Where Dinosaurs Roam and Programmers Play: Reflections on Infrastructure, Maintenance, and Inequality

Jeffrey R. Yost, Charles Babbage Institute, University of Minnesota

In March 1959 Burroughs Corporation computer scientist Mary Hawes called for an industry and government consortium in order to develop a standard programming language for business—promoting greater portability for organizational users transitioning mainframe computers. With appearances of Autocode, FLOW-MATIC, FORTRAN, ALGOL-58, and other 1950s programming languages, she recognized the high costs of proliferation.

Rear Admiral Grace Murray Hopper was an unparalleled leader in the early software field. In addition to her pioneering work with the A-0 compiler, her FLOW-MATIC was particularly influential. More than any other language, FLOW-MATIC provided a model for the COBOL development team. (Image: United States Navy)

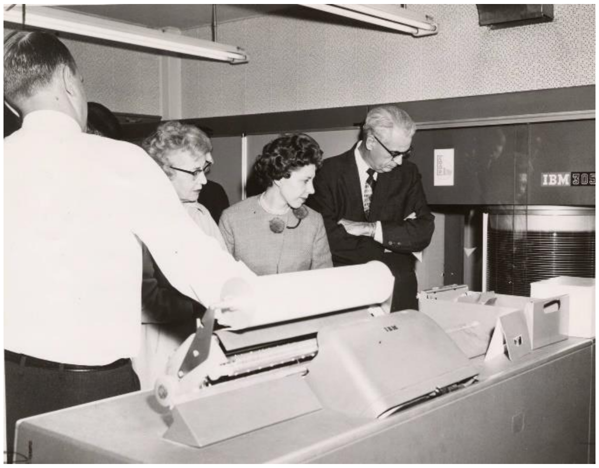
Jean Sammet, COBOL Co-developer. She was the first woman president of ACM, a visionary leader. Sammet had a long and distinguished career at IBM. Despite prolific achievements and stellar intellectual and managerial contributions to her company and her field, she was not named an IBM Fellow, an honor granted disproportionately to men—approximately 90% of the 275 IBM Fellows are male (Image: Charles Babbage Institute Archives)

The following month, Hawes' call evolved into the founding Conference on Data Systems Languages (CODASYL), sponsored by the U.S. Department of Defense (DoD). CODASYL's ongoing efforts drew inspiration from Sperry-Univac's Grace Murray Hopper, her FLOW-MATIC, and her advocacy for languages approximating English syntax. For these important contributions, Hopper is sometimes referred to as the "Mother of COBOL," (Common Business-Oriented Language). Despite some crediting her as its "inventor," a CODASYL committee of six—Howard Bromberg (RCA), Howard Discount (RCA), Vernon Reeves (Sylvania), Jean Sammet (Sylvania, joined IBM in 1961), William Seldon (IBM), and Gertrude Tierney (IBM)—developed COBOL. DoD published COBOL 60 specifications in January 1960, eight iterations followed, the most recent in 2014. It began as a standard for the DoD and became a standard of American National Standards Institute (ANSI), and International Organization of Standardization (ISO). Today, though still widespread global technologies, common descriptors for mainframes and COBOL are dinosaurs and dinosaur code.

Still Coding After All These Years

To highlight COBOL's staying power, and perhaps glimpse into its future, in 2014 the Defense Contract Management Agency (DCMA) stated it was not looking to replace its system composed of two million lines of COBOL code (handling 330,000 contracts worth $1.2 trillion), but re-upping on COBOL. DCMA put out a statement "bragging" its new COBOL system would "probably be around for another 20 to 30 years" (Mazmanian 2014).

Back in 2004, IT research firm Gartner, Inc. had estimated there were two million programmers knowledgeable in COBOL—eight percent of all programmers globally—but that the number was decreasing at five percent a year (King, 2020).



