If Guinness World Records gave out an award for prolonged futility in successfully developing a US state government IT system, then South Carolina’s Department of Social Services would surely be a contender for the “honor,” in recognition of its protracted attempt to create an automated child-support enforcement system (CSES). The system, which is intended to aid court-ordered child-support collections, was part of a federal government mandate under the Family Support Act of 1996.1 Originally, all states were required to have a federally certified CSES in place by October 1995; however, when only Montana met the deadline, Congress extended it to October 1997 without imposing a financial penalty.1 Currently, South Carolina’s CSES is scheduled to become fully operational by October 2019—assuming nothing else untoward happens.2

The state’s lengthy, tortuous path to creating a CSES began in early 1992,3 and it’s been downhill ever since. The first attempt, which missed both the October 1995 and 1997 deadlines, ended in a bitter lawsuit between South Carolina and its CSES developer, Unisys Corporation, which wasn’t resolved until late 2001. That effort cost $34.7 million. The next attempt began in 2008, but it too ended in a rancorous lawsuit between the state and its system developer, the Hewlett-Packard Company, which was finally settled in early 2015. This second effort cost $108 million. South Carolina’s latest CSES implementation attempt, this time involving the Xerox Corporation, is expected to cost $140 million when (or if) it goes operational as planned in October 2019.

The total projected cost of South Carolina’s CSES will thus exceed $280 million (with US taxpayers on the hook for $218 million of that amount under federal-state cost-sharing agreements).4 Furthermore, the state faces a cumulative federal fine for missing the October 2007 deadline extension that’s estimated to reach $200 million (with $61 million of that paid by the previous two CSES vendors as part of the lawsuit settlements).5

On top of this staggering measurable cost for failing to develop a working CSES are the immeasurable yet very painful personal and financial burdens felt by the untold number of families in the state who over the past two decades didn’t receive their court-ordered child-support payments because of the absent system.6 Consider also the opportunity costs incurred by both South Carolina and US taxpayers who’ve been paying for the repeated, expensive foul-ups. The only true beneficiaries seem to be the system developers who didn’t deliver.


THE CONSEQUENCES OF IT GONE WRONG

South Carolina’s sorrowful saga is just one among some 200 troubled IT projects and operational systems around the world that I, along with Josh Romero, IEEE Spectrum’s former senior interactive editor, captured in a September 2015 Spectrum special online interactive report titled “Lessons from a Decade of IT Failures” (spectrum.ieee.org/static/lessons-from-a-decade-of-it-failures). Through our report, we wished to visually convey some of the many negative consequences of IT-related mishaps, whether developmental or operational (excluding cybercrime events), since the magazine’s special issue on software failure a decade earlier (spectrum.ieee.org/magazine/2005/September).

We further wanted to find out if there were any discernible patterns to these myriad incidents. Therefore, we structured the dataset to be viewed from multiple perspectives such as magnitude of impact (monetary cost, duration, number of people affected, and so on), region (for example, the Americas, Asia Pacific, and EuMEA—Europe, the Middle East, and Asia), type of problem, type of organization, and industry area. We were also interested in any new lessons to be learned about the causes or sources of IT failures since those explored in the Spectrum special issue 10 years earlier.

Even given the data’s limitations—other than government audit reports and occasional court filings from lawsuits, there’s sparse detailed, reliable, and verifiable information on IT failures—some interesting observations can be made.

One is the growing cost of such failures. An article I wrote for the 2005 Spectrum special issue titled “Why Software Fails” listed some of the most prominent IT fiascoes since 1992. Although the list contained a handful of billion-dollar boondoggles, notably the aborted IRS Tax Systems Modernization program ($4 billion) and Federal Aviation Administration System ($2.6 billion) in the US, the vast majority of individual IT project failure costs were under $500 million. However, our 2015 review found a dozen or more developmental IT failures in the past decade in the billion-dollar range, including the UK Department of Health’s National Programme for IT (NPfIT) electronic health record system ($20 billion) and the US National Polar-Orbiting Operational Environmental Satellite System (NPOESS) ($6 billion), with another seven efforts exceeding $500 million. This group doesn’t include exorbitantly expensive IT projects that are now operational like the US Air Force Defense Enterprise Accounting Management System (DEAMS) or Melbourne’s myki public transportation ticketing system, which incurred $1.1 billion and $800 million, respectively, in cost overruns.

