Tim O'Reilly

Government as a Platform

During the past 15 years, the World Wide Web has created remarkable new methods for harnessing the creativity of people in groups, and in the process has created powerful business models that are reshaping our economy. As the Web has undermined old media and software companies, it has demonstrated the enormous power of a new approach, often referred to as Web 2.0. In a nutshell: the secret to the success of bellwethers like Google, Amazon, eBay, Craigslist, Wikipedia, Facebook, and Twitter is that each of these sites, in its own way, has learned to harness the power of its users to add value to—no, more than that, to co-create—its offerings.

Now, a new generation has come of age with the Web, and it is committed to using its lessons of creativity and collaboration to address challenges facing our country and the world. Meanwhile, with the proliferation of issues and not enough resources to address them all, many government leaders recognize the opportunities Web 2.0 technologies provide not just to help them get elected, but to help them do a better job. By analogy, many are calling this movement *Government 2.0*.

What the heck does that mean?

Tim O'Reilly is the founder and CEO of O'Reilly Media, Inc., thought by many to be the best computer book publisher in the world. In addition to Foo Camps ("Friends of O'Reilly" Camps, which gave rise to the "un-conference" movement), O'Reilly Media also hosts conferences on technology topics, including the Web 2.0 Summit, the Web 2.0 Expo, the O'Reilly Open Source Convention, the Gov 2.0 Summit, and the Gov 2.0 Expo. Tim's blog, the O'Reilly Radar, "watches the alpha geeks" to determine emerging technology trends, and serves as a platform for advocacy about issues of importance to the technical community. Tim's long-term vision for his company is to change the world by spreading the knowledge of innovators. In addition to O'Reilly Media, Tim is a founder of Safari Books Online, a pioneering subscription service for accessing books online, and O'Reilly AlphaTech Ventures, an early-stage venture firm.

This essay first appeared as chapter 1 in Daniel Lathrop, Laurel Ruma, eds. (2010), Open Government: Collaboration, Transparency, and Participation in Practice," *O'Reilly Media. We have published it in* Innovations*, with permission from the author, under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 United States license.*

Much like its predecessor, Web 2.0, "Government 2.0" is a chameleon, a white rabbit term, that seems to be used by people to mean whatever they want it to mean. For some, it is the use of social media by government agencies. For others, it is government transparency, especially as aided by government-provided data APIs. Still others think of it as the adoption of cloud computing, wikis, crowdsourcing, mobile applications, mashups, developer contests, or all of the other epiphenomena of Web 2.0 as applied to the job of government.

All of these ideas seem important, but none of them seem to get to the heart of the matter.

Web 2.0 was not a new version of the World Wide Web; it was a renaissance after the dark ages of the dotcom bust, a rediscovery of the power hidden in the original design of the World Wide Web. Similarly, Government 2.0 is not a new kind of government; it is government stripped down to its core, rediscovered and reimagined as if for the first time.

And in that reimagining, this is the idea that becomes clear: government is, at bottom, a mechanism for collective action. We band together, make laws, pay taxes, and build the institutions of government to manage problems that are too large for us individually and whose solution is in our common interest.

Government 2.0, then, is the use of technology—especially the collaborative technologies at the heart of Web 2.0—to better solve collective problems at a city, state, national, and international level.

The hope is that Internet technologies will allow us to rebuild the kind of participatory government envisioned by our nation's founders, in which, as Thomas Jefferson wrote in a letter to Joseph Cabell, "every man…feels that he is a participator in the government of affairs, not merely at an election one day in the year, but every day."[1]

As President Obama explained the idea during his campaign: "We must use all available technologies and methods to open up the federal government, creating a new level of transparency to change the way business is conducted in Washington, and giving Americans the chance to participate in government deliberations and decision making in ways that were not possible only a few years ago."

Allowing citizens to see and share in the deliberations of government and creating a "new level of transparency" are remarkable and ambitious goals, and would indeed "change the way business is conducted in Washington." Yet these goals do not go far enough.

LESSONS LEARNED FROM THE SUCCESS OF COMPUTER PLATFORMS

There is a new compact on the horizon: information produced by and on behalf of citizens is the lifeblood of the economy and the nation; government has a responsibility to treat that information as a national asset. Citizens are connected like never before and have the skill sets and passion to solve problems affecting them locally as well as nationally. Government information and services can be provided to citizens where and when they need them. Citizens are empowered to spark

the innovation that will result in an improved approach to governance. In this model, government is a convener and an enabler rather than the first mover of civic action.

This is a radical departure from the existing model of government, which Donald Kettl so aptly named "vending machine government."[2] We pay our taxes, we expect services. And when we don't get what we expect, our "participation" is limited to protest—essentially, shaking the vending machine. Collective action has been watered down to collective complaint. (Kettl used the vending machine analogy in a very different way, to distinguish between the routine operation of government and the solution of new and extraordinary problems, but I owe him credit for the image nonetheless.)

What if, instead of a vending machine, we thought of government as the manager of a marketplace? In *The Cathedral & the Bazaar*, Eric Raymond uses the image of a bazaar to contrast the collaborative development model of open source software with traditional software development, but the analogy is equally applicable to government.[3] In the vending machine model, the full menu of available services is determined beforehand. A small number of vendors have the ability to get their products into the machine, and as a result, the choices are limited, and the prices are high. A bazaar, by contrast, is a place where the community itself exchanges goods and services.

But not all bazaars are created equal. Some are sorry affairs, with not much more choice than the vending machine, while others are vibrant marketplaces in which many merchants compete to provide the same goods and services, bringing an abundance of choice as well as lower prices.

In the technology world, the equivalent of a thriving bazaar is a successful platform. If you look at the history of the computer industry, the innovations that define each era are frameworks that enabled a whole ecosystem of participation from companies large and small. The personal computer was such a platform. So was the World Wide Web. This same platform dynamic is playing out right now in the recent success of the Apple iPhone. Where other phones have had a limited menu of applications developed by the phone vendor and a few carefully chosen partners, Apple built a framework that allowed virtually anyone to build applications for the phone, leading to an explosion of creativity, with more than 100,000 applications appearing for the phone in little more than 18 months, and more than 3,000 new ones now appearing every week.[4]

This is the right way to frame the question of Government 2.0. How does government become an open platform that allows people inside and outside government to innovate? How do you design a system in which all of the outcomes aren't specified beforehand, but instead evolve through interactions between government and its citizens, as a service provider enabling its user community?

This chapter focuses primarily on the application of platform thinking to government technology projects. But it is worth noting that the idea of government as a platform applies to every aspect of the government's role in society. For example, the Federal-Aid Highway Act of 1956, which committed the United States to build-

ing an interstate highway system, was a triumph of platform thinking, a key investment in facilities that had a huge economic and social multiplier effect. Though government builds the network of roads that tie our cities together, it does not operate the factories, farms, and businesses that use that network: that opportunity is afforded to "we the people." Government does set policies for the use of those roads, regulating interstate commerce, levying gasoline taxes and fees on heavy vehicles that damage the roads, setting and policing speed limits, specifying criteria for the safety of bridges, tunnels, and even vehicles that travel on the roads, and performing many other responsibilities appropriate to a "platform provider."

While it has become common to ridicule the 1990s description of the Internet as the "information superhighway," the analogy is actually quite apt. Like the Internet, the road system is a "network of networks," in which national, state, local, and private roads all interconnect, for the most part without restrictive fees. We have the same rules of the road everywhere in the country, yet anyone, down to a local landowner adding a driveway to an unimproved lot, can connect to the nation's system of roads.

The launch of weather, communications, and positioning satellites is a similar exercise of platform strategy. When you use a car navigation system to guide you to your destination, you are using an application built on the government platform, extended and enriched by massive private sector investment. When you check the weather—on TV or on the Internet—you are using applications built using the National Weather Service (or equivalent services in other countries) as a platform. Until recently, the private sector had neither the resources nor the incentives to create space-based infrastructure. Government as a platform provider created capabilities that enrich the possibilities for subsequent private sector investment.

There are other areas where the appropriate role of the platform provider and the marketplace of application providers is less clear. Health care is a contentious example. Should the government be providing health care or leaving it to the private sector? The answer is in the outcomes. If the private sector is doing a good job of providing necessary services that lead to the overall increase in the vitality of the country, government should stay out. But just as the interstate highway system increased the vitality of our transportation infrastructure, it is certainly possible that greater government involvement in health care could do the same. But if the lesson is correctly learned, it should do so not by competing with the private sector to deliver health services, but by investing in infrastructure (and "rules of the road") that will lead to a more robust private sector ecosystem.