IBM 305, Random Access Memory Accounting, the first hard disk drive system, developed in 1956, offering ability for real-time accounting. Real-time was both a challenge and opportunity for COBOL programmed systems that tended to be batch/remote batch. (Image: Charles Babbage Institute Archives)

Early Quincy ATM by Burroughs Corporation. COBOL code was and remains the backbone of ATM transaction processing. (Image: Charles Babbage Institute Archives).

Today, almost half of banks in the U.S. run systems programmed in COBOL, and 95 percent of all ATM transactions rely on COBOL (Allyn, 2020)—trillions every day. Even in normal times, demand for COBOL experts exceeds supply.

Feeding the Beast

From the 1960s into the 1990s, many universities offered COBOL courses, as did companies and vocational schools like Control Data Institutes. Today, in an age where AI/analytics, games, robotics, cloud, and the internet of things are foremost for many computer science students, few consider learning legacy systems and legacy languages. Accordingly, COBOL courses are scarce. A *Slate* article quoted Prof. John Zeanchock, Robert Morris University, stating just 37 colleges and universities globally have a "mainframe course" on the curriculum. Most schools' faculty are unable to suggest legacy specialist students/graduates when banks or local governments call. (Botella 2020). In our culture, *Innovation* is revered, and *maintenance* is not. In IT there is a myopic attention to the latest tech and a failure to recognize and value that IT maintenance requires great skill and can be innovative (new processes, new fixes, etc.). Privileging innovation over maintenance is also in part tied to gender stereotypes and discrimination as historically women have had greater opportunity in the critical areas of services, maintenance (both machines and debugging), and programming (from plug board to languages), and fewer opportunities in computer and software engineering (Yost, 2011, 2017). The percentage of women majors in computer science declined sharply the past quarter century—from more than 35 percent in the 1980s to 18.1 percent in 2014, only varying slightly since (nsf.gov/statistics). The reasons are varied, but gender stereotyping, a male dominant computing culture, and educational and workplace discrimination are factors (Abbate, 2012; Hicks, 2017; Misa, 2011). This has furthered labor shortages (all areas, including legacy) and held back computer science. Labor shortages can become all the more profound in times of crisis, including the current health and economic crisis.

Women students at PLATO terminals at University of Illinois in 1963. Women's participation in CS as majors increased as the field gained traction in the 1960s and 1970s, and percentage of women peaked in the 1980s. Since then there was a sharp decline to a low plateau. This lack of gender diversity holds back CS and IT labor in all specializations, including legacy. (Image: Charles Babbage Institute Archives).

More than a Jersey Thing

On April 6, 2020, New Jersey Governor Phil Murphy made a public plea for volunteer "Cobalt" programmers (meaning COBOL) to aid New Jersey and help with glitches to an overburdened unemployment benefits computer system more than 40 years old. New Jersey was having difficulties with timely processing of unemployment payments to the flood of new filers. The increased burden (volume and parameters) on the unemployment system was a major bottleneck, or to borrow Thomas Hughes' term, reverse salient, to timely and accurate data processing to respond to those in need (Hughes, 1983).

This sparked an onslaught of journalist articles as well many Twitter, Facebook and other social media posts. The critiques ranged from Governor Murphy/New Jersey having an antiquated unemployment insurance computer system to calling for volunteers from a population segment that would undoubtedly be the most susceptible to COVID-19 risk—the elderly. Meanwhile, social media erupted with jokes with ageist images of elderly individuals as potential volunteers.