Furthermore, there were numerous consequential IT operational failures. For example, in 2012, the Royal Bank of Scotland (RBS) suffered a computing system disruption that prevented 1.7 million customers from accessing their financial accounts for a week, and some customers were affected for more than six weeks. The ongoing cost of the incident including customer compensation, government fines, and IT system upgrades has topped $1.9 billion. Some operational IT failures affected millions of people around the world. In 2011, for instance, some 40 million BlackBerry users across Europe, the Middle East, and Africa encountered various service disruptions for three days due to a hardware issue, making it the fourth such outage in as many years. Additionally, small, localized outages...
environment is considerably changed from that of the past.

Alaska Airlines CEO Gary Kelly recently underscored this point: "Where once Alaska Airlines had essentially two computerized systems—its reservations platform, and the technology on each aircraft—it now has hundreds, controlling everything from maintenance to crew scheduling to its mobile app."8 A problem with any one of these mutually dependent systems might affect many others, as Alaska discovered in 2011 when a central computer outage took out its reservations system, which in turn affected its flight planning and other support systems. Some 150 flights were canceled, and more than 12,000 passengers' plans disrupted. Virgin Blue (now Virgin Australia) Airlines, British Airways, American Airlines, JetBlue, Southwest Airlines, and United Continental Airlines have all had their share of operational outages over the past decade, too.

Many of the IT operational failures in our Spectrum dataset can be traced to system upgrades or routine maintenance to legacy systems, some decades old, that didn’t go as expected. The need to modernize legacy IT systems has created plentiful opportunities for IT failure, especially in government. The RBS incident wasn’t an aberration: several other banks in the UK, US, Australia, and elsewhere have experienced repeated IT outages due to upgrades or maintenance issues along with failed backup systems. The recurring disruptions have motivated UK and Australian regulators to strongly urge their country’s banks to improve outdated IT systems.9,10 Stock exchanges around the world have also experienced outages due to system modifications—for instance, the New York Stock Exchange shut down for four hours in 2015 because of an unrecoverable bug in a software upgrade.

The need to modernize legacy IT systems has also created plentiful opportunities for IT failure, especially in government. South Carolina wasn’t alone in struggling to modernize its CSES; eight other states missed the extended 2007 deadline and were fined.11 Efforts to modernize other state government legacy IT systems, such as social services, motor vehicle registration, and unemployment insurance, have similarly been fraught with problems. For example, in 2013, California canceled its second major motor vehicle system modernization halfway through development, citing “a lack of progress” after six years and $135 million. Similarly, in 2014, New Jersey canceled its Consolidated Assisted Support System (CASS) after spending $118 million over six years.

Government IT modernization efforts in other parts of the world have been equally disappointing. For example, the UK government canceled a revamped national benefits payments system in 2006 after spending $267 million, and the current multibillion-dollar effort to replace it has seen at least $700 million written off in system rework so far. Likewise, the Netherlands Social Insurance Bank, which implements national insurance policies on behalf of the Dutch government, discarded a new $58 million benefits system in 2014 after persistent failures.

An emerging trend over the past decade has been national governmental efforts to modernize healthcare delivery and management through the use of IT by encouraging or mandating the use of electronic health records (EHRs) in hospitals and medical provider offices. Many of these efforts have run into serious trouble. In addition to the bungled UK NPfIT project, Australia, Canada, and the US each have had expensive EHR initiative failures.

More predictably, motor vehicles are increasingly a source of IT-related problems as software use in automobiles has exploded over the past decade. Typical was Toyota’s 2014 worldwide recall of 1.9 million Prius hybrids—nearly half of all sold—to correct a software flaw that could damage the vehicle’s engine. From 2011 to 2015, software-related recalls jumped from less than 5 percent to 15 percent and involved more than 13 million vehicles. Of the 189 software-specific recalls issued, nearly a quarter were for errors that raised the risk of crashing with injury.12

There have also been some very public human–machine interface incidents during the past 10 years. Some are attributable to operator error, such as when a “fat-fingered” Tokyo Stock Exchange trader tried to order shares totaling $617 billion—more than Sweden’s annual gross domestic product—in one day in 2014. Others were more subtle, such as the 2009 crash of Air France Flight 447, in which pilot confusion over how the Airbus 330-200 flight management system worked, coupled with poor warning system ergonomics and inadequate pilot training, conspired to bring down a perfectly flyable aircraft in bad weather. This incident, in conjunction with two other deadly 2009 accidents—a train-on-train collision in Washington, D.C.’s Metro system and the crash of Turkish Airlines Flight 195113—have raised questions about how much trust should be placed in automation. Such questions are starting to be raised about autonomously driven vehicles as well.