At the same time, platforms always require choices, and those choices must be periodically revisited. Platforms lose their power when they fail to adapt. The U.S. investment in the highway system helped to vitiate our railroads, shaping a society of automobiles and suburbs. Today, we need to rethink the culture of sprawl and fossil fuel use that platform choice encouraged. A platform that once seemed so generative of positive outcomes can become a dead weight over time.

Police, fire services, garbage collection: these are fundamental platform services, just like analogous services in computer operating systems. And of course, here we have an "antipattern" from technology platforms: the failure to provide security, for example, as a fundamental system service, leaving it instead to the "private sector" of application vendors, has imposed a huge downstream cost on the technology ecosystem.

The question of Government 2.0, then, is this: if government is a platform, how can we use technology to make it into a better platform?

This question allows us to fruitfully extend the platform metaphor and ask: what lessons can government take from the success of computer platforms, as it tries to harness the power of technology to remake government?

LESSON 1: OPEN STANDARDS SPARK INNOVATION AND GROWTH

Time and again, the platforms that are the most generative of new economic activity are those that are the most open. The modern era in computing began in 1981 when IBM published the specifications for a personal computer that anyone could build using off-the-shelf parts. Prior to the introduction of the PC, IBM had a stranglehold on the computer market. It was a valuable but limited market, with very few vendors serving a small number of very big customers.

After the introduction of the PC, barriers to market entry were so low that Michael Dell, a Texas college student, was able to start what became a multibillion dollar company out of his dorm room. The market for personal computers exploded. IBM had estimated a total of 245,000 PCs would be sold over five years; as we now know, the eventual market size was in the billions, as scrappy little companies like Microsoft worked to put "a computer on every desk and in every home."[5]

At the same time, the standardization of the personal computer led to unexpected consequences: software became a higher-margin business than hardware; industry power shifted from IBM to Microsoft.

In its early years, Microsoft triumphed by establishing the best platform for independent software developers. Just as the standard architecture of the IBM PC lowered the barriers to marketplace entry by hardware manufacturers, the standardized APIs of MS-DOS and, later, Microsoft Windows made it easy for developers to "add value" to the personal computer.

Over time, Microsoft began to abuse their market power as the platform provider to give advantage to their own applications. At that point, the PC software marketplace became less and less vibrant, with most of the profits accruing to a few dominant companies. As a result, many people mistakenly take the lesson from the PC era that owning a platform is the secret of marketplace control and outsized profits.

In fact, by 1995, the PC era had run out of gas. The PC became less and less like a bazaar and more and more like a vending machine. We'd moved from the open personal computer as the platform to the closed and tightly controlled

Microsoft Windows as the platform. When one vendor controls the platform, innovation suffers.

What reinvigorated the industry was a new open platform: the Internet, and more specifically, the World Wide Web. Both were radically decentralized—a set of rules for programs to cooperate and communicate, with applications provided by anyone who had a good idea and the skills to write one. Once again, barriers to marketplace entry were low, with multibillion dollar companies created out of college dorm rooms, and tens of thousands of companies competing to provide previously unimaginable new services. The bazaar was back.

We see the same dynamic playing out today in the cell phone market. Cell phone providers have traditionally operated on the vending machine model. Apple changed the rules of the game with the iPhone developer platform. Suddenly, anyone could develop smartphone applications.

The smartphone platform story is perhaps the one most comforting to those inside government. Unlike the IBM PC or the Internet, the Apple iPhone is not a completely uncontrolled Wild West. Apple actively manages the platform to encourage innovation and choice while enforcing clear rules. Some observers believe that over time, the iPhone platform will not prove open enough, and will be superseded by other, more open platforms. But for the moment, Apple appears to be creating an effective balance between control and what Jonathan Zittrain calls *generativity*.[6]

There are two lessons for government in these stories. The first is the extraordinary power of open standards to foster innovation. When the barriers to entry to a market are low, entrepreneurs are free to invent the future. When barriers are high, innovation moves elsewhere. The second is that vibrant platforms become less generative over time, usually because the platform vendor has begun to compete with its developer ecosystem.

Some readers may take the lesson to be that government plays an important role in antitrust enforcement, keeping a level playing field. Facing the crises of the day, from banking to health care, we see a story in which entrenched players have grown large and have used their resulting power to remove choice from the marketplace, extracting outsized profits not by creating value but by cornering it.

There may be an "antitrust 2.0" alternative. Rather than simply limiting the size or power of an entrenched player, can government insistence on openness and interoperability be used to cause a "market reset," through which innovation can once again flourish? Antitrust actions against Microsoft were focused on existing business models, yet the real competition for Microsoft came not from other businesses selling software, but from an entirely new class of advertising-based business models that were invented in the initially noncommercial, wide-open spaces of the World Wide Web.

One of the most important ways that government can promote competition is not through after-the-fact antitrust enforcement but by encouraging more innovation. And as has been argued here, the best way to do that is with open standards. So, for example, faced with the race by major players to dominate the emerging

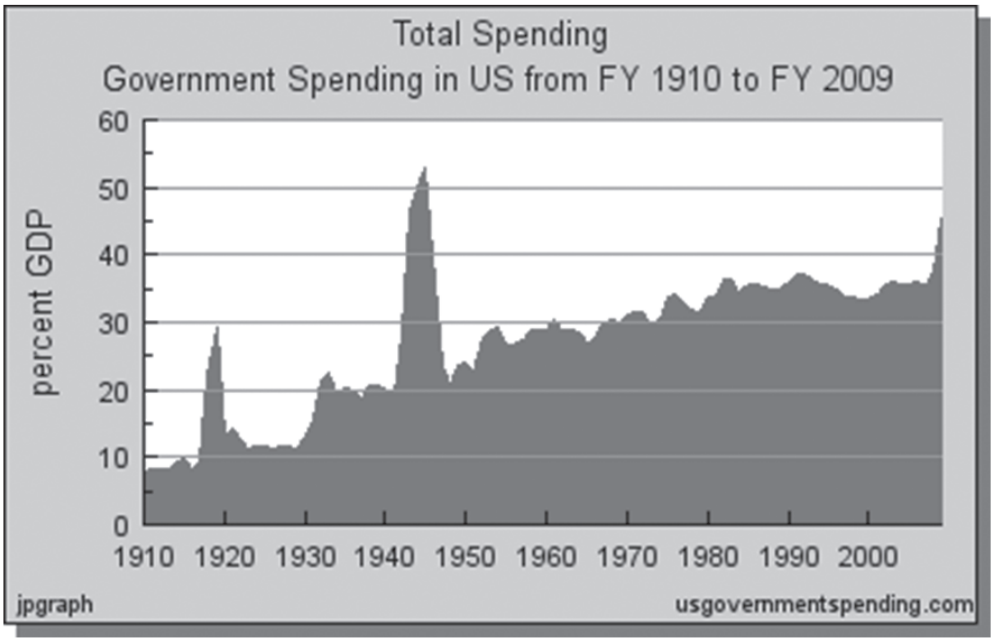

Figure 1. Government spending as percent of GDP since 1910.

Source: USGovernmentSpending.com

world of cloud computing, the government can forestall the risk of single-player dominance by throwing its weight behind open standards and interoperability in cloud computing. And in fact, this is just what we're seeing. The recent General Services Administration (GSA) Infrastructure as a Service (IaaS) solicitation devoted 5 of its 25 questions to vendors to the subject of interoperability:[7]

5. Please address the following Interoperability and Portability questions:

5.1. Describe your recommendations regarding "cloud-to-cloud" communication and ensuring interoperability of cloud solutions.

5.2. Describe your experience in weaving together multiple different cloud computing services offered by you, if any, or by other vendors.

5.3. As part of your service offering, describe the tools you support for integrating with other vendors in terms of monitoring and managing multiple cloud computing services.

5.4. Please explain application portability; i.e., exit strategy for applications running in your cloud, should it be necessary to vacate.