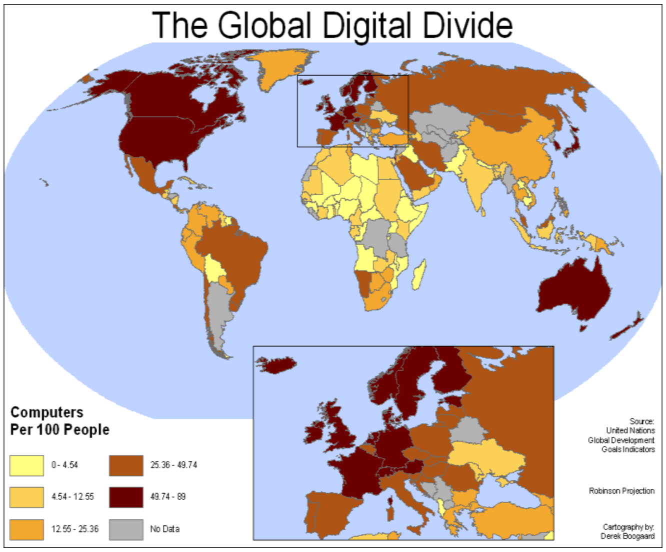
Both university courses and those at IT vocational schools like Control Data Institutes (CDI) were critical to teaching a generation of COBOL programmers in the 1960s and 1970s—many now retired. Here we see a CDI classroom in 1967. (Image: Charles Babbage Institute Archives)

Other states, including Connecticut and Kansas, had similar shortages of trained COBOL experts to confront unemployment insurance system challenges. Understandably, unemployed workers waiting for unemployment benefits are extremely frustrated and angry, expressing as much on the Kansas Department of Labor (KDoL) platform. Much *is the matter* with Kansas' system, with its origins in the 1970s, and inadequate updates for flexibility and scale. In late April, KDoL indicated a timeline where processing could occur by late May (for many that will push the wait to *months*). For states that have prioritized investing in updating other computer systems, but not unemployment insurance, it amounts to neglecting infrastructure that serves the most vulnerable in society.

Why do so many states have ill-equipped IT systems for unemployment benefits processing? Replacing long existing systems is complex and expensive (hundreds of millions of dollars). Change is also disruptive to existing labor, existing skill sets. Unemployment systems serve those lacking political power; federal and state governments deprioritize them. Further, systems (in all their technical, political, economic, and other contexts) become entrenched, or to use Hughes' concept, gain *momentum* (Hughes, 1983). Failures/pressures can redirect momentum, some states scrambled for cloud solutions once systems crashed in April—possibly the least bad option, but also suboptimal timing, new systems and processes on the fly are especially difficult. Regardless, the problem is one of infrastructure—not valuing maintenance, labor, and recipients. It is not merely COBOL versus the cloud, in fact, COBOL can and does integrate with AWS, Azure, and IBM clouds; hybrid cloud is common.

State IT Workers and Hired Guns' Heroic Efforts

North Texas' COBOL Cowboys staffing firm, larger IT services enterprises, and COBOL-skilled independent contractors are in great demand. The governors, state DoLs, and state CIOs are doing their best to staff up to address problems. For the systems analysts, programmers, and other state employees and contractors the hours are long, work difficult, and efforts truly heroic. The Federal CARES Act's unemployment benefits, PUA/PECU, allows states to extend the duration of benefits, and include those usually not eligible—the self-employed. This adds greatly to *both* volume and complexity. In my playful title, "play" is used for where work plays/is performed (fewer coders choosing legacy) and to highlight coders' creativity—in the spirit of CS metaphors like "sandbox" for building (non-live) code.


The Global Digital Divide
Computers Per 100 People
0 - 4.54
4.54 - 12.55
12.55 - 25.36
25.36 - 49.74
49.74 - 89
No Data
Source: United Nations Global Development Goals Indicators
Robinson Projection
Cartography by: Derek Boogaard

As this United Nations' graphic shows, the global digital divide is profound. Ramifications during a global pandemic are extreme, where digital connectivity influences opportunities to shelter, connect, and safely earn income. This map is not intended to and does not capture the deep digital divide in the U.S., along class and race lines. (Image: Dakman5, granting public domain rights, Wikicommons)

