-Driving,' Computerworld — computerworld.com/article/3171160/car-tech/heres-why-self-driving-cars-may-never-really-be-self-driving.html — Accessed on 27 January 2018.
- **[MOHL, 1973]** Van der Poel and Maarssen (Eds.), *`Machine Oriented Higher Level Languages (MOHL),'* IFIP Working Conference on MOHL, Trondheim (Norway), 1973.
- **[Moor, 1978]** Moor, 'Three Myths of Computer Science,' British Journal for the Philosophy of Science, 1978.
- **[Morris, 1893]** Morris, `Introduction,' in: Steele, `Medieval Lore,' E. Stock, London, 1893.
- **[Naur, 1992]** Naur, 'Computing: A Human Activity,' ACM Press, 1992.
- **[Nofre et al., 2014]** Nofre, Priestley, Alberts, 'When Technology Became Language: The Origins of the Linguistic Conception of Computer Programming, 1950–1960,' Technology & Culture, 55:1, 40–75, 2014.
- **[Nofre, 2015]** Nofre's book review of [Dasgupta, 2014] in: Technology & Culture, 56:2, 537–538, 2015.
- **[Parnas, 1985]** Parnas, `Software Aspects of Strategic Defense Systems,' CACM, 28:12, 1326–1335, 1985.
- **[Parnas, 2012]** Parnas, `The use of mathematics in software quality assurance,' Frontiers of Computer Science in China, 6:1, 3–16, 2012.
- **[Parnas, 2012]** Parnas, `On Proving Continuity of Programs,' in: Letter to the Editor of the Comm. of the ACM, 55:11, 9, 2012.
- **[Priestley, 2011]** Priestley, 'A Science of Operations,' Springer, 2011.
- **[Schüttpelz, 1996]** Schüttpelz, `Figuren der Rede: Zur Theorie der rhetorischen Figur,' Eric Schmidt Verlag GmbH & Co., 1996.
- **[Slayton, 2013]** Slayton, `Arguments that Count: Physics, Computing, and Missile Defense, 1949–2012,' MIT Press, 2013.
- **[Somers, 2017]** Somers, `The Coming Software Apocalypse,' The Atlantic — www.theatlantic.com/technology/archive/2017/09/saving-the-world-from-code/540393/ — 26 Sep. 2017.
- **[Standaard, 2017]** De Standaard, `Duizenden pacemakers kwetsbaar voor hacking,' —www.standaard.be/cnt/dmf20170901_03048305 — 2 September 2017.
- **[Standaard, 2018]** De Standaard, `Bekijk uw medisch dossier online,' — www.standaard.be/cnt/dmf20180102_03276243 — 2 January 2018.

Sources

- **[Stearns, 1998]** Stearns, `Why Study History? (1998),' American Historical Association — historians.org/about-aha-and-membership/aha-history-and-archives/historical-archives/why-study-history-(1998) — Accessed on 28 January 2018.
- **[The Economist, 2017]** Two entries in The Economist:
  - `Europe is trying to keep Russia from influencing its elections,' — www.economist.com/news/europe/21720665-france-and-germany-fear-propaganda-and-espionage-favouring-pro-kremlin-candidates-europe-trying — 15 April 2017.
  - `The crooked timber of humanity,' by Tom Standage — www.1843magazine.com/technology/rewind/the-crooked-timber-of-humanity — 5 October 2017.
- **[Turner, 2013]** Turner, 'The Philosophy of Computer Science,' Stanford Encyclopedia of Philosophy, 2013.
- **[Vanhoef et al., 2017]** Vanhoef, Piessens, `Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2,' ACM SIGSAC Conference on Computer and Communications Security 2017, 1313–1328, ACM, 2017.