5.5. Describe how you prevent vendor lock in.

The recent U.S. Department of Defense guidance on the use of open source software by the military is a similar move that uses open standards to enhance competition.[8] The government's move to push for open patient records[9] also recognizes the power of open standards to promote innovation and bring down costs. And of

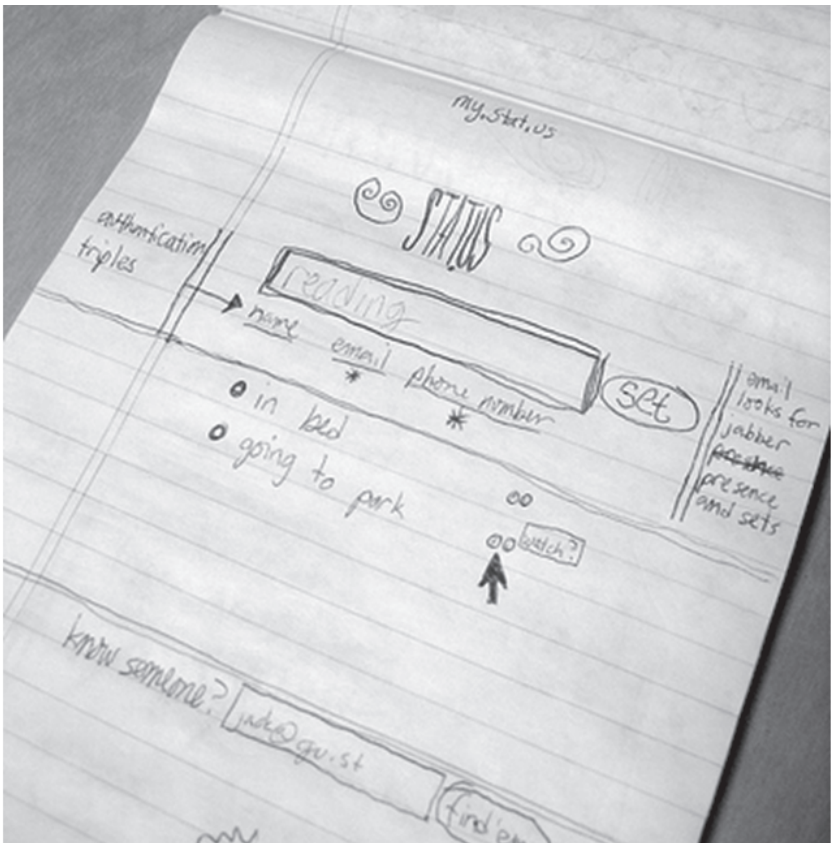

Figure 2. Jack Dorsey's original vision of Twitter.

course, the White House's Data.gov initiative, a portal for open APIs to government data, takes this idea to a new level.

In considering how open, generative systems eventually become closed over time, losing their innovative spark in the process, there is also a lesson for government itself. Figure 1, "Government spending as percent of GDP since 1910" shows the rising share of the U.S. gross domestic product consumed by all levels of government during the past 100 years.

As a platform provider, when does government stop being generative, and when does it start to compete with the private sector? When do its decisions raise barriers to marketplace entry rather than reduce them? What programs or functions that were used to bootstrap a new market are now getting in the way? There is no Justice Department that can bring an antitrust action against government; there is no Schumpeterian "creative destruction" [10] to bring unneeded government programs to an end. Government 2.0 will require deep thinking about how to end programs that no longer work, and how to use the platform power of the government not to extend government's reach, but instead, how to use it to better enable its citizenry and its economy.

LESSON 2: BUILD A SIMPLE SYSTEM AND LET IT EVOLVE

In one of the early classics of software engineering, *Systemantics*, John Gall wrote: "A complex system that works is invariably found to have evolved from a simple system that worked. The inverse proposition also appears to be true. A complex system designed from scratch never works and cannot be made to work. You have to start over beginning with a working simple system." [11]

Again, the Internet is a case in point. In the 1980s, an international standards committee got together to define the future of computer networking. The Open Systems Interconnect (OSI) model was comprehensive and complete, and one of the industry pundits of the day wrote, in 1986: [12]

> Over the long haul, most vendors are going to migrate from TCP/IP to support Layer 4, the transport layer of the OSI model. For the short term, however, TCP/IP provides organizations with enough functionality to protect their existing equipment investment and over the long term, TCP/IP promises to allow for easy migration to OSI.

Au contraire. It was the profoundly simple protocols of the Internet that grew richer and more complex, while the OSI protocol stack became relegated to the status of an academic reference model used to describe network architecture.

Meanwhile, over on the TCP/IP standardization side, there was this wonderful, naive, glorious statement by Jon Postel in RFC 761: [13] "TCP implementation should follow a general principle of robustness. Be conservative in what you do. Be liberal in what you accept from others." It sounds like something out of the Bible, the Golden Rule as applied to computers. What a fabulous statement of philosophy! "We're not going to specify all of the details of how you interoperate; we're just going to say, 'Please do it.'"

Twitter is another good example of a fundamentally simple system. Jack Dorsey's original design sketch fit on a few lines of paper (see Figure 2, "Jack Dorsey's original vision of Twitter"). Much has grown from that sketch. There are now thousands of Twitter applications, precisely because the core Twitter service does so little. By thinking simple, Twitter allowed its users and an ecosystem of application developers to evolve new features and functionality. This is the essence of generativity.

Of course, in a government context when you say "build a simple system; let it evolve," that sounds like a real challenge. But let's remember that TCP/IP was a government-funded project. It can be done. The first step is getting a philosophy of simplicity into your work, understanding that designing foundations that others can build on is an important part of platform thinking. It's about creating the starting point, something that others can reuse and extend.

Designing simple systems is one of the great challenges of Government 2.0. It means the end of grand, feature-filled programs, and their replacement by minimal services extensible by others.

This quest for simplicity is one of the drivers behind Federal CIO Vivek Kundra's emphasis on Data.gov, a collection of APIs to government data. Kundra

Service-Oriented Architecture at Amazon

Amazon revolutionized the computer world in 2006 with the introduction of its cloud computing platform: the Elastic Compute Cloud, or EC2; the Simple Storage Service, or S3; and a series of other related services that make it possible for developers to host their applications on the same infrastructure that Amazon itself uses.

Amazon's revolutionary business model included cheap, transparent, pay-as-you-go pricing without contracts or commitments, making launching a web application a completely self-service proposition. But what's perhaps more important was the architectural commitment Amazon had made over the previous five years to building a true service-oriented architecture.* As Amazon Chief Technology Officer Werner Vogels described it in a 2008 *Information Week* interview:**

> Each of those pieces that make up the e-commerce platform are actually separate services. Whether it's Sales Rank, or Listmania, or Recommendations, all of those are separate services. If you hit one of Amazon's pages, it goes out to between 250 and 300 services to build that page.
>
> It's not just an architectural model, it's also organizational. Each service has a team associated with it that takes the reliability of that service and is responsible for the innovation of that service…. [W]e found that a lot of those teams were spending their time on the same kind of things. In essence, they were all spending time on managing infrastructure, and that was a byproduct of the organization that we had chosen, which was very decentralized.
>
> So…we decided to go to a shared-services platform and that became the infrastructure services platform that we now know in the outside world as AWS [Amazon Web Services].

Amazon is a bellwether example of why Robinson et al. urge that "federal websites themselves use the same open systems for accessing the underlying data as they make available to the public at large." Amazon's ability to deliver low-cost web services to the public started with its own total embrace of an internal web services architecture, in which Amazon's own applications are based on the same services that they offer to the public.

* http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html
**http://www.informationweek.com/news/global-cio/interviews/showArticle.jhtml?articleID=212501404

realizes that rather than having the government itself build out all of the websites and applications that use that data, providing application programming interfaces to the private sector will allow independent developers to come up with new uses for government data.