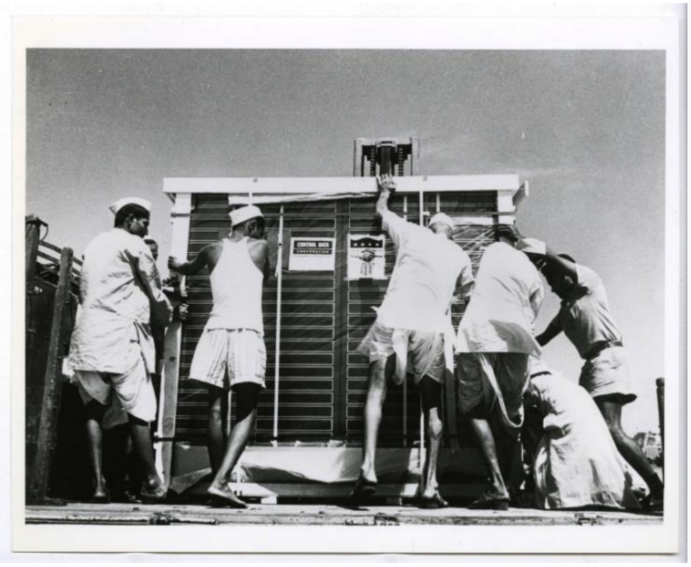
Domestic and Global Digital Divides

In the coming year, the overall percentage of Americans below the poverty line will peak higher than any time in more than 50 years—the impact for African-American, Hispanic, and Native-American populations is particularly severe. The disparity of access to health insurance, banking, loans, and information technology, as well as exposure to risk, and disparity of incidence and mortality with COVID-19, highlights extreme and growing race and class inequality in the United States.

Washington D.C.'s unemployment platform urges benefits filers to use Microsoft Explorer. Microsoft retired Explorer in January 2016, an unsupported version remains for computers, not smart phones. A Pew Research Center 2019 survey showed 54 percent of Americans under $30,000/year income have a computer, while 71 percent have a smart phone. For those making over $100,000, 94 percent have a computer/broadband at home. (Anderson and Kumar, 2019). Only 58 percent of African-Americans have a computer, versus 82 percent for whites. (Perrin and Turner, 2019) In digital, just like education, healthcare, housing, and other infrastructure, there are two Americas.

Y2K: Why to Care

An earlier crisis largely involving COBOL, one with a long and visible runway, is both consequential context and instructive to current challenges. About a quarter century ago, governments and corporations began seriously addressing the pending Y2K crisis—caused by two digits for date often in COBOL code—to avert risks to life and the economy, to make it a nonevent. Investments and global cooperation were key and the International Y2K Cooperation Center played a meaningful role in fostering collaboration. The shortage of programmers knowledgeable in COBOL, and the lower expense and overwhelming volume of code, led to outsourcing to an emerging Indian IT services industry. This lent momentum to this trade, and to a shifting geography in IT work that remains impactful (though corporate decision-makers are accelerating artificial intelligence applications producing further labor transformations, ones detrimental to Indian IT laborers, developments standout ABD sociologist and CBI IDF Fellow Devika Narayan is insightfully analyzing). Gartner Inc. estimated U.S. government and business expenditures were up to $225 billion, a breathtaking sum indicative of costs of putting off maintenance until a time-sensitive crisis. In passing into the new millennium with few major problems, the situation lent credence to two diverging interpretations—that heavy investment in maintenance had been necessary to avert catastrophe, or more common (and less accurate), that it was an overhyped problem leading to squandered funds in preparing, in maintenances fixes. Offshoring saved money in the short run, but may not have in the longer run, it left a legacy of less and less current, on-shore COBOL expertise (for maintenance, updates, security, etc.), a workforce and talent helpful in global crises, particularly ones in which unfortunate (U.S.) nationalistic tendencies and policies have inhibited international cooperation.



