service design collective

Success & Failure in Federal Service Delivery

Published July 28, 2023: Version 1.2
Service Design Collective, Inc.
A public benefit company
Contact: leadership.project@servicedesigncollective.com

Table of contents

Introduction

Service Design Collective interviewed more than 40 Federal government employees serving in a broad array of leadership and security positions between August 2022 and June 2023. Participants included present or past Assistants to the President, Directors and Deputy Directors of large Federal agencies, Chief Information, Innovation, and Operating Officers, Executive Directors, Heads of Compliance, Policy, and Transformation, Senior Advisors, Professional Staff Member, and team leads.

Nine are or were political appointees (with and without Senate confirmation) and eight were Senior Executives. Fourteen held the rank of GS-15, the highest level on the Federal Government's "General Schedule" pay band. The rest held senior roles at private technology companies or in Congress. All participants currently work for the Federal government or have worked directly with the government in the last two years.

Research participants described why delivering services in government is different from their private sector or personal experiences. Across interviews, people identified their own definitions and causes of failure in government programs. This report summarizes the most common observations and provides insights into what leaders did to successfully deliver products and services despite the challenges that government presents.

This research summary does not try to define all types of success and failure. Instead, we focus on the relationship that Federal Government projects have with success and failure when delivering public services. Observations are organized as answers to three questions:

- What does failure look like in the Federal Government?
- What causes government projects to fail?
- What increases the likelihood that a Federal project will be successful?

We asked research participants about their successes and failures when delivering large-scale, mission-critical programs, across multiple agencies in the U.S. Federal Government. Participants served in a wide variety of roles, including service delivery, policymaking, and oversight. We asked about the enablers and barriers to delivering public services and explored which environments fostered greater success.

We found that successful leaders often have at least a basic understanding of product management and enable success by setting iterative goals, tracking project-specific metrics, and removing team distractions. They leverage agile budget and contracting opportunities whenever possible, are not afraid to wade into policy discussions, and prioritize team morale. Many leaders relied on technology to deliver public services but treated technology as a means to deliver services and not a goal in-and-of-itself.

Most importantly, successful Federal leaders define success relative to project goals that impact people. They regularly communicated with their teams, contractors, policymakers, and the public. They use data to measure their progress and make critical decisions. They embrace transparency and accountability.

Traditional waterfall program management and its associated metrics–on time, on budget, and to contract specification–fails to track real-world impact because those metrics are not tied to service delivery success. Projects in government fail in many and sometimes predictable ways but success is rarely defined or talked about. Success is traditionally measured as simply a lack of failure. Unfortunately, government oversight agencies, Congress, and the press have focused heavily on government failures and spend little time documenting when and how the government achieves its goals. Extensive oversight creates a body of evidence that points almost solely to negative outcomes which has harmful effects on morale and, ironically, can incentivize behaviors that are more likely to end in failure.

This report is a work in progress and may be updated periodically to reflect new insights from additional research interviews. Quotes have been edited for clarity and anonymity.

What does failure look like in the Federal Government?

As many as 87% of large government projects fail. This often cited statistic comes from the Standish Group's 2015 Haze report, is included in The General Service Administration's De-risking technology guides, and used by organizations such as the Beeck Center for Social Impact and Innovation at Georgetown University. This statistic is based on a combination of "challenged projects," defined as projects that are over budget, late, and/or have an unsatisfactory target and "failed projects" that were either never completed, abandoned, or not used after completion.

Most of our interview participants defined failure in terms of outcomes. To them, failure meant government services and programs were difficult to navigate or delivered the wrong result. This contrasts with traditional government definitions of failure, which focuses on process. Traditionally, a project fails if it goes over budget and/or misses original timelines. Sometimes what is seen as a traditional failure, like canceling a project before it launches, is actually what our participants defined as success; saving millions of dollars and preventing years of development toward a bad result.

36 out of 41 interview participants described failure as a routine part of an iterative development process; a necessary part of learning how to build the right thing and solve the right problem. Failure in this sense is expected and accounted for in the learning process. When product delivery is broken up into small, iterative pieces, failure is easier to recover from than traditional multi-year, multi-million dollar efforts.

Likewise, our participants considered traditional success, like launching on time and under budget, as a failure if the end product doesn't solve the core problems being addressed. Because it can be so difficult to deliver anything at all, government culture often celebrates the delivery of any product as success, regardless of whether it met public needs or was sustainable.

Long delivery timelines

> *"Government will have like 10 year modernization plans for certain things and by the time that plan is complete, the technology has totally changed."*

Failure is more likely in large, multi-year and multi-million dollar technical efforts. Some projects stretch on seemingly endlessly. These types of failures waste tremendous amounts of money, time, and resources and can result in the abandonment of unfinished or partially completed work. Development then starts over and the cycle repeats. Even when projects launch successfully, the time that passed between identifying the need to delivering the service was often so long that the solution is outdated and insufficient. This explains why many public services look and feel 5-10 years behind the consumer technology that people see in their everyday lives.