FORGETTING HARD-WON LESSONS

The aforementioned litany of IT developmental or operational failures is, of course, merely an outward symptom of deeper organizational, process, and
technology-related decisions gone awry. In my 2005 Spectrum article on software failures, I listed several common factors that typically unite to help create such failures such as stakeholder politics, unrealistic or unarticulated project goals, poor project oversight, unmanaged risks, and an inability to handle the project’s complexity, to mention but a few. Unsurprisingly, these factors haven’t changed.

One combination stands out in the 2015 dataset, however: the toxic merging of overweening political interference, quixotic project goals, and ignored risks, coupled with an abysmal memory of past IT failures. The data contain example after example of executive decision making perverted by what Bent Flyvbjerg, Massimo Garbuio, and Dan Lovallo call “delusional optimism.” The decisions underlying the IT failures Romero and I reviewed involved Icarus-like hubris, especially in government programs where the incentives to manage costs are low.

Take, for instance, the US Air Force Expeditionary Combat Support System (ECSS), which was finally terminated in 2012 after eight years and more than $1 billion worth of effort had produced no “significant military capability.” This was the Air Force’s third failed attempt to implement an integrated logistics system going back to the mid-1970s. Despite repeated pledges that they had learned from prior mistakes, especially with regard to being overly optimistic about what could be accomplished, when, and for how much money, ECSS managers still believed that they could successfully consolidate into a single IT system 28 times the number of systems any Department of Defense service or agency had previously tried to integrate. When the project inevitably failed yet again, the Air Force claimed the $1 billion spent was a good “first step” in learning how to build large, integrated systems and that it wasn’t “necessary” to hold anyone responsible for the fiasco.

A series of high-profile government IT system failures occurred in 2013. California, Florida, and Massachusetts unveiled new unemployment insurance payments systems, and North Carolina rolled out two new social benefits systems—all with significant problems. Officials in each state assured the public that their system was ready to be deployed despite strong internal signals—and in the case of North Carolina, explicit warnings from the state auditor—to the contrary. Predictably, but still dishearteningly, each system immediately suffered vexing operational malfunctions that lasted weeks or months and, in some cases, even longer. As a result, hundreds of thousands of vulnerable residents faced totally unnecessary financial and health risks for prolonged periods of time.

It would be easy to argue that these “go live before its time” executive decisions crossed into the realm of what Guy Adams and Danny Balfour characterize as “administrative evil”—unjustly or needlessly inflicting pain on people. In no instance were there circumstances sufficiently compelling to risk deploying an unready system.

Another case for administrative evil can be made for what has happened in Rhode Island. The state rolled out its $364 million Unified Health Infrastructure Project (UHIP) public assistance program in September 2016, years late and hundreds of millions of dollars over original cost estimates. One would think that government officials would have investigated what happened in North Carolina and other states with similarly troubled programs to avoid their fates. Nevertheless, officials decided to go live with the system even in the face of warnings from both the state auditor and the federal government about faulty code development practices, insufficient testing, and improperly functioning interfaces. Not surprisingly, UHIP suffered severe problems upon rollout that remain unresolved, causing financial hardship and healthcare disruptions for thousands of Rhode Island’s poorest residents. Pressed for an explanation, the director of the Department of Administration stated without irony, “Obviously, we’re not where we want to be, but this is actually where we expected we’d be.” The federal government, unamused, has threatened the state with heavy fines if the system isn’t fixed soon.

Lest one think this delusional mindset only affects government projects, consider recent IT debacles at Allied Irish Banks, Avantor Performance Materials, Bridgestone Corporation, British Gas, and ScanSource, which all ended in lawsuits and reveal neither world-class IT project management nor well-thought-out executive decision making.