Unloading a Control Data 3600 mainframe in 1964 at Tata Institute, Bombay, India. Both the IITs and Tata were fundamental IT educational and vocational infrastructure that allowed a major software and services industry to prosper in the 1990s with COBOL Y2K compliance work and much more. (Image: Charles Babbage Institute Archives)

CONNECT and Disconnects

Maintaining infrastructure is important. Anemic IT budgets have not only hurt opportunities to change and move to innovative new solutions, but also to best maintain existing systems and to better assure their ability to perform and to perform to scale in both normal times and crises. The reverse salient certainly is not always COBOL or COBOL alone. State auditors warned Florida Governor Ron DeSantis that Florida's unemployment site, its "CONNECT" cyberinfrastructure, had more than 600 systems errors in need of fixing, but that state officials had "no process to evaluate and fix." (Mower, 2020). It was a 2013 $77 million system, which he is quick to point out, his administration inherited. This underlines the challenge not just in Florida but many States—inadequate infrastructure is the predecessors' fault, is not the current leaders' problem, and fixes should lie with successors. Often the (now) multi-hundred million-dollar cost typical of major upgrades to new unemployment insurance systems (and ongoing refinement) is difficult without federal assistance. Florida's CONNECT is a reminder of damaging disconnects, and leaders' inattention to infrastructure for vulnerable people. The problem is also one of meager and dwindling federal support. Federal aid for state unemployment administration has been dropping for a quarter century with severe cuts in 2018 and 2019. In a survey (pre-COVID-19) more than half of the states responded their unemployment system problems were "serious" or "critical." (Botella 2020).

Neglected Infrastructure and Crashes

Working two tenths of a mile from the site of the 2007 Interstate 35 West Mississippi River Bridge collapse, in Minneapolis, is a frequent reminder that strong, safe, and well-maintained infrastructure is essential. Twenty-eight percent of infrastructure project funding at the state level comes from federal grants (primarily for physical infrastructure). States' invisible software infrastructure is starved, especially unemployment systems. Hopefully the COVID-19 pandemic leads not only to evaluating our medical preparedness with ICUs, PPE, and unmet needs in free enterprise insurance and healthcare, but also greater evaluation of IT infrastructures. Ideally, the developments will lead all governors with poor unemployment insurance system performance to the same conclusion as Governor Murphy about the need for post-mortems on digital infrastructure. As he put it "how the heck did we get here when we literally needed COBOL programmers"— learning from the past is important.

Minneapolis Interstate 35W Bridge collapse, August 2007. Physical infrastructure gets far more federal support for states than ethereal software infrastructure. Both require evaluation, audits/checks, and timely maintenance—or they break—for software in the form of crashes, delays, breaches, etc. (Image: Kevin Rofidal, United States Coast Guard. Wikicommons. Public domain USCG Image. 17 U.S.C. § 101 and § 105).

History Matters

One thing clear from the two COBOL crises is that history and archives matter—my thoughts here have at best just scratched the surface on fundamental IT infrastructure and contexts someone could analyze with tremendous depth using Charles Babbage Institute resources. CBI's archival and oral history resources (most transcripts online, all free) to study the Y2K crisis and the history of CODASYL and COBOL (and many other topics and themes in the history and social study of computing) are the finest and most extensive in the world. A talented University of Pennsylvania doctoral candidate in the History and Sociology of Science, Zachery Loeb, has drawn on CBI's International Y2K International Cooperation Center Records for his important dissertation on the cultural, political, and technical history of Y2K.

Control Data Corporation CEO William Norris signing an agreement in 1960 with NBS. NBS was critical in 1959 and 1960, along with sponsoring DoD, to CODASYL launch and COBOL's development and prevalence as an industry standard. (CBI Archives)
(Image: Charles Babbage Institute Archives)