Government projects are often multi-year efforts with many moving parts. Traditional waterfall delivery methods collect specific requirements up front and plot them on a delivery timeline. They focus on building every requirement and then delivering a completed product all at once. Using waterfall development processes, the government attempts to change large technical systems, update business processes, and digitize billions of data records. Because of the scale of many government projects, this approach to development can take years or even decades to complete. Over that time frame, many things change, including priorities, budgets, and policies. If a technology product is delivered, the product is likely to be out of date or to no longer reflect the needs of the public.

Leadership, cultural, and political changes are more likely to derail projects on long timelines. Many projects last longer than the tenure of the people in charge of the program. Most large government development projects also last longer than a single presidential term, meaning that political priorities can also shift before development is completed. Contracting delays or modifications can increase that timespan or risk starting the work over with a new vendor with a completely different approach.

Solving the wrong problem

> *"I think they fail in two discreet ways: One, they either fail to build the thing right, or two, they fail to build the right thing."*

One of the most common ways projects fail is that they don't solve the problems people have in a way that is intuitive and understandable to them. Participants spoke about the need to deeply understand what problems their programs were trying to address and craft a solution around those needs.

Many government technical initiatives are not driven by service delivery or program needs. Instead, these projects are focused on business needs. This type of failure solves the problems of the government at the expense of the public.

Participants spoke extensively about how technology projects may meet the traditional on time, on budget, on specification criteria for success but still fail to solve the real challenges people face. Projects can meet the letter of the law or policy but fail to consider who will use the service or how they will interact with it.

Services that are hard to use

> *"There's different categories of failure, a system being down, having downtime, not being reliable is a very different category of failure than it being really hard to use a system, not being able to be changed for something like an emergency versus being really expensive for normal recurring changes."*

Many leaders defined failure as launching a product no one uses. Poor adoption can be exacerbated by systems that are unreliable, slow, or error-prone. If the technology tool or system in question is hard to use, people will find other methods of accomplishing tasks, adopting work-arounds and process hacks.

Hard-to-use products can stem from a lack of human-centered research or assumptions derived from talking to people who will be delivering or receiving the service only once in a project's development cycle. Building technology with little involvement from the people who will be using the product or involvement only in an early "requirements gathering" phase causes products to quickly lose touch with the day-to-day operations they are meant to support.

Alternatively, a service can be built around the internal needs of an agency without considering the people using it. For example, the use of opaque legal or statutory language in application forms may protect the government from some liability, but may also prevent applicants from being able to accurately fill out the forms. This can result in unexpected burden on other parts of the service, like increased phone calls to help lines. Interviewees universally championed services that work for both internal government workers delivering the program or service and people who will be using that program or service, considering a wide range of abilities.

> *"The software didn't work how it was supposed to work and we went live with it."*

Unreliable technical tools and systems were also considered failures by our participants. Poor performance can mean anything from the system being unstable and crashing, to slow load times, limited scalability, changes or updates that break the system, and other issues. Security issues, technical errors, and other performance issues undermine trust in government systems and increase risk for those who use them.

Poor system performance ultimately shifts work outside of the system. Core tasks move to paper processes and manual tracking methods, dispersing data sources. Time consuming work-arounds and increased burden on people involved in the process lead to data integrity problems. This creates a lack of transparency as more work is done outside of the system of record. Such inconsistent use also creates programmatic problems like unequal and variable benefit delivery. It can provide inaccurate data to leaders which can have detrimental effects on decision making.

Rigid & unchangeable systems

> *"If a system is really hard to change in reaction to a big shock, that's more because it doesn't have a baseline level of resilience and there's not enough baseline change happening. So that's a source of that failure mode."*

Projects fail when technology can't be easily changed or updated to meet new system demands or adapt to changes in service delivery. Inflexibility can come from political, policy, legal, or software development decisions or may derive from existing aging software or development environments. Programmatic change may also be limited by vendor contracts, making updates prohibitively expensive or impossible without time-consuming contract modifications.

Products can be locked into a system that is not flexible or adaptable due to the rigidness or the age of its technology stack. Many case management systems in government still use older programming languages like COBOL and MUMPS or assembly languages, which make them difficult or time-consuming to update or integrate with newer technologies. A lack of developers with expertise in these older languages poses a real and significant risk to many government programs.

Regardless of the cause, systems and products that cannot change grow brittle over time, leading to environments that limit service delivery rather than supporting it. The longer systems go between changes, the harder they are to improve. This is one reason the government still has numerous legacy systems in operation: Older systems become too hard to replace but remain too important to fail.

Failure we haven't noticed yet

Many government programs or systems that are in production today, even some used by the public on a daily basis, have already failed but the government has not taken notice. This can take several forms.

