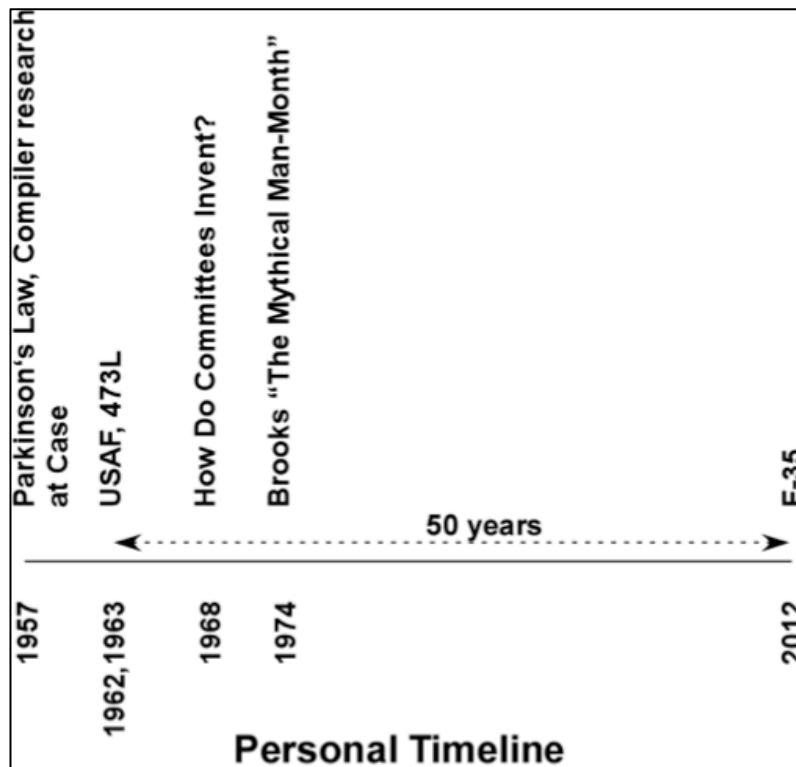


The Tower of Babel and the Fighter Plane

RESER2013 Keynote Address
October 9, 2013

Mel Conway

I'm going to talk about something important that has occurred over a 50-year period in my lifetime in the field of Defense Department large-system procurement. My story is a good news-bad news story. The subject is system complexity.



My exposure to DOD system acquisition practices began in 1962, when I was a lieutenant in the Air Force Electronic Systems Division in Massachusetts, which designed and procured large computer systems for the Air Force. I was involved in one major source selection, the 473L Air Force Logistics support computer system, which (I assume) sits somewhere in the Pentagon helping staff officers plan the flow of large operations. It had a continually updated database with a query language that would help the user answer questions like: name the bases within 1000 miles of Paris that have full tankers complete with ready crews can fly within the next 24 hours.

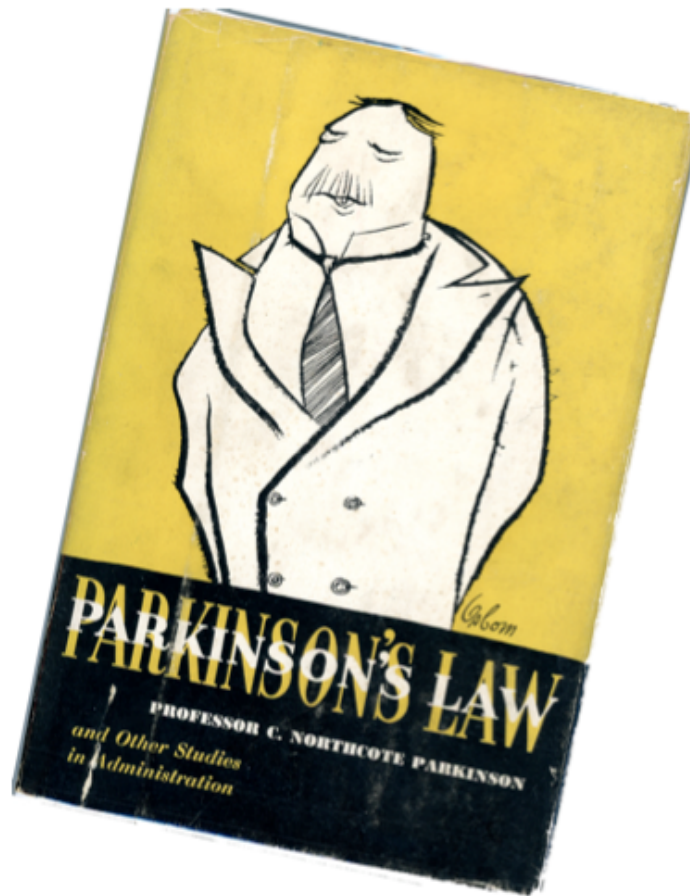
I left active duty two years later and went on to do my own things, but just in the last few months I discovered the largest and most complex weapons system procurement in the history of the USA. It struck me as a perfect case study, so I tore up my old talk and created this one.