The rationale for Data.gov was laid out convincingly by David G. Robinson et al. in "Government Data and the Invisible Hand" (see Chapter 6 for an updated take on this), and the emphasis below is mine:[14]

> In the current Presidential cycle, all three candidates have indicated that they think the federal government could make better use of the Internet…. But the situation to which these candidates are responding—the wide gap between the exciting uses of Internet technology by private parties, on the one hand, and the government's lagging technical infrastructure on the other—is not new. The federal government has shown itself consistently unable to keep pace with the fast-evolving power of the Internet.
>
> In order for public data to benefit from the same innovation and dynamism that characterize private parties' use of the Internet, the federal government must reimagine its role as an information provider. Rather than struggling, as it currently does, to design sites that meet each end-user need, *it should focus on creating a simple, reliable and publicly accessible infrastructure that "exposes" the underlying data.* Private actors, either nonprofit or commercial, are better suited to deliver government information to citizens and can constantly create and reshape the tools individuals use to find and leverage public data. The best way to ensure that the government allows private parties to compete on equal terms in the provision of government data is to *require that federal websites themselves use the same open systems for accessing the underlying data as they make available to the public at large.*
>
> Our approach follows the engineering principle of separating data from interaction, which is commonly used in constructing websites. Government must provide data, but we argue that websites that provide interactive access for the public can best be built by private parties. This approach is especially important given recent advances in interaction, which go far beyond merely offering data for viewing, to offer services such as advanced search, automated content analysis, cross-indexing with other data sources, and data visualization tools. These tools are promising but it is far from obvious how best to combine them to maximize the public value of government data. Given this uncertainty, the best policy is not to hope government will choose the one best way, but to rely on private parties with their vibrant marketplace of engineering ideas to discover what works.

Data.gov reflects another key Gov 2.0 and Web 2.0 principle, namely that data is at

the heart of Internet applications. But even here, the goal is not just to provide greater access to government data, but to establish a simple framework that makes it possible for the nation—the citizens, not just the government—to create and share useful data

LESSON 3: DESIGN FOR PARTICIPATION

Closely related to the idea of simplicity is the idea of designing for participation. Participatory systems are often remarkably simple—they have to be, or they just don't work. But when a system is designed from the ground up to consist of components developed by independent developers (in a government context, read countries, federal agencies, states, cities, private sector entities), magic happens.

Open source software projects like Linux and open systems like the Internet work not because there's a central board of approval making sure that all the pieces fit together but because the original designers of the system laid down clear rules for cooperation and interoperability. (Yes, there is some oversight: Linus Torvalds and his codevelopers manage the development of the Linux kernel; the Apache Software Foundation manages the development of Apache; the Internet Engineering Task Force [IETF] and the Internet Architecture Board develop and manage Internet standards; and the World Wide Web Consortium manages web standards. But there is little or no official coordination between any of these "local" governance mechanisms. The coordination is all in the design of the system itself.)

In the case of Unix, the original design on which Linux was based, the creators started out with a philosophy of small cooperating tools[15] with standardized inputs and outputs that could be assembled into pipelines. Rather than building complex solutions, they provided building blocks, and defined how anyone could write additional building blocks of their own simply by following the same set of rules. This allowed Unix, and then Linux, to be an operating system literally created as an assemblage of thousands of different projects. While the Linux kernel, developed by Linus Torvalds, is the best known part of the operating system and gave its name to the entire system, it is a tiny part of the overall code.

The Internet took a similar approach.

Tim Berners-Lee's first implementation of the World Wide Web is a great example of the Internet approach at work. Berners-Lee was a developer at CERN, the high energy physics lab in Switzerland, trying to figure out how to make collaboration easier between scientists. To do that, he simply wrote some code. He didn't have to get permission from some central design body. All he needed was one other site to install his server. And it grew from there. He built on top of existing platform components, the Internet Protocol, the Transmission Control Protocol, the Domain Name System, which were already part of the TCP/IP stack. What he defined in addition was HTTP, a protocol for web servers and clients to exchange documents, and HTML, the data format of those documents. He wrote a sample client and a sample server, both of which he put into the public domain. The industry has been off to the races ever since.

There were a number of key design breakthroughs in the World Wide Web's "architecture of participation": [16]

- The HTML syntax for formatting a web page was not embedded in a proprietary document format. Instead, HTML documents are ordinary, human-readable text files. What's more, every web browser includes a "View Source" menu command, which allows users to study and understand the formatting of web pages, and to copy innovative new features. Many early web pages weren't written from scratch, but were modifications of other people's pages.
- Anyone could link to any other page on the Web, without the permission or knowledge of the destination page's owner. This idea was the reversal of one taken for granted in previous hypertext systems, that links must always be two-way—an agreement between the parties, so to speak. If the document on the other end of a link goes away, an error (the famous "404" seen by any web surfer) appears, but no further action is taken. This tolerance of failure is a good example of Jon Postel's Robustness Principle at work.

Another way to frame the idea that anyone could link to any other web page without permission is to say that the Web was open "by default." That is, when developers design software, they make certain choices on behalf of their users about the way that software will work unless the user intervenes to change it. For example, in the design of the World Wide Web, it was possible to make web pages that were private and accessible only after login, but unless proactive steps were taken to hide it, any web page was visible to anyone else on the Internet.

In many ways, the choice of "open by default" is the key to the breakaway success of many of the Internet's most successful sites. For example, early Internet photo-sharing sites asked their users to identify people with whom they'd like to share their photos. Flickr made "public" the default value for all photos, and soon became the gold standard for online photo sharing. Wikipedia allowed anyone to create and edit entries in their online encyclopedia, miraculously succeeding where more carefully curated online encyclopedias had failed. YouTube provided mechanisms whereby anyone could embed their videos on any web page, without coming to the central YouTube portal. Skype doesn't ask users for permission to share their bandwidth with other users, but the system is designed that way. Twitter took off because it allows anyone to follow status updates from anyone else (by default—you have to take an extra step to make your updates private), in stark contrast to previous social networks that required approval.

Cass Sunstein, now head of President Obama's Office of Information and Regulatory Affairs, is no stranger to the importance of default choices in public policy. In his book, *Nudge*, coauthored with economist Richard Thaler, he argues that "choice architecture" can help nudge people to make better decisions. [17] The most publicized policy proposal in the book was to make 401K participation "opt out" rather than "opt in" (i.e., participation by default), but the book is full of many other examples. As Sunstein and Thaler wrote:

> A choice architect has the responsibility for organizing the context in which people make decisions…. If you design the ballot voters use to choose candidates, you are a choice architect. If you are a doctor and must describe the alternative treatments available to a patient, you are a choice architect. If you design the form that new employees fill out to enroll in the company health plan, you are a choice architect. If you are a parent, describing possible educational options to your son or daughter, you are a choice architect.

And of course, if you are designing a government program, you are a choice architect. The ideas of Thaler and Sunstein have great relevance to areas such as agricultural policy (why are we subsidizing corn syrup when we face an obesity epidemic?); job creation (how do we encourage more entrepreneurs,[18] including immigrants?); health care (why does Medicare provide reimbursement for treatments that don't work?); and tax policy (where this concept is of course well understood, and the traditional bone of contention between America's political parties). Venture capitalist John Doerr's suggestion on immigration policy[19] that we "staple a Green Card to the diploma of anyone that graduates with a degree in the physical sciences or engineering" is another example of how policy defaults could have an impact on innovation. Pigovian taxes[20] are another application of this principle to government.[21]

In the context of government as a platform, the key question is what architectures will lead to the most generative outcome. The goal is to design programs and supporting infrastructure that enable "we the people" to do most of the work.

A ROBUSTNESS PRINCIPLE FOR GOVERNMENT

President Obama's memorandum calling for transparent, participatory, collaborative government is also just a statement of philosophy.[22] But it's a statement of philosophy that's fundamentally actionable in the same way that the TCP robustness principle was, or the design rules that are the heart of Unix. And even though none of these things is a formal specification, it is a set of design principles that guide the design of the platform we are collectively trying to build.

It's important to think deeply about what the three design principles of transparency, participation, and collaboration mean in the context of technology.

For example, the word "transparency" can lead us astray as we think about the opportunity for Government 2.0. Yes, it's a good thing when government data is available so that journalists and watchdog groups like the Sunlight Foundation can disclose cost overruns in government projects or highlight the influence of lobbyists. But that's just the beginning. The magic of open data is that the same openness that enables transparency also enables innovation, as developers build applications that reuse government data in unexpected ways. Fortunately, Vivek Kundra and others in the administration understand this distinction, and are providing data for both purposes.

Do It Ourselves: An Example from Hawaii

One of the most dramatic contemporary examples is a story reported by CNN, "Island DIY: Kauai residents don't wait for state to repair road":* "Their livelihood was being threatened, and they were tired of waiting for government help, so business owners and residents on Hawaii's Kauai island pulled together and completed a $4 million repair job to a state park—for free."