Systems in use may have made sensitive data available on the open internet but the breach hasn't yet been detected. Systems in development may have flawed requirements that will not be revealed until the system is placed into production years later. Policy, however well-intentioned, may have a flawed premise or be overly prescriptive in a manner that will inevitably make success impossible.

> *"You can't just throw money [at it] and put add-ons onto something. That's not how it works. I worry that some people still think of it that way. I think the zero trust architecture is a much better way to go and I'm glad that that's picking up steam in government. But I'm sure there's also the next thing that we're not aware of and [not] ready for."*

Other examples of unknown failures include legacy features that supported a program at the time the service was launched that become bottlenecks that slow or break the service years later. Reliance on older technologies or requirements, such as fax machines or wet ink signatures, can ultimately undermine the services they are meant to support.

This category of failure is troublesome because, when detected, it is often unexpected and few contingencies are in place. If a system fails to launch after five years of development, there is little recourse but to fall back on legacy systems. When legacy systems fail under increased demand, as happened to many unemployment systems during the pandemic, there is often no backup option and the government effectively stops delivering critical services.

This type of failure can happen due to a lack of proper procedures that continuously take stock of programs and improve systems over time. Other times it is the result of willful blindness, such as when programs avoid the use of processes like disclosure vulnerability programs because they do not want to know if a system is vulnerable. Knowing about these types of issues can come with the uncomfortable responsibility to fix them.

What causes government projects to fail?

Knowing what failure looks like is only the first step toward improving outcomes. Research participants also discussed the causes of failure. Technical projects can be delivered on time and under budget but still fail. Government technical projects also fail (or are perceived to fail) often. Working backward from their definitions of failure, we noted behaviors or environments that were more likely to result in failure.

Waterfall project management

Programs developed using waterfall project management approaches depend on a single, big bang launch after years of development. If the product fails at launch, the failure is final or the recovery is lengthy and complex. The long development period, during which few enhancements are made to the legacy systems, means that the government must fall back on even older technology with little to show for years of development. Long timelines set the government back even farther, making modernization harder with each subsequent failed attempt.

> *"We are so disjointed and so far behind on our IT infrastructure and our ability to adjust to what is required to deliver modern programs or, you know, to deliver programs in a modern way. And [the] government we're so far behind because of a lot of legacy investments and a lot of, frankly, poorly written contracts from 10, 20 years ago with large scale software providers that own our infrastructure and a lot of our data.... It's very hard and unfortunately it's really dehumanizing in a lot of ways because you get a lot of people who [it's] their job to enable this 30 year old clunker of a piece of software and they get attacked about failures all the time that are not their own responsibility."*

Waterfall timelines and delivery practices mean that often no one sees the project until it is delivered. The development, decision making, and prioritization of tasks are completed separate from the program delivery. Generally this results in the release of a product that does not meet expectations and/or is mis-aligned with actual service delivery needs.

Unfortunately, the Federal government encourages and incentivises waterfall project management. Many senior positions require a project management certificate (e.g. Project Management Professional) that teaches the tenets of waterfall development. Budget planning encourages multiyear contracts with fixed requirements and payouts. The most common forms of contracting create specific requirements many years in advance of product launches. The use of alternative and more effective methods, such as iterative development or agile contracting often invite scrutiny and force leaders to accept greater personal and professional liability.

Traditional procurement & contracting methods

> *"The procurement process was just misery."*

Universally, interview participants said that traditional government procurement and contracting approaches hindered projects and held back success. Government contracting rules focus predominantly on physical products, providing the goods, supplies, and construction work that make up the bulk of government purchases. Software products and public service delivery are relatively new additions to government purchasing needs. The Federal Acquisitions Regulations are particularly poorly suited to software development.

Federal contracts are managed by contracting officers, an official role that requires a professional certificate (a warrant) and strict adherence to regulation. Contracting officers have specialized knowledge in contracting but they rarely have technical or service delivery expertise. Some contracting officers have technical teams or review boards supporting their technology contracts, but that is not the norm.

Any hitch in the contracting process can delay or even stop an initiative before it is started. Procurements often go through several rounds of clarifications and updates, adding months to vendor selection. Contract protests can delay projects for years or even cause the effort to be completely abandoned.

> *"People will sometimes say, well it's a failure because . . . it didn't launch on time or it didn't meet the need that we said was in the contract, but you know*

. . . how specified was that need in the contract? Maybe it met exactly what was in the contract and that's the point."

Participants said the level of specificity required in traditional contracting prevented choosing iterative development methods or the ability to adapt to changing program needs. Lengthy contract documents are full of specific features. Requirements bind agencies to those items, even when delivering to the letter of the contract doesn't make sense. This contracting model causes contracts to be executed without continued learning and often without user feedback. This creates an environment in which issues cannot be fixed or updated even when they are known.