So this talk is about how (particularly when it comes to designing large weapons systems) we have dug ourselves

into a complexity hole with no apparent way out. This is causing worldwide consternation, as I will describe.

My experience with 473L was formative, and was an influence in my formulation of Conway's Law. The process at the time was rigid waterfall, governed by legalistic regulations. In this case it led to the choice of noncommercial computer hardware that I had never heard of and spawned a lawsuit along the way. I was a fly on the wall witnessing a process that seemed disconnected from the reality I understood about how to build a good system.

Fifty years later I returned to this world and am now able to see how the situation has progressed over this interval of time.



In 1957 C Northcote Parkinson, a history professor at Raffles University at the University of Malaya, published “Parkinson’s Law and other Studies in Administration.” The book has 10 chapters, each discussing a different phenomenon of bureaucratic life. In my view, 8 of these chapters are satirical, but all hit on some nugget of truth. For example he has a chapter on the board of directors budget meeting. This law states that the amount of time spent on a budget line item varies inversely with the value of the item: for example, the nuclear reactor took 15

minutes and the coffee pot was considered for 1¼ hour and then tabled in order to obtain more information.

I'll only briefly mention one of the two chapters that struck me as particularly insightful. In the first, Parkinson focuses on the event in which the successful organization dedicates its brand new headquarters building. Parkinson states that that event marks the decline of the organization. He cites many examples from history.



League of Nations, Geneva

The League of Nations building comes to mind.



The contemporary example that is particularly relevant to me is Apple's construction of its new headquarters in Cupertino.

Now, on to the important chapter. The first sentence of the first chapter of Parkinson's book is one that most people know: "Work expands to fill the time available for its completion." But there's much more to what Parkinson did on this subject. He actually did some statistical studies on the manpower levels of several elements of the British bureaucracy and he found an amazing consistency: bureaucracies, like bacteria colonies, grow at a stable, internally defined rate averaging 5.5 percent per year over time. Most importantly, the size of the organization is totally independent of the amount of its useful output.

Colonial Office Staff Levels, from 1935 to 1954

1935	1939	1943	1947	1954
372	450	817	1139	1661

This table is from the book; it shows the staffing levels of the British Colonial Office, which administered the colonies, from years 1935 to 1954. Notice the growth during World War 2, when many of the colonies were in enemy hands and not even under the governance of the Colonial Office. In 1954 the number of colonies had shrunk and the Colonial Office was at its largest.



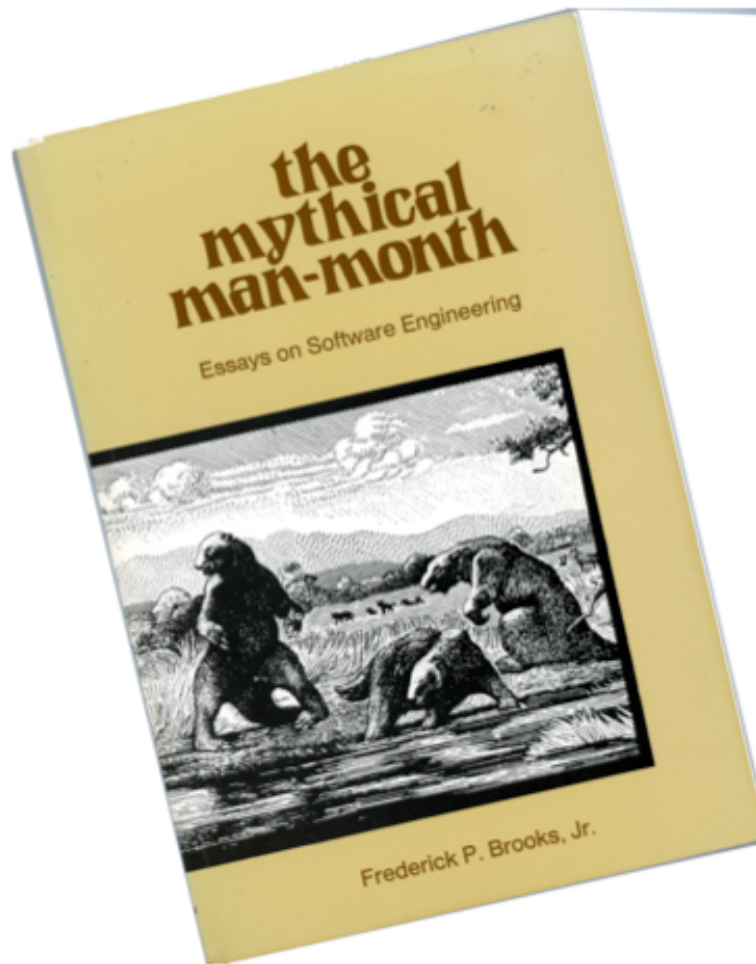
But now here is Parkinson's big contribution. He explains how a bureaucracy can be fully busy totally independent of the amount of its useful output. Here's how it happens. As each new branch is created it justifies itself by challenging the established order. Thus, after a while, the organization is fully occupied in internal political warfare. I remember one participant observing that the enemy is not the country across the ocean but the office across the hall.