Especially striking in the story are the cost and time savings:

"It would not have been open this summer, and it probably wouldn't be open next summer," said Bruce Pleas, a local surfer who helped organize the volunteers. "They said it would probably take two years. And with the way they are cutting funds, we felt like they'd never get the money to fix it."

And if the repairs weren't made, some business owners faced the possibility of having to shut down….

So Slack [owner of a kayak tour business in the park], other business owners and residents made the decision not to sit on their hands and wait for state money that many expected would never come. Instead, they pulled together machinery and manpower and hit the ground running March 23.

And after only eight days, all of the repairs were done, Pleas said. It was a shockingly quick fix to a problem that may have taken much longer if they waited for state money to funnel in….

"We can wait around for the state or federal government to make this move, or we can go out and do our part," Slack said. "Just like everyone's sitting around waiting for a stimulus check, we were waiting for this but decided we couldn't wait anymore."

Now is the time for a renewal of our commitment to make our own institutions, our own communities, and our own difference. There's a kind of passivity even to most activism: collective action has come to mean collective complaint. Or at most, a collective effort to raise money. What the rebuilding of the washed out road in Polihale State Park teaches us is that we can do more than that. We can rediscover the spirit of public service, and apply the DIY spirit on a civic scale. Scott Heiferman, the founder of Meetup.com, suggests going beyond the term DIY (Do It Yourself) to embrace a new spirit of DIO: Do It Ourselves!

* http://www.cnn.com/2009/US/04/09/hawaii.volunteers.repair/index.html

Likewise, we can be misled by the notion of participation to think that it's limited to having government decision-makers "get input" from citizens. This would be like thinking that enabling comments on a website is the beginning and end of social media! It's a trap for outsiders to think that Government 2.0 is a way to use new technology to amplify the voices of citizens to influence those in power, and by insiders as a way to harness and channel those voices to advance their causes.

Participation means true engagement with citizens in the business of government, and actual collaboration with citizens in the design of government pro-

Everyone Has Something to Offer

The reflex exerted by government to gather new information, whether in pursuit of spreading around money for housing or planning its next steps in Afghanistan, is to convene an advisory committee of experts. A whole set of laws and regulations, such as the Federal Advisory Committee Act (FACA), controls this process. Such panels are typically drawn from a limited group of academics and industry experts. A list of these advisors would no doubt show a familiar pattern of high-ranking universities.

Recent popular research on crowdsourcing and the wisdom of crowds suggests a totally different approach. Asking everybody for input generates better results than just asking the experts. Certainly, a single recognized expert will tend to offer better facts, predictions, or advice than a random individual. But put a few dozen random individuals together—on the right kind of task—and the facts, predictions, or advice that shake out are better than what the experts alone produce.

The reasons behind the success of crowdsourcing are still being investigated, but the key seems to be this: in a mix of right and wrong answers, the wrong ones tend to cancel each other out, leaving the right ones. This is the secret behind the famous appeals to the audience in the game show *Who Wants to Be a Millionaire*, as well as the success of prediction markets such as the University of Iowa's Electronic Market.*

Wikipedia, which invariably makes a central appearance in every reference to crowdsourcing, plays the different opinions of the crowd against each other in more explicit ways. On relatively uncontroversial articles, contributors are expected to discuss their differences and reach consensus. This process is aided by a rarely cited technical trait of web pages: because they present no artificial space limitations, there can always be room for another point of view. On controversial topics, Wikipedia has over the years developed more formal mechanisms, but the impetus for change still wells up from the grassroots.

It's also worth mentioning, in regard to crowdsourcing, the use of low-paid or volunteer labor to carry out simple tasks such as identifying the subjects of photographs. These are called Mechanical Turk projects, in reference to a crowdsourcing technology platform provided by Amazon.com, which is itself named after an eighteenth-century hoax** in which a person pretended to be an intel-

grams. For example, the Open Government Brainstorming conducted by the White House is an attempt to truly engage citizens in the making of policy, not just to hear their opinions after the fact.[23]

Open government APIs enable a different kind of participation. When anyone can write a citizen-facing application using government data, software developers have an opportunity to create new interfaces to government.

ligent machine; in the modern incarnation, thousands of people are serving as functions invoked by a computer application.

Crowdsourcing has already slipped into government procedures in low-key ways. Governments already use input from self-appointed members of the public on all kinds of things, ranging from reports of potholes to anonymous tips that put criminals behind bars.

One of the key skills required of both technologists and government officials is how best to aggregate public opinion or data produced by public actions to reveal new information or patterns. For example, cities learn a lot about neighborhoods by aggregating crime reports from residents. They could understand their needs for broadband network access much more accurately if they took resident reports into account and didn't depend just on what the broadband vendors told them (because geographic anomalies often cause dead zones in areas that the vendors claim to serve).

In general, people can provide input on several levels:

Observations such as reports of potholes and crimes

Feedback on government proposals

New ideas generated through brainstorming sessions

Full-fledged applications that operate on publicly available data

Some of those applications may operate on existing government data, but they can also be designed to collect new data from ordinary people, in a virtuous circle by which private sector applications (like SeeClickFix) increase the intelligence and responsiveness of government.

Governments are more likely to use some form of filtering than to rely on public consensus, as Wikipedia does. The combination of free debate among the public and some adult supervision from a government official makes a powerful combination, already seen in the open government brainstorming session mentioned in Lesson 3.

Finally, crowds can produce data without even realizing it—implicit data that smart programmers can collect and use to uncover whole worlds of information. In fact, smart programmers in the private sector have been doing that for years. Lesson 5 covers this trend.

—Andy Oram

* http://www.biz.uiowa.edu/iem/index.cfm

** http://en.wikipedia.org/wiki/Amazon_Mechanical_Turk

Perhaps most interesting are applications and APIs that allow citizens to actually replace functions of government, in a self-service analogue to Craigslist. For example, FixMyStreet, a project developed by UK nonprofit mySociety, made it possible for citizens to report potholes, broken streetlights, graffiti, and other problems that would otherwise have had to wait on an overworked government inspector. This concept has now been taken up widely by forward-thinking cities as well as entrepreneurial companies like SeeClickFix, and there is even a stan-

dard—Open311—for creating APIs to city services of this kind, so that third-party developers can create applications that will work not just for one city, but for every city.

Taking the idea of citizen self-service even further, you can imagine government using a platform like Meetup to support citizens in self-organizing to take on major projects that the government would otherwise leave undone. Today, there are thousands of civic-minded meetups around issues like beach, road, and waterway cleanups. How many more might there be if local governments themselves embraced the idea of harnessing and supporting citizen concerns as expressed by self-organized meetups?

Citizen self-organization is a powerful concept. It's worth remembering that early in our nation's history, many functions now handled by government were self-organized by citizens: militias, fire brigades, lending libraries, not to mention roads, harbors and bridges. And even today, volunteer fire departments play a major role in protecting many of our communities. Traditional communities still perform barn raisings. Those of us who spend our time on the Internet celebrate Wikipedia, but most of us have forgotten how to do crowdsourcing in the physical world.

LESSON 4: LEARN FROM YOUR "HACKERS"

The secret of generative systems is that the most creative ideas for how a new platform can be used don't necessarily come from the creators of the platform. It was not IBM but Dan Bricklin and Bob Frankston (VisiCalc), Mitch Kapor (Lotus 1-2-3), and Bill Gates who developed the "killer applications" that made the IBM personal computer such a success. It was Tim Berners-Lee, not Vint Cerf and Bob Kahn (the designers of the Internet's TCP/IP protocol), who developed the Internet's own first killer application, the World Wide Web. And it was Larry Page and Sergey Brin, not Tim Berners-Lee, who figured out how to turn the World Wide Web into a tool that revolutionized business.

Such stories suggest how technology advances, as each new generation stands on the shoulders of preceding giants. Fundamental technology breakthroughs are often not exploited by their creators, but by a second generation of entrepreneurs who put it to work.

But advances don't just come from entrepreneurs playing by the rules of new platforms. Sometimes they come from those who break the rules. MIT professor Eric von Hippel has written extensively about this phenomenon, how "lead users" [24] of a product push it to its limits and beyond, showing vendors where their product wants to go, in much the way that rushing water carves its own path through the earth.