Poor vendor management

Once awarded, contract delivery becomes another risk. The intense focus on contract management can itself distract from project performance. Years of incomplete or unsuccessful technical projects have a pattern of tightly-scoped, inflexible contracts. This can create an environment of distrust between the government and the contractors it relies on to deliver critical services, instead of fostering a collaborative, mutually-beneficial relationship.

Vendor oversight is often done through a combination of contracting officers and contracting officer's representatives. Contracting officers are rarely connected closely with service needs and have little insight into how those processes currently work. Contracting officer's representatives traditionally have subject matter expertise in a program or benefit but they often lack service delivery experience. Importantly, the people overseeing contracts rarely have experience with technology or design best practices and must rely on vendors to raise salient points or fall back on contract milestones to judge progress.

Part of the governance, we ask them how much is it gonna cost? How do you know how much it's gonna cost? You ask a vendor partner, right?

An example of this lack of technical expertise is when the government needs to develop a cost estimate. In many cases, the government asks technology vendors how much they should be paid. This dilemma, similar to progress updates, places the vendor fully in control of the project narrative instead of an objective civil servant.

Too often, contracts are written as rigid documents that lock vendors into a specific set of features or requirements that contracting officers and representatives do not understand. Progress is measured by a checklist of line items completed alongside budget and chronological milestones. Little regard is given to a service's utility.

Although cost and timelines are important to track, too few managers track whether the product actually works or if the people who need to use it can do so.

Likewise, the nature of waterfall contracting can lock vendors into producing products they know will not work or will not work well. Vendors have little recourse to push back against work that doesn't make sense, duplicative functionality, or other wasteful practices unless they have an informed contracting officer with whom to work.

Using the wrong measures for success

> *"I inherited a role that had no vision or mission. It had no meaningful scope of responsibility, it had no goals, it had no plan, it had no measures of success."*

Just as traditional contracting methods measure progress in ways that don't accurately reflect progress, government program metrics often measure the wrong things. Many leaders spoke about the importance of measuring success and, just as importantly, how and what is being measured. Leaders spoke of "vanity metrics" like thousands of records in a system, millions of queries, or certain percentages of technology products using modern technical stacks. None of these metrics identify how well those systems work (e.g. system uptime), how easy they are to use (e.g. task completion rates), or how many people are actively using them. Measurement is important, but measures need to track against meaningful program goals and outcomes.

"Lift & shift" projects

> *"All of the applications were there and worked but there was no streamlining or refinement, they lifted and shifted a whole Linux box with an outdated version of Postgres on it and just left it there."*

"Lift and shift" is a term that describes a project when the goal is to migrate the data, functions, and content of a technical product or system over to a new product or system with few or no changes. These types of projects are often celebrated as successes as they "modernize" the product by using a new hosting environment (e.g. moving software from a local server to cloud infrastructure) or programming language (e.g. rewriting 60 million lines of COBOL as 30 million lines of Java). This process does not update business processes or address existing usability issues. Lift and shift projects embody the false notion that new technology alone will lead to better service delivery.

Many of these types of projects seek to update very old technical systems, some of which were developed before digital processes, smartphones, and widespread use of the internet. Those systems served the needs of the program as it was in 1970, 1980, or

1990, but no longer reflect the way people interact with the government or receive a benefit. Simply updating the software platform or technical stack does little to resolve these issues.

Separating technology & program offices

> *"People were scared of technology. We would pull up a text editor or we would pull up code and people would be like, 'Why is your computer black... with things I don't understand? Like it's really scary, I don't wanna touch it.'... There was this idea that I'm not the expert in that: I have to fully defer and I don't want to understand because I'm not even capable of understanding. So if I need to defer to this other person, that dynamic is really unproductive for building good services."*

In most Federal government agencies, the information technology (IT) department is separate from the program office, creating a divide where the people with technical experience work separately from the people who use their products or administer public services. The people who are using the agency's services may have limited technical skills and will defer to the IT professionals to make technical decisions. This cultural divide is sometimes reinforced by forcing people to open tickets, make change requests through managers, or by the overuse of technical jargon.

Another troubling outcome of separating technical and program staff is the lack of business process improvement. When development occurs separate from programmatic teams, the technologies will reflect the existing communication patterns of the program office. This is a concept known as "Conway's law". The more decisions that need to be made before a task can be completed, the more complex the software will have to be to manage that task.

Successful development projects work directly with program offices to identify redundancies and streamline both business practices and software. This results in better functioning service with less expensive, more secure, easier to use software. Iterative, collaborative development leads to both better technology and better service delivery.

The oversight trap

Federal leaders commented that technical successes are rarely celebrated or spoken about. They largely go unnoticed. When successes are celebrated, they are usually told as stories of rescuing a failing project. Everyday success stories are rare.