Space limits me here to highlighting only a few of the many insights to be gained from a decade of IT failures, and I encourage readers to explore the online IEEE Spectrum data themselves. However, one final lesson must be mentioned: future IT systems are likely to be more interdependent and algorithmically complex, with society more dependent upon them to operate effectively, efficiently, securely, and safely—making system failures much more consequential. Thus, it’s time to treat their development and operations with the respect, resources, and professionalism they deserve. To paraphrase what I wrote a decade ago in “Why Software Fails,” we already know how to do software well. It’s way past time to act on what we know. If we continue to push our luck, the next decade of IT failures might create financial and moral consequences that society just can’t afford.

REFERENCES
AFTERSHOCK


The IEEE Computer Society is the world’s largest association of computing professionals and is the leading provider of technical information in the field.

MEMBERSHIP: Members receive the monthly magazine Computer, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.

OMBUDSMAN: Email ombudsman@computer.org.

COMPUTER SOCIETY WEBSITE: www.computer.org

Next Board Meeting: 30 January–3 February 2017, Anaheim, CA, USA

EXECUTIVE COMMITTEE

President: Jean-Luc Gaudiot; President-Elect: Hirofumi Kasahara; Past President: Roger U. Fujii; Secretary: Forrest Shull; First VP, Treasurer: David Lomet; Second VP, Publications: Gregory T. Byrd; Third VP, Education: Virginie Caubillon; Fourth VP, Technical Activities: Sumit S. Chakravarty; Fifth VP, Geographic Activities: Cecilia Metra; Sixth VP, Standards Activities: S.K. Ramesh; Seventh VP, Member & Geographic Activities: Cecilia Metra; VP, Professional & Educational Activities: Andy T. Chen; VP, Standards Activities: Jon Rosdahl; VP, Technical & Conference Activities: Haust A. Muller; 2017–2018 IEEE Director & Delegate Division VIII: Dejan S. Milojicic; 2016–2017 IEEE Director & Delegate Division V: Harold Javid; 2017 IEEE Director-Elect & Delegate Division V-Elect: John W. Walz

BOARD OF GOVERNORS

Term Expiring 2017: Alfredo Benso, Sy-Yen Kuo, Ming C. Lin, Fabrizio Lombardi, Haust A. Muller, Dimitrios Serpanos, Forrest J. Shull

Term Expiring 2018: Ann DeMarle, Fred Douglass, Vladimir Grekov, Bruce M. McMillin, Cecilia Metra, Kunio Uchiyama, Stefano Zanero

Term Expiring 2019: Saurabh Bagchi, Leila De Floriani, David S. Ebert, Jill I. Gostin, William Gropp, Sumi Helal, Avi Mendelson

IEEE Computer Society Office

Washington, D.C.:
107-0062, Japan • Phone: +81 3 3408 3118 • Email: help@computer.org

Asia/Pacific:
Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan • Phone: +81 3 3408 3118 • Email: help@computer.org

COMPUTER SOCIETY OFFICES

Phone: +1 202 371 0101 • Fax: +1 202 728 9614 • Email: help@computer.org
Los Alamitos: 10662 Los Vaqueros Circle, Los Alamitos, CA 90720
Phone: +1 714 821 8380 • Email: help@computer.org

MEMBERSHIP & PUBLICATION ORDERS
Phone: +1 800 272 6657 • Fax: +1 714 821 4641 • Email: help@computer.org

IEEE BOARD OF DIRECTORS

President & CEO: Karen Bartleson; President-Elect: James Jefferies; Past President: Barry L. Shoop; Secretary: William Walsh; Treasurer: John W. Walz; Director & President, IEEE-USA: Karen Pedersen; Director & President, Standards Association: Forrest Don Wright; Director & VP, Educational Activities: S.K. Ramesh; Director & VP, Membership and Geographic Activities: Mary Ellen Randall; Director & VP, Publication Services and Products: Samir El-Ghazaly; Director & VP, Technical Activities: Marina Ruggieri; Director & Delegate Division V: Harold Javid; Director & Delegate Division VIII: Dejan S. Milojicic

revised 2 Dec. 2016