There's no better contemporary example than Google Maps, introduced in 2005, nearly 10 years after MapQuest, the first Internet site providing maps and directions. Yet today, Google Maps is the dominant mapping platform by most measures. How did this happen?

When Google Maps was introduced, it featured a cool new AJAX (Asynchronous JavaScript and XML) interface that made it easy to dynamically drag and zoom the map. But there was a hidden feature as well, soon discovered by independent developers. Because JavaScript is interpreted code, it was possible to extract the underlying map coordinate data. A programmer named Paul Rademacher introduced the first Google Maps mashup, HousingMaps.com, taking data from another Internet site, Craigslist.org, and creating an application that put Craigslist apartment and home listings onto a Google Map.

What did Google do? Far from shutting down Rademacher's site and branding him a pirate, Google hired him, and soon put out an API that made it easier for anyone to do what he did. Competitors, who had long had mapping APIs but locked them up behind tightly controlled corporate developer programs, failed to seize the opportunity. Before long there were thousands of Google Maps mashups, and mapping had become an integral part of every web developer's toolkit.

Today, according to the site ProgrammableWeb.com, which tracks mashups and reuse of web APIs, Google Maps accounts for nearly 90% of all mapping mashups, versus only a few percent each for MapQuest, Yahoo!, and Microsoft, even though these companies had a huge head start in web mapping.

There are potent lessons here for governments opening up access to their data via APIs. Developers may use those APIs in unexpected ways. This is a good thing. If you see signs of uses that you didn't consider, respond quickly, adapting the APIs to those new uses rather than trying to block them.

In this regard, consider an instructive counterexample to Google Maps from the government sector. The New York Metropolitan Transit Authority recently attempted to stop the distribution of an iPhone app called StationStops, which provides schedule information for Metro-North trains. After a legal battle, the MTA relented.[25] Other cities, meanwhile, realized that having independent developers build applications that provide information to citizens is a benefit both to citizens and to overworked government agencies, not "copyright infringement and intellectual property theft," as the MTA had originally maintained.

The whole point of government as a platform is to encourage the private sector to build applications that government didn't consider or doesn't have the resources to create. Open data is a powerful way to enable the private sector to do just that.

DATA IS THE "INTEL INSIDE"

Open data is important not just because it is a key enabler of outside innovation. It's also important to place in the context of current Internet business models. To explain, we require a brief excursion.

One of the central platform lessons of the PC era is summed up in a principle that Harvard Business School Professor Clayton Christensen called "the law of conservation of attractive profits":[26]

When attractive profits disappear at one stage in the value chain because a product becomes modular and commoditized, the opportunity to earn attractive profits with proprietary products will usually emerge at an adjacent stage.

As the IBM PC—built from commodity off-the-shelf parts—became dominant, hardware margins declined, over time becoming razor thin. But according to Christensen's law, something else became valuable, namely software, and Microsoft was soon earning the outsized profits that once were claimed by IBM. But even in an ecosystem of standard off-the-shelf parts, it is sometimes possible to corner a market, and that's just what Intel did when it broke with IBM's policy that every component had to be available from at least two suppliers, and refused to license its 80386 design to other chip manufacturers. That was the origin of the other half of the famous "Wintel" duopoly of Microsoft and Intel. If you can become the sole source of an essential commodity that is key to an otherwise commoditized product, you too can aspire to a logo like the ubiquitous "Intel Inside."

Reflecting on the role of open source software and open protocols and standards in commoditizing the software of the Internet, I concluded in my 2003 paper "The Open Source Paradigm Shift"[27] that something similar would happen on the Internet. Exactly what that was didn't become clear to me till 2005, when I wrote "What Is Web 2.0?"[28]

If there's one lesson that is central to the success of Web 2.0, it's that data and the algorithms that produce value from it—not the software APIs and applications that were the key to the PC era—are the key to marketplace advantage in today's Internet. Virtually all of the greatest Internet success stories, from eBay, Craigslist, and Amazon through Google, Facebook, and Twitter, are data-driven companies.

In particular, they are companies whose databases have a special characteristic: they get better the more people use them, making it difficult for competitors to enter the market. Once eBay or Craigslist had a critical mass of buyers and sellers, it became far more difficult for competitors to enter the market. Once Google established a virtuous circle of network effects among its AdWords advertisers, it was hard for others to achieve similar results.

The Internet business ecosystem can thus be seen as a competition to establish monopolies over various classes of data. It is indeed data that is the "Intel Inside" of the Internet.

What does this have to do with Government 2.0? If data is indeed the coin of the realm of Internet business models, it stands to reason that companies will find advantage in taking data created at public expense, and working to take control of that data for private gain.

Consider the story of Routesy, an application providing iPhone users with bus arrival data in the San Francisco Bay Area. Like StationStops in New York, it was taken down from the iPhone App Store after a legal complaint. While Muni (the San Francisco transit authority) was supportive of Routesy and believed that its data was public, the contract that Muni had signed with technology provider NextBus allowed NextBus to claim copyright in the data.[29] If you want to have the kind of responsiveness that Google showed in supporting HousingMaps.com and

launching the Google Maps mashup ecosystem, you have to make sure that public data remains public!

Fortunately, the NextBus/Routesy dispute was resolved, like MTA/StationStops, with a win for the public sector. The San Francisco Municipal Transit Authority has now released an XML API to the NextBus data.[30]

LESSON 5: DATA MINING ALLOWS YOU TO HARNESS IMPLICIT PARTICIPATION

When thinking about user participation and the co-creation of value, it's easy to focus on technology platforms that explicitly feature the creations of their users, like Wikipedia, YouTube, Twitter, Facebook, and blogs. Yet in many ways, the breakthroughs in Web 2.0 have often come from exploring a far wider range of possibilities for collaboration:

- Open source technology platforms such as the TCP/IP protocol suite and utilities created as part of Berkeley Unix, as well as Linux, Apache, and MySQL, and open source programming languages such as Perl, Python, PHP, and Ruby, all built and maintained by collaborative communities, provided the fundamental building blocks of the Internet as we know it today.
- The World Wide Web itself has an architecture of participation. Anyone can put up a website and can link to any other website without permission. Blogging platforms made it even easier for any individual to create a site. Later platforms like Facebook and Twitter are also enablers of this kind of explicit participation.
- First-generation web giants like Yahoo! got their start by building catalogs of the content assembled by the participatory multitudes of the Net, catalogs that later grew into search engines. eBay aggregated millions of buyers and sellers into a global garage sale. Craigslist replaced newspaper classified advertising by turning it all into a self-service business, right down to the policing of inappropriate content, having users flag postings that they find offensive. Even Amazon.com, nominally an online retailer, gained competitive advantage by harnessing customers to provide reviews and ratings, as well as using their purchase patterns to make automated recommendations.
- Google's search engine dominance began with two brilliant insights into user participation. First, the PageRank algorithm that Larry Page and Sergey Brin created while still at Stanford was based on the realization that every link on the World Wide Web was a kind of vote on the value of the site being pointed to by that link. That is, every time any of us makes a link to another site on the Web, we're contributing to Google. Second, Google realized that it could provide better advertising results not by selling advertisements to the highest bidder, but by measuring and predicting user click-through rates on ads. A $10 ad that is twice as likely to be clicked on is worth more than a $15 ad. Google could only deliver these results by understanding that every click on a Google search result is a kind of user contribution. Since then, Google has gone on to

mine user participation in many other aspects of its core business as well as in new businesses, including speech recognition, location-based services, automated translation, and much more. Google is a master at extracting value from implicit participation. It makes use of data that its users provide simply in going about their lives on the Internet to provide them with results that quite literally could not exist without them.

Just as Google has become the bellwether company of the Internet era, it is actually systems for harnessing implicit participation that offer some of the greatest opportunities for Government 2.0.

There are great examples to be found in health care. As costs soar, we discover that costs and outcomes aren't correlated. Atul Gawande's *New Yorker* article[31] on this disconnect—outlining how McAllen, Texas, the city with the highest health care costs in the U.S., also had the worst health outcomes—led to what Health and Human Services CTO Todd Park referred to in a conversation with me as a "holy cow moment." Todd is now working on what he calls a "holy cow machine," a set of services that will allow every city to understand how its health care costs and outcomes compare to those of other cities.