While success stories are quiet affairs, government failures often make national or international headlines and bring the ire of the public, political leadership, and congress. Oversight bodies produce overwhelmingly critical reporting, rarely commenting on positive behavior. If 80% of a program is managed well, public reporting is likely to center on the 20% of potential improvements. This deluge of criticism, and a corresponding lack of praise, casts a negative light over all Federal programs and civil servants. Ironically, responding to the oversight and press inquiries that follow can quickly become a project's focus, distracting from the work of identifying the cause of the problem and remedying it.

> *"The level of oversight, regardless of its rigor and applicability to real world challenges that people inside agencies are dealing with on a day-to-day basis, creates a lot of public data that folks can point back to and say, this went wrong."*

The Federal government has many sources of oversight. Government oversight bodies like the Inspectors General and the General Accountability Office already exist, but additional specialized bodies like the Pandemic Response Accountability Committee are becoming increasingly common. The Office of Management and Budget and organizations like the National Institutes of Standards and Technology provide even more routine oversight and standards setting tasks. This is supplemented by Congressional oversight. Journalism and public watchdog groups are also routinely named as sources of oversight.

> *"You're not incentivized to have a big vision for technology in government."*

Harsh oversight also creates an unhelpful cycle of reactive measures that can increase the likelihood of failure. Fraud in one program, for example, can drive interest in more restrictive rules across all programs, even those with low incidences of fraud. Such reactive policies may resolve a single incidence of fraud while preventing thousands of other users from accessing their benefits or setting untenable requirements on other government programs.

> *"I think the core piece of bad cultures and low performing cultures is just hopelessness."*

This also creates a more insidious form of oversight: self-censorship. When there are few rewards for making progress and many downsides to attention of any kind, oversight incentivises inaction. Even when success is likely, a disproportionately small chance of failure may discourage individuals or entire agencies from attempting to improve public services at all. Innovation cannot thrive in a fear-based environment.

What increases the likelihood that a project will be successful?

Interview participants had many definitions of failure, often accompanied by specific stories with details about how and why things went wrong. Success is much more nuanced. No single definition emerged. Success generally included user research, product adoption, iterative development processes, and a focus on agency and program goals but, ultimately, is different with each program.

Notably, success is not technology dependent. Newer technologies like cloud storage or APIs were mentioned by several participants but were not necessarily an indicator of success unto themselves. As long as the core technology is able to scale, change, and communicate across systems, the platform or specific technology stack used was less important.

Across all interviews, traditional success in government was defined simply as “not failing.” If a program is delivered on time, under budget, and is released publicly, it is considered a success. A commitment to changing that narrative is a key attribute of successful technology leaders. Shifting the focus from launching a product or service to delivering outcomes for constituents is a defining trait of modern government leadership.

Many leaders also noted that not completing a project can sometimes be an achievement. Building the right solution for the right problem is considered a success just as much as *not* building solutions for the wrong problems. Sometimes stopping a technical project, even one that has cost many millions of dollars and has been underway for many years, is the right thing to do.

Well-defined problems

> *“You're defining and redefining your specifications and criteria from the get go or along the way. So how do you know you got there? That means that you need to have a really clear definition of what the problem is that you're trying to solve.”*

Government technology projects hyperfocus on the technical aspects of a service or tool, often identifying “outdated technology” or “modernization” as the stated problem. Likewise, they may define a problem as needing to incorporate changes in policy or legislation. Defining the problem in these ways leads to large efforts with unspecified outcomes and extensive requirements.

Better solutions are achieved by narrowly defining problems, setting clear goals, and centering product requirements around specific outcomes. This may include breaking a larger, more complex effort into smaller, more manageable issues to solve. Great leaders work with the people providing and receiving services to define problems and identify solutions instead of sourcing goals solely from the business or IT department, vendors, Congress, or political leadership.

> *"I think a lot of times when technology modernization goes poorly, it's specifically because the problem definition is not well specified. It's simply, well we want a new system because we're unhappy with the old system."*

Similarly, interview participants linked technology goals to program outcomes. Just as the technology's function is driven by the need to provide services to people, the technology's goals should align with the program's goals and objectives. Technology choices such as coding language, platform, and display method should be made after program goals are defined. All technology choices should support the agency or program's core services.

Iterative development practices

Over time, many private sector organizations have shifted away from waterfall delivery methods to more iterative approaches. Often referred to as "agile development," these approaches work in small increments with narrowly-defined goals and a frequent cadence for launching updates to software and services.

Across government there are multiple commercial frameworks and interpretations of "agile" practices, many of which are overly complex, proprietary, or not actually iterative. These frameworks often mimic the rituals and use the language of iterative delivery without actually changing the traditional waterfall process.

> *"They made a transition to Agile that I lived through and enabled me to see a different way of managing not just technology projects, but projects. And it was very well executed. And then I went to [another project]: There was a lot of, you know, fake agile, faux agile, and I saw what it's like when it's not well executed."*

Our leaders, especially those directly involved directly in the product delivery process, tended to use agile as a shorthand for the practice of continuous user research and iterative service delivery changes. All of them spoke about the need for iterative approaches to development. A development process that focuses on small changes in short periods of time informed by real-world feedback is more important than any brand name framework.