Over the years, a number of researchers have used our Conference on Data Systems Languages (CODASYL) Records. While it stands out on documenting COBOL and the group's work with databases (what occurred in 1959 and far beyond), we have many other COBOL materials in a variety of collections. One such collection (a recent one) is our largest overall collection at more than 500 linear feet, the Jean Sammet Papers—Sammet may have been the single most important developer with COBOL. Likewise, our Frances E. ("Betty") Holberton Papers has rich material on CODASYL and COBOL. There is also great COBOL content in our Burroughs Corporate Records, Control Corporation Records, Gartner Group Records, Auerbach and Associates Market and Product Reports, IBM SHARE, Inc., HOPL 1978, Charles Phillips Papers, Jerome Garfunkel Papers, Warren G. Simmons Papers, National Bureau of Standards Computer Literature, Computer Manuals, and many other collections. COBOL's history is one of government, industry, and intermediaries' partnerships, standards, maintenance, labor, gender, politics, culture and much more. In a technical area that always seems focused on the new, new thing, its 60-year past and its continuing presence deserve greater study.

Sources:

Abbate, Janet. *Recoding Gender: Women's Changing Participation in Computing* (MIT, 2012).
Allyn, Bobby "COBOL Cowboys Aim to Rescue the Sluggish State Unemployment Systems. "*NPR* (April 22, 2020).
Anderson, Monica and Madhumitha Kumar. "Digital Divide Persists…" *Pew Research Center* (May 7, 2020).
Botella, Ella. "Why New Jersey's Unemployment System Uses a 60-Year-Old Programming Language." *Slate* (April 9, 2020).
Charles Babbage Institute Archives (finding aids to the collections mentioned in final paragraph).
Hicks, Marie. *Programmed Inequality: How Britain Discarded Women Technologists and Lost Its Edge in Computing* (MIT Press, 2017).
Hughes, Thomas P., *Networks of Power*: *Electrification in Western Society, 1880 to 1930* (Johns Hopkins University Press, 1983).
Kennelly, Denis, "Three Reasons Companies are only 20% Into Cloud Transformation." IBM.com (March 5, 2019).
King, Ian. "An Ancient Computer System is Slowing Giant Stimulus." *Bloomberg.com* (April 13, 2020).
Mazmanian, Adam "DoD Plans Upgrade to COBOL-based Contract System" *FCW* (July 7, 2014).
Misa, Thomas J., ed. *Gender Codes: Why Women are Leaving Computing* (Wiley, 2011).
Mower, Lawrence, "Ron DeSantis…"*Tampa Bay Times* (March 31, 2020).
Perrin, Andrew and Erika Turner "Smartphones Help Blacks and Hispanics Bridge Some—But Not All—Digital Gaps with Whites," Pew Research Center (August 20, 2019).
Yost, Jeffrey R. "Programming Enterprise: Women Entrepreneurs in Software and Computer Services," in Misa, ed. *Gender Codes* (2011) [full cite above].
Yost, Jeffrey R., *Making IT Work: A History of the Computer Services Industry* (MIT Press, 2017).

Special thanks to CBI Acting Archivist Amanda Wick for discussion/insights on COBOL and our collections.

Jeffrey R. Yost, "Where Dinosaurs Roam and Programmers Play: Reflections on Infrastructure, Maintenance, and Inequality." *Interfaces: Essays and Reviews on Computing and Culture* (Charles Babbage Institute, University of Minnesota, May 2020): 1-8.

About the author: Jeffrey R. Yost is CBI Director and HSTM Research Professor at the University of Minnesota. He has published six books (and dozens of articles), most recently *Making IT Work: A History of the Computer Services Industry* (MIT Press, 2017) *and FastLane: Managing Science in the Internet World* (Johns Hopkins U. Press, 2016) [co-authored with Thomas J. Misa]. He is a past EIC of *IEEE Annals of the History of Computing*, and current Series Co-Editor [with Gerard Alberts] of Springer's History of Computing Book Series. He has been a principal investigator of a half dozen federally sponsored projects (NSF and DOE) on computing/software history totaling more than $2 million. He is Co-Editor [with Amanda Wick] of *Interfaces: Essays and Reviews in Computing & Culture*. (yostx003@umn.edu).