I experienced this warfare first-hand. In the 1970s I was consulting to the Bureau of Standards, writing the ANSI standard document for the Mumps language (now called M). Mumps was a minicomputer-based time-shared database system with which people in the VA hospitals were writing many of their own applications. I spent a week or two at VA headquarters in DC and found myself in a political snake pit. I had walked into a power struggle in the headquarters between advocates of a headquarters-based mainframe-centered IT structure and the hospital-based Mumps approach. There were massive position papers and memos flying back and forth on the topic, each requiring a written response. Then I discovered how this warfare was powered.



Bureaucratic Artillery





I was walking down the hall in VA headquarters and saw this machine sitting in a corner: a Selectric typewriter connected to a magnetic tape drive, IBM's first mechanical word processor. I puzzled for a minute and had my epiphany: *bureaucratic artillery*! This machine greatly increases writer productivity and, in combination with a photocopier, can bombard the enemy with text. Parkinson never would have believed how seriously he had been taken.



Around the time of this VA experience Fred Brooks published “The Mythical Man-Month”, his classic set of essays on the management of very large software projects. Brooks managed the design of Operating System 360, which attempted to span the entire range of IBM’s machines with one operating system. Each of his 15 chapters covers a different aspect of the management of large system development. (Incidentally, Brooks originated the name “Conway’s Law” in this book.)

The one lesson from the book I want to cite here, because we're going to see it later, is what Brooks called the "Second System Effect." (Now I'm making a synthesis here for effect; he didn't say it just this way.) It takes four generations of experience to learn to build large systems well.

Brooks's Second System Effect

0. Plan to throw one away; you will anyway. 
1. First system. 
2. Second System. 
3. Finally ready. 

These generations are:

0. Plan to throw one away; you will anyway.
1. Second try: The first generation goes into production. It lacks features and nobody is particularly happy, but it does the job.
2. Third try: This is the second system. Everybody has been waiting for the chance to build it. They put in all the features they had to leave out before. Consequently, it is massive and inefficient, and might fall of its own weight.
3. Fourth try: A wiser team, having had all this experience, is now ready to build good systems.

Brooks was careful to distinguish the development of very large systems and smaller systems, whose management requirements are different. The Pentagon has been leading the way in learning how to manage the development of massively complex weapons systems.

Now let's move ahead 50 years to the present. Here is where Conway's Law comes in. Parkinson's 5½ percent per year growth rate for 50 years is equivalent to multiplication by a factor of about 14. Not that this is particularly significant numerically, but if you accept a connection between the complexity of a design organization (remember, the Pentagon is the designer) and the complexity of its products, an order of magnitude growth certainly suggests that the complexity issues of today's systems are going to be qualitatively different from when I was a lieutenant. Now I'm going to show you an example.