Successful leaders encourage their teams to take chances within reasonable boundaries. This allows for the upsides of experimentation while reducing the downward boundaries of failure. It also allows frequent opportunities to celebrate interim successes, which helps drive momentum and increase team morale.

Iterative teams also maintain and regularly review a backlog; the list of work they still needed to complete. This helps prioritize what is most important, remove features that are no longer needed, and add new work based on how people are actively using the product, rather than developing only to program milestones.

> *"Making sure you're working with the users, making sure you're shipping and then iterating and pivoting and course correcting."*

This type of approach has many benefits. Usability and technical problems are spotted earlier in the development process because working products are released on a regular basis and checked for errors. Instead of waiting months or years for changes, new features or functionality are delivered and people provide rapid feedback on how well things work. Products are developed continuously as long as the product is being used and never placed in "operation and maintenance" mode. Instead of being replaced every few decades, systems evolve over time to meet current user needs.

Iterative development is common practice in many private sector companies but it is especially well-suited to government. Consistent user feedback is in line with the tenets of democracy. It is also a practical solution to the lack of notable success in government. When a sprint has a poor outcome, that failure is contained and provides feedback that can be used to avoid similar mistakes in the future. Consistent success and contained failures mitigate many of the real and cultural fears that are pervasive in government offices.

Likewise, iterative development happens on a time-frame that mitigates larger problems of leadership or political turnover. Features delivered before a project lead moved on to another job or political administration turnover become the status quo. They are much less likely to be disregarded or dismissed. If new leadership wants to shift the focus of a program, they can do so using the same iterative process, gradually shifting development in a way that is less disruptive for users or civil servants. Finally, projects that are based on what people actually need are less skewed by politics or personalities. Good user research practices create a record that leaders can point at to justify why decisions were made and help define success based on real-world goals.

Meaningful success metrics

> *“I make sure that our team is not focused on those vanity metrics but actually we sit down and talk about, ‘What does success mean to our team? What are we trying to accomplish...?’ And then make sure that those are the things that we're focused on.”*

Successful leaders tie usability and program goals to measurable metrics which provide insight into how well the product or service is performing. They measure success by tracking people-centered metrics like adoption, successful applications, fewer account lockouts, faster application approval processes, or fewer duplicate records. Many participants spoke about how traditional projects could meet standard on time, on budget, on spec metrics and yet fail to meet user needs.

> *“So you may technically deliver something that was right according to the requirements, but it's no longer relevant. And so that's a massive failure that I call a failed success. I mean, you delivered according to the tenants of what you agreed on, but there's no adoption, there's no utilization of stuff because of its [ir]relevance.”*

Requirements, costs, and timelines are the traditional success measures because they are concrete, fixed elements that are easy to track. Unfortunately these metrics can also be arbitrary and disconnected from program and service delivery goals. How successful would a technical product be if it was delivered on time, but “on time” was five years down the road when program needs have changed? How successful is a technical tool with all 500 listed business requirements if only 10 of those requirements are actually useful?

> *“I think when people stop having experiences when they're like, oh my God, I'm shocked that this government service actually worked. When it's the assumption that it works properly and is a good experience and people aren't surprised that it's a good experience. That is kind of a silly metric but, I think, a real one.”*

Cost and time are important metrics, but should not be the only measures of success. More insightful metrics based on the program goals and user research drive useful improvements and greater adoption.

Purposeful data

> *“The data team allowed me to see the American people.”*

Participants used data to help them understand progress and validate decisions. Many stressed the need to select data that gave an honest view of success metrics. Others cautioned that data could be misused to back up false narratives. Done incorrectly, data collection can quickly become a goal in-and-of-itself.

> *“...the biggest mistake anybody can make is trying to make the data fit their perceptions and their assumptions.”*

When paired with clear success metrics, data is a key indicator of whether programs were delivering the correct outcomes for the right people. While manual data collection is necessary at times, especially early on in a project, key data points are automated whenever possible. Successful leaders communicate data as broadly as possible. Data transparency drives team consensus and makes it easier to focus on results.

People-centered

> *“I try to always orient our goals and our work around what's best for users and for the public because that's the point. It's easy to get caught up in the swirl of political priorities or in the weeds of this one agency or these agencies have these categories of problems. But at the end of the day, what we need to be doing, what really counts as innovation, is delivering results for users.”*

Successful projects are built with the people who will use the product, including both the people who deliver (civil servants) and receive the services (the public). In successful projects, people interact with products throughout the development and delivery cycle, not just at the beginning or the end. They continually experience how the product functions and help inform what would be built next. This feedback loop allows teams to focus on functionality that solves real, well-defined problems.