We have all the data we need—generated by the interactions of our citizens with our health care system—to understand how to better align costs and outcomes. Taking this idea to its full potential, we need to get beyond transparency and, as Google did with AdWords, start building data-driven feedback loops right into the system. Google's tools for estimating the effectiveness of keyword advertising are available to advertisers, but that's wonky, back-office stuff. The real magic is that Google uses all its data expertise to directly benefit its users by automatically providing better search results and more relevant advertisements. The most amazing thing about Google is how dynamically the prices for its advertising are set. *Every single Google search has its own automated ad auction. The price is set dynamically, matching supply and demand, seven or eight billion times a day.* Only financial markets operate at this kind of speed and scale.

A Gov 2.0 analogue would not just be a "holy cow machine" for transparency; it might, for example, be a new, dynamic pricing system for Medicare. Currently, an outside advisory board makes recommendations to Congress on appropriate Medicare reimbursement rates. As David Leonhardt noted in the *New York Times*, "Congress generally ignores them, in deference to the various industry groups that oppose any cuts to their payments."[32] Leonhardt's solution: an independent body, akin to the Federal Reserve, empowered to set reimbursement rates in the same way the Fed sets interest rates.

But shouldn't such a body go even further than periodic resets? Technology would allow us actually to manage reimbursements in much the same way as Google dynamically adjusts its algorithms to produce optimal search results and optimal ad placements. Google takes into account hundreds of factors; so too could a Medicare rate-setting algorithm. To take two examples from Leonhardt's article:

> Each year, about 100,000 people die from preventable infections they contract in a hospital. When 108 hospitals in Michigan instituted a simple process to prevent some of these infections, it nearly eliminated them. If Medicare reduced payments for the treatment of such infections, it would give hospitals a huge financial incentive to prevent them….
>
> There are a handful of possible treatments for early-stage prostate cancer, and the fastest-growing are the most expensive. But no one knows which ones work best.

By measuring outcomes and linking reimbursements to those outcomes—rather than the current "fee for service" model, which encourages unnecessary procedures—Medicare could pave the way to a real revolution in health care.

Because of the political difficulty of such an intervention, it's unlikely that Medicare would be allowed to unilaterally introduce such an algorithmic payment system. As a result, I do suspect that this kind of innovation will come first from the private sector, which will trounce its competition in the same way that Google trounced its competitors in the search advertising market. As a platform provider, though, it's possible to see how government investment in the data infrastructure to measure and report on outcomes could jump-start and encourage private sector investment.

Real-time linkage of health costs and outcomes data will lead to wholesale changes in medical practice when an innovative health care provider uses them to improve its effectiveness and lower its costs. Such a breakthrough would sooner or later be copied by less effective providers. So rather than attempting to enforce better practices through detailed regulations, a Government 2.0 approach would use open government data to enable innovative private sector participants to improve their products and services. And to the extent that the government itself is a health care provider (as with the Veterans Administration) or medical insurer (as with Medicare), it can best move the ball forward by demonstrating in its own operations that it has been able to harness technology to get the job done better and more cost-effectively.

LESSON 6: LOWER THE BARRIERS TO EXPERIMENTATION

In a memorable moment during the Apollo 13 moon mission, when mechanical failures required that the mission be aborted and the astronauts rescued using only materials on board the craft, mission controller Gene Kranz famously said, "Failure is not an option." In that case, he was right. But far too often, government programs are designed as though there is only one right answer, and with the assumption that the specification developed by a project team must by definition be correct.

In reality, for most projects, failure is an option. In fact, technology companies embrace failure, experimentation, and rapid iteration.

This has been true long before the latest wave of technology companies. In

describing his quest for a working electric light bulb, Thomas Edison said, "I didn't fail 10,000 times. I succeeded 10,000 times in figuring out something that did not work."

You can conceive of the technology marketplace as a series of competitive experiments. But even within a single company, one of the advantages of web-based business models is the ease of experimentation. Companies routinely run A/B tests of new features on subsets of their users. They add and subtract features in real time in a process of constant improvement that I've sometimes called the "perpetual beta."

More recently, thinkers such as Steve Blank and Eric Ries have described an idea that Ries refers to as "the lean startup," in which he describes exploring the market via a series of "minimal viable products," each of which tells you more about what the market really wants. [33]

This is at great variance with typical government thinking, which, by ignoring the possibility of failure, paradoxically creates the conditions that encourage it. Government 2.0 requires a new approach to the design of programs, not as finished products, perfected in a congressional bill, executive order, or procurement specification, but as ongoing experiments.

Quite frankly, this is likely the greatest challenge in Government 2.0, not only because of the nature of the government procurement process, but also because government programs are often dictated by legislation, or by agency regulations that are outside the scope of the agency actually making the decisions. What's more, while the commercial marketplace benefits from Schumpeterian "creative destruction," government programs are rarely scrapped or sunsetted.

This is all the more reason why government programs must be designed from the outset not as a fixed set of specifications, but as open-ended platforms that allow for extensibility and revision by the marketplace. Platform thinking is an antidote to the complete specifications that currently dominate the government approach not only to IT but to programs of all kinds.

A cultural change is also required. Empowering employees to "fail forward fast" accepts and acknowledges that even when an experiment fails, you will still learn something. Software and web culture not only embraces this mindset, but revels in it—you never know which idea will be the million-dollar idea. Once the cost of that experimentation is reduced, you can quickly scrap a product or feature that no one uses and accept that it just wasn't the thing that needed to be built after all.

Finally, it is essential for best practices—and even working code—to be shared between agencies of the federal government, between states, and between municipalities. After all, as Justice Louis Brandeis wrote in 1932, "It is one of the happy incidents of the federal system that a single courageous state may, if its citizens choose, serve as a laboratory; and try novel social and economic experiments without risk to the rest of the country." [34]

How Platform Thinking Changes the Big Government/Small Government Debate

It should be obvious by now that platform thinking provides a real alternative to the endless argument between liberals and conservatives that has so dominated U.S. political discourse in recent decades. The idea that we have to choose between government providing services to citizens and leaving everything to the private sector is a false dichotomy. Tim Berners-Lee didn't develop hundreds of millions of websites; Google didn't develop thousands of Google Maps mashups; Apple developed only a few of the tens of thousands of applications for the iPhone.

Being a platform provider means government stripped down to the essentials. A platform provider builds essential infrastructure, creates core applications that demonstrate the power of the platform and inspire outside developers to push the platform even further, and enforces "rules of the road" that ensure that applications work well together.

LESSON 7: LEAD BY EXAMPLE

When Microsoft introduced Microsoft Windows, it didn't just introduce the platform; it introduced two applications, Microsoft Word and Microsoft Excel, that showed off the ease of use that came with graphical user interfaces. When Apple introduced the iPhone, it didn't even introduce the platform until its second year. First, it built a device with remarkable new features and a suite of applications that showed off their power.

Despite everything I've said about the importance of a platform provider not competing with its developer ecosystem, it's also a mistake to think that you can build a platform in the abstract. A great platform provider does things that are ahead of the curve and that take time for the market to catch up to. It's essential to prime the pump by showing what can be done.

This is why, for example, Apps.DC.gov, the "App Store" for the city of Washington, D.C., provides a better Gov 2.0 platform model than the federal equivalent Data.gov (see Figure 3, "Apps.DC.gov home page"). Although Apps.gov provides a huge service in opening up and promoting APIs to all the data resources of the federal government, it's hard to know what's important, because there are no compelling "applications" that show how that data can be put to use. By contrast, Apps.DC.gov features a real app store, with applications written by the city of Washington, D.C.'s own technology team (or funded by them) demonstrating how to use key features. D.C. then took the further step of highlighting, at a top level, third-party apps created by independent developers. This is a model for every government app store to follow.