Lockheed Martin F-35 Lightning II Joint Strike Fighter

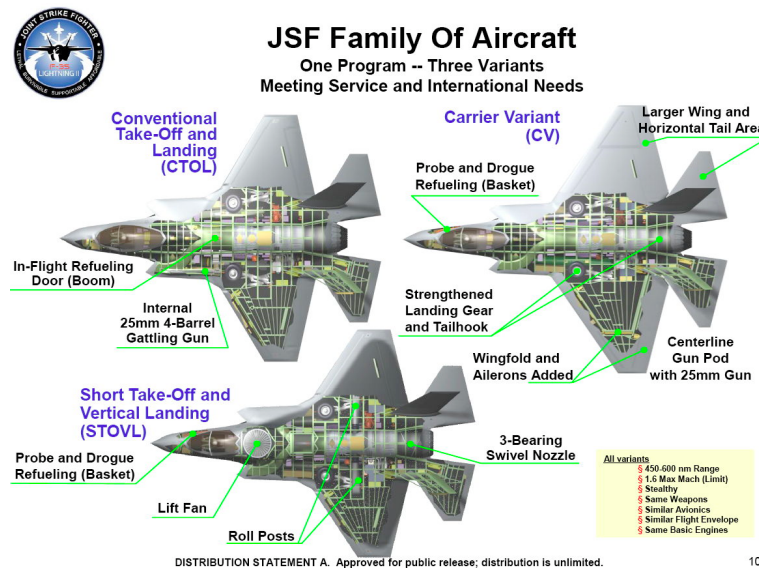
The contemporary system I want to describe today is a single-seat, single-engine fighter plane: the Lockheed Martin F-35 Lightning II Joint Strike Fighter. The name alone shows a kind of perverse application of Conway's Law: you can see that two different groups were pushing their own agendas to name the plane.



**Lockheed P-38 Lightning
Interceptor**

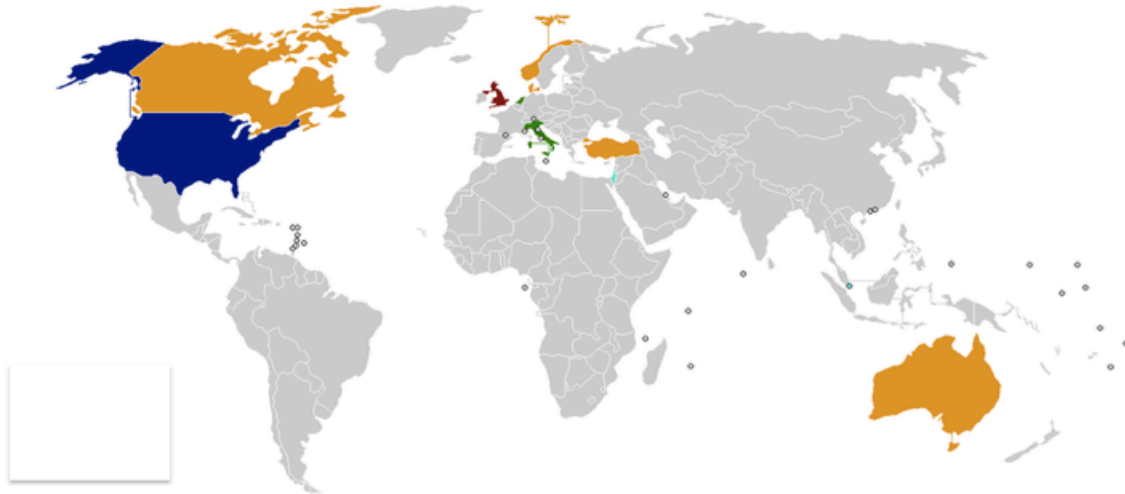
There *was* a Lightning I. It was built by Lockheed for World War II; you can see from its configuration that it was a real hot-rod. It was the only fighter plane design in production during the full duration of World War II; the German air force named it the “Fork-tailed devil.”

The term “joint” in the name “Joint Strike Fighter” refers to the fact that there is a distinct model for each military service:



a conventional takeoff and landing version for the Air Force, a short takeoff/vertical landing version for the Marines, and a carrier-based version for the Navy. Even though the basing requirements are different, there is an unprecedentedly high degree of commonality among the three models and their support systems. This is a very important factor in the life-cycle cost of the total procurement.

There are eight other countries participating in the development that are also committed to purchase planes.



Herein begins the trouble. Regarding this category of weapon, all three US services and much of the western world are putting their eggs in this basket. The F-35 is intended eventually to replace ten aircraft in the US inventory. Largely because of its complexity, the program is slipping both in schedule and cost, causing its customers to scramble to maintain their military effectiveness by upgrading their current weapons. This delay, of course, introduces an unanticipated cost.

The most recent Government Accountability Office (you will see it in the readings I'll be giving you as "GAO") report contains this chart, which shows increases in development cost as the program has progressed. (There are two charts because the cost of the engine development is figured separately.)