Avoiding distraction

> *“It was like every day five people came to me with another complicated thing that they really needed this program to do and I had the air cover I needed to tell them ‘no’ and know that my boss was going to tell them all [the same].”*

Successful leaders provide their teams “air cover.” Air cover was defined as an environment where leadership supported and protected the goals of the people they oversaw. Leaders empower their teams to say no to superfluous requests even, or especially, when they come from higher authorities. They also remove barriers that were set in the way of progress. This type of support was especially important when a team was trying a new method or delivering a sensitive initiative. Participants all said their job was to make sure their teams could do work without political pressures, interagency disagreement, or other distractions. Successful leaders absorb the risks associated with trying something new, build trust with stakeholders, and allow teams to focus on delivering against program goals.

> *"I always felt that as a leader, no matter where I was, that I had to provide the cover. I had to really ensure that people believed that these things could be done and that when they ran into a roadblock it was my job to fix it."*

Leaders also commented that they needed air cover themselves to do their jobs. Air cover comes in many forms depending on the leadership level, such as support from political appointees or a presidential administration, or the reinforcement of their priorities in the press and through public talking points. Protection from these distractions, particularly from issues that can spiral into reactive work, led to greater chances of success.

Great teams

> *"I have a firm belief . . . that a huge part of being a successful leader is solved by really, really great people that are thoughtfully matched against carefully crafted problem sets. And that if you get those things right, then it makes almost everything else much, much easier to be successful."*

Successful technology projects need teams of talented, forward-thinking people to do the work. Leaders stated that real success came when technical teams were carefully matched with a set of solvable problems, aligning their strengths and skill sets to what the technical product and agency mission needed.

This includes providing a large enough team to sustain an iterative product. Instead of having one or two IT specialists maintaining a technical product long-term, successful teams are multidisciplinary and embedded into the business unit or program office, with direct access to the people using the product.

Team members are selected for their individual abilities. Participants eschewed the notion that program managers were interchangeable, a common government misconception. Participants also considered lawyers, policy analysts, contracting professionals, and other positions outside of the program or IT office part of their team. If a single individual has the ability to veto critical decisions, such as a General Counsel, they should be included in team communications as early as possible.

> *"We could do all the training in the world and get contracting people comfortable with new approaches to contracting for agile and then some*

random middle manager somewhere will say "no." And, well, you know, we're starting over.

Leaders overwhelmingly emphasized the need for more user research, design, and contracting support across all agencies and all projects.

Modern tools

"A more serious constraint is actually, at least in this and maybe other agencies as well, we can't use all of these software and tools in the workplace that are actually relevant to us understanding how to do our jobs better.

Too often the government overlooks the many benefits of modern tools; the technologies used to build government systems and services. What can sometimes be seen as nice-to-have software can have significant effects on success and failure. Perceived costs and security concerns related to modern tools often lead to increased effort, with commensurate cost, and a reliance on less secure development practices.

Many leaders commented on how difficult it can be to understand what technology options are available due to the siloed and outdated nature of many Federal government systems and IT policies.

People within the government often can't access the latest technologies. Even when they can, useful features are sometimes disabled or unavailable. For example, participants noted that they had collaborative tools but were not allowed to use them to collaborate with anyone outside of their office, including other agencies and government contractors. The lack of tools or critical features was frequently tracked back to the way the Government manages technical risk.

"When Office 365 rolled out they turned off all of those [collaboration] features and it's like, what is the point of having Office365?"

Modern tools encourage open communication (e.g. Slack, Github), consistency and security (e.g. continuous delivery, monitoring), and collaboration (e.g. Google workspace, Microsoft 365). Others are used to track more useful and finite metrics, such as completion rates and usability feedback. These types of tools incentivise the positive behaviors listed in this report, creating a more efficient and effective work environment and increased user feedback. Where they exist, modern tools help to attract and retain talent, create a transparent record, foster employee growth, reduce effort (and therefore burnout), and increase consistency.

Agile contracting & budgeting

The details of agile contracting and budgeting are complex. While not the primary focus of this research, it is important to note that leaders who used Federal contracting to fit project needs were more successful than their counterparts who used or inherited traditional government contracts. Notably, agile contracting works well with both the traditional government success metrics (on budget, on time) and the program-specific metrics advocated for in this report.

The Federal government manages the vast majority of technical development and service delivery through private-sector contracts. Just as human-centered design, iterative development, and open communication are critical for leadership and civil servants, they are critical for government contractors. Even if leaders create productive, successful cultures within the government they cannot be successful without creating the same opportunities for their private sector partners.

> *"There's just this sentiment that all of our projects need to be $75 million and be road-mapped out seven years from now. I'd love for a little bit more iteration in life."*

There is significant flexibility to include iterative development within the Federal Acquisitions Regulations, but agencies rarely use it. Programs like the TechFAR Hub in the Office of Management and Budget, the General Services Administration's Acquisition Stack, and the Department of Homeland Security's Procurement Lab have proven the value of so-called "agile contracting" but they remain a small part of overall Federal contracting.