It is true that the sheer size and scope of the federal data sets, as well as the remoteness of many of them from the everyday lives of citizens, makes for a bigger challenge. But that's precisely why the federal Gov 2.0 initiative needs to do

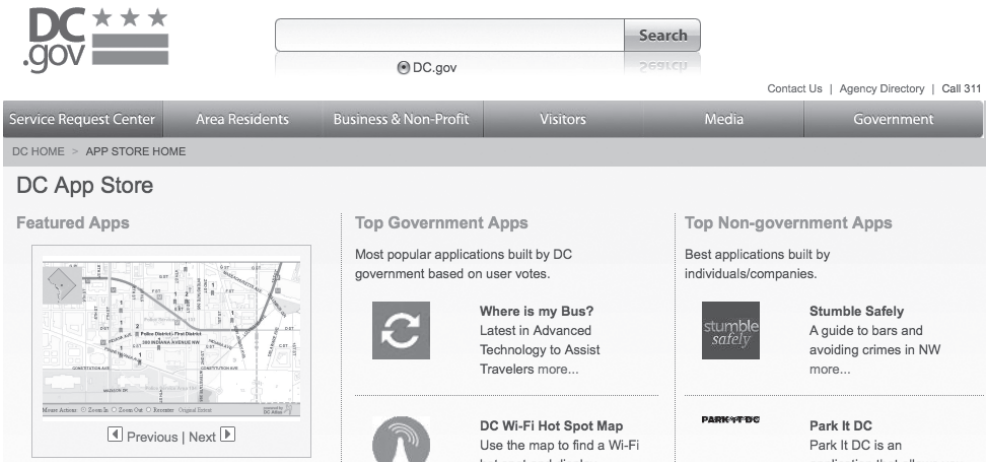

Figure 3. Apps.DC.gov home page

deep thinking about what federal data resources and APIs will make the most difference to citizens, and invest strategically in applications that will show what can be done.

But the idea of leading by example is far bigger than just Data.gov. Once again, consider health care.

If the current model of "health care reform" were an operating system, it would be Windows Vista, touted as a major revisioning of the system, but in the end, a set of patches that preserve what went before without bringing anything radically new to the table.

If the government wants buy-in for government-run health care, we need the equivalent of an iPhone for the system, something that re-envisions the market so thoroughly that every existing player needs to copy it. I've suggested that an opportunity exists to reinvent Medicare so that it is more efficient than any private insurance company, and to make the VA better than any private hospital system. But being realistic, technology teaches us that it's always harder to refactor an existing system or application than it is to start fresh.

That's why the "public option" proposed in some current health care bills is such an opportunity. Can we create a new health insurance program that uses the lessons of technology—open standards, simplicity in design, customer self-service, measurement of outcomes, and real-time response to what is learned, not to mention access via new consumer devices—to improve service and reduce costs so radically that the entire market follows?

This is the true measure of Gov 2.0: does it make incremental changes to the existing system, or does it constitute a revolution? Considering the examples of Microsoft, Google, Amazon, Apple, and other giants of the technology world, it's clear that they succeeded by changing all the rules, not by playing within the existing system. The personal computer, the World Wide Web, and the iPhone have

each managed to simultaneously bring down costs while increasing consumer choice—each by orders of magnitude.

They did this by demonstrating how a radically new approach to existing solutions and business models was, quite simply, orders of magnitude better than what went before.

If government is a platform, and Gov 2.0 is the next release, let's make it one that shakes up—and reshapes—the world.

PRACTICAL STEPS FOR GOVERNMENT AGENCIES

1. Issue your own open government directive. San Francisco Mayor Gavin Newsom has done just that. You might consider his Open Data Executive Directive as a model.[35]

2. As Robinson et al. propose, create "a simple, reliable and publicly accessible infrastructure that 'exposes' the underlying data" from your city, county, state, or agency. Before you can create a site like Data.gov, you must first adopt a data-driven, service-oriented architecture for all your applications. The "Eight Open Government Data Principles" document outlines the key requirements for open government data.[36]

3. "Build your own websites and applications using the same open systems for accessing the underlying data as they make available to the public at large" (Robinson et al. again).[37]

4. Share those open APIs with the public, using Data.gov for federal APIs and creating state and local equivalents. For example, cities such as San Francisco (DataSF.org) and Washington, D.C. (Data.DC.gov and Apps.DC.gov) include not only data catalogs but also repositories of apps that use that data, created by both city developers and the private sector.

5. Share your work with other cities, counties, states, or agencies. This might mean providing your work as open source software, working with other governmental bodies to standardize web services for common functions, building a common cloud computing platform, or simply sharing best practices. Code for America is a new organization designed to help cities do just that.

6. Don't reinvent the wheel: support existing open standards and use open source software whenever possible. (Open311 is a great example of an open standard being adopted by many cities.) Figure out who has problems similar to yours, and see if they've done some work that you can build on.

7. Create a list of software applications that can be reused by your government employees without procurement.

8. Create an "app store" that features applications created by the private sector as well as those created by your own government unit (see Apps.DC.gov).

9. Create permissive social media guidelines that allow government employees to engage the public without having to get pre-approval from superiors.

10. Sponsor meetups, code camps, and other activity sessions to actually put citizens to work on civic issues.

1. *The Founders' Constitution*, Chapter 4, Document 34.
2. *The Next Government of the United States: Why Our Institutions Fail Us and How to Fix Them*, Donald Kettl, W. W. Norton & Company, 2008.
3. *The Cathedral & the Bazaar*, Eric Raymond, O'Reilly, 1999.
4. http://radar.oreilly.com/2009/07/itunes-app-store-incubation-period-increases.html
5. http://www.microsoft.com/about/companyinformation/ourbusinesses/profile.mspx
6. *The Future of the Internet-And How to Stop It*, Jonathan Zittrain, Yale University Press, 2008.
7. https://www.fbo.gov/index?tab=core&s=opportunity&mode=form&id=d208ac8b8687dd9c6921d2633603aedb&tabmode=list&cck=1&au=&ck=
8. http://radar.oreilly.com/2009/10/defense-department-releases-op.html
9. http://healthit.hhs.gov/blog/faca/
10. http://en.wikipedia.org/wiki/Creative_destruction
11. *Systemantics: How Systems Work and Especially How They Fail*, John Gall, Quadrangle, 1977.
12. "TCP/IP: Stairway to OSI," Robert A. Moskowitz, Computer Decisions, April 22, 1986.
13. DOD Standard: Transmission Control Protocol report.
14. "Government Data and the Invisible Hand," David G. Robinson, Harlan Yu, William Zeller, and Edward W. Felten, *Yale Journal of Law & Technology*, Vol. 11, 2009.
15. Unix Programming Environment, Brian W. Kernighan and Rob Pike, Prentice Hall, 1984.
16. http://www.oreillynet.com/pub/a/oreilly/tim/articles/architecture_of_participation.html
17. *Nudge: Improving Decisions About Health, Wealth, and Happiness*, Richard H. Thaler and Cass R. Sunstein, Penguin, 2009.
18. http://www.feld.com/wp/archives/2009/09/the-founders-visa-movement.html
19. http://blog.actonline.org/2008/11/doerr-staple-a-green-card-to-diplomas.html
20. http://en.wikipedia.org/wiki/Pigovian_tax
21. For an excellent summary of Thaler and Sunstein's ideas on government policy, see Nudge-ocracy: Barack Obama's new theory of the state.
22. http://www.whitehouse.gov/the_press_office/TransparencyandOpenGovernment/
23. http://www.whitehouse.gov/blog/wrap-up-of-the-open-government-brainstormingparticipation/
24. http://en.wikipedia.org/wiki/Lead_user
25. "M.T.A. Is Easing Its Strict, Sometimes Combative, Approach to Outside Web Developers," *New York Times*, September 27, 2009.
26. *The Innovator's Solution: Creating and Sustaining Successful Growth*, Clayton M. Christensen and Michael E. Raynor, Harvard Business Press, 2003.
27. http://tim.oreilly.com/articles/paradigmshift_0504.html
28. http://oreilly.com/web2/archive/what-is-web-20.html
29. "Does A Private Company Own Your Muni Arrival Times?", SF Appeal, June 25, 2009.
30. http://www.sfmta.com/cms/asite/nextmunidata.htm
31. "The Cost Conundrum," Atul Gawande, *The New Yorker*, June 1, 2009.
32. "Falling Far Short of Reform," David Leonhardt, *The New York Times*, November 10, 2009.
33. http://www.startuplessonslearned.com/2009/10/inc-magazine-on-minimum-viable-product.html
34. http://www.whitehouse.gov/blog/2009/11/19/open-government-laboratories-democracy
35. http://www.sfmayor.org/wp-content/uploads/2009/10/ED-09-06-Open-Data.pdf
36. http://resource.org/8_principles.html
37. "Government Data and the Invisible Hand," David G. Robinson, Harlan Yu, William Zeller, and Edward W. Felten, *Yale Journal of Law & Technology*, Vol. 11, 2009