> *"There are these perverse incentives in government where, if you show progress, you need to ask for as much money as possible when someone is paying attention to have a sustainable budget long-term. And this, the way that budgeting works, does not incentivize iterative progress."*

For fiscal year 2022, the White House proposed a Federal IT budget of $109.4 billion. The vast majority of this is spent on maintaining legacy systems with only a small fraction dedicated to iterative development projects. When contractors implement human-centered design, iterative development, and openly communicate with Federal employees that have technical knowledge, it is possible to increase success rates, save billions of dollars, and reduce the burden on both civil servants and the public.

Similarly, iterative development is limited by traditional budget structures which are often planned years in advance. While there have been advances in budget structuring, they remain largely in the pilot phase, even decades on. The Information Technology

Oversight and Reform budget at the Office of Management and Budget, the Federal Citizen Services Fund and the Technology Modernization Fund at the General Services Administration, and revolving funds developed under the Modernizing Government Technology Act provide multi-year, flexible budgeting options. These flexible sources of funding have fostered positive changes in the way contracts are written, technology products are developed, and government programs are administered. Even so, they have not yet become standard practice and it is likely that significant legislative, regulatory, and cultural changes need to be made in order to reform the way we fund service delivery in government.

Conclusion

The overwhelming amount of oversight over government programs, the voluminous documentation of failure, and the absence of positive recognition when the government successfully delivers public value has discouraged reasonable risk taking and created a fraught environment for Federal leadership.

At the same time, criticisms of past performance have merit. Adherence to outdated project management practices, tools, and mindsets have led to significant failures. Some outdated practices have been encoded into the business process of government, such as annualized budgets, strict acquisition regulations, and waterfall project management. Others derive from hierarchical policy or reactive legislation. Some failures are simply a product of time passing or priorities shifting.

While difficult to navigate, there is sufficient latitude in existing policy, regulation, and process to successfully deliver government programs. Many onerous ‘requirements’ are actually cultural tropes embedded in the way things have always been done. Understanding the difference between cultural expectations and legal requirements is key. Successful leaders change the culture of their organizations to one that takes advantage of the full breadth of opportunity provided in law and policy in order to deliver better outcomes for the public.

Leaders who use structured, iterative development methods achieve greater success. They also see increased morale. Using flexible budgeting and agile contracting methodologies aligned high-functioning teams inside the government with capable private sector providers, increasing rates of success. Teams with regular access to people delivering and using a service understood first-hand how technology supported their mission. Changes driven by service delivery needs and tracked using program-specific metrics were critical to successful product delivery.

Ultimately, neither success nor failure should be seen as absolute. Successful leaders see public service delivery as a complex network of incentives. Rather than see success or failure as good or bad, they see their programs as getting either better or worse. They focus on the essential parts of service delivery, such as hiring, contracting, product management, and communication, where they can more clearly see and address clear patterns of what is working well or going poorly. Successful leaders seek to maximize improvements and encourage experimentation and risk taking by setting well-defined goals, providing air cover, and containing failure to short increments rather than delivering “modernization” as a monolithic goal.

Acknowledgements

Thank you to the past and present staff members from the Executive Office of the President, the Office of the Chief Administrative Officer of the House of Representatives, the Government Accountability Office, the Department of Homeland Security, the General Services Administration, the Department of Veterans Affairs, the Department of Agriculture, the Department of Health and Human Services, the Department of Labor, The Department of Commerce, The National Aeronautics and Space Administration, and the many others who supported this research with their time and input.

Thank you to our early readers for their help in making this report comprehensive, readable, and accurate.

Thank you to the Partnership for Public Service for their research support and background assistance, to the U. S. Digital Response for our leadership conversations and Nava PBC for sharing their perspectives on this topic. Thank you to the Tech Talent Project, U.S. Digital Service, Coding it Forward, the Digital Corps, and the Beeck Center for their work in this space.

And thank you to Jen Pahlka whose encouragement supported this project. You can read more about technology in government in her new book here.

This work was funded by a philanthropic grant.

Appendix A: Reports & references

- HAZE report
- USCIS Has Been Unsuccessful in Automating Naturalization Benefits Delivery
- Review of the Bureau of Consular Affairs' ConsularOne Modernization Program–Significant Deployment Delays Continue
- Air Force cancels Air Operations Center 10.2 contract, starts new pathfinder effort
- GAO Faults DOD for Lax Oversight of F-35 Spare Parts
- Agencies Need to Develop Modernization Plans for Critical Legacy Systems
- Lessons learned from the government's biggest attempt to fix tech procurement
- Oversight.gov
- How Complex Systems Fail
- The Agile Manifesto
- Agile Method Criticism
- Conway's Law
- De-Risking Guide
- TechFAR Hub
- Develop a data-driven culture and workforce