Figure 1: JSF Aircraft Development Contract Changes

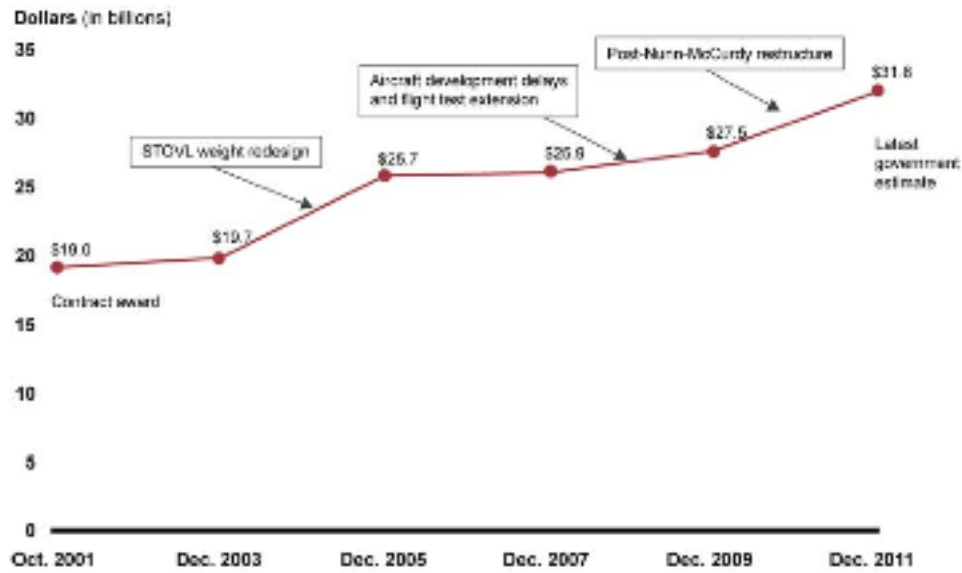
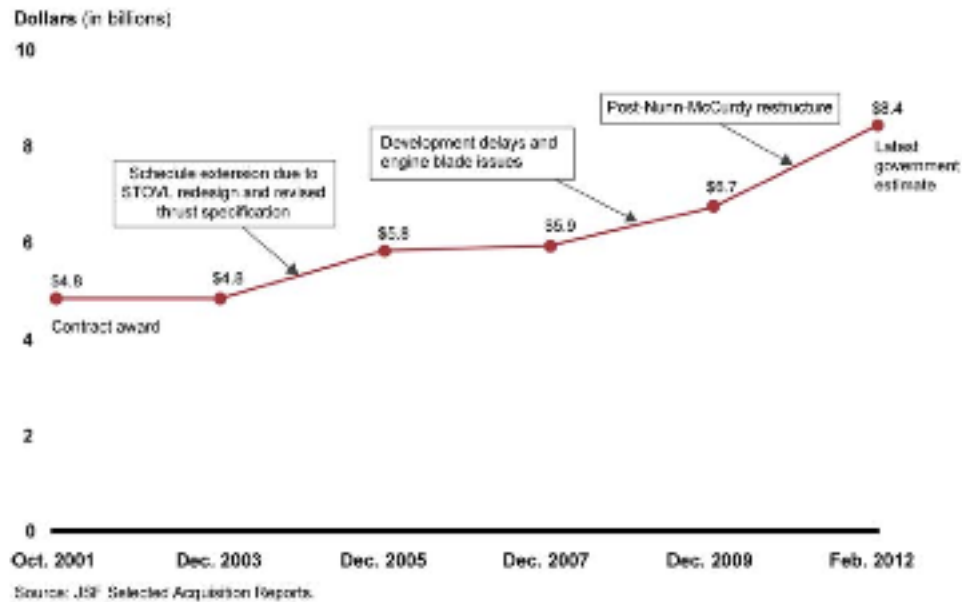


Figure 2: Primary Engine Development Contract Changes





The Pentagon's experience with the F-35 looks like an application of Brooks's Second System Effect. Remember, that's when you try to put in all the goodies you had to leave out the last time. The only other plane in the US inventory like the F-35, and its immediate predecessor, the F-22 Raptor, which entered service in 2005, was also built by Lockheed Martin. (The reason they look alike has more to do with designing to minimize radar reflections than airplane fashion.)

Lockheed threw every bit of advanced technology into the F-35 it could to meet the Pentagon's requirements. As a result of this effort, the F-35 is planned to be an amazing collection of technologies in an unprecedented combination.

First, the F-35 is a software-defined aircraft. The on-board electronics is projected to contain about 9 million lines of source code, vs. the F-35's immediate predecessor, the F-22, which has 2 million lines of code. The total software in the F-35 system, both on-board and ground, will be closer to 24 million lines of code. Just about every mechanical, electrical, and hydraulic subsystem in the F-35 is software-sensed and controlled. The estimate of on-board code has increased by one-third since the 2005 design review. The ground-based code is largely classified, but it seems to be directed to make sure that all aspects required for a successful mission, for example, personnel, training, fuel, maintenance, and munitions, come together at the right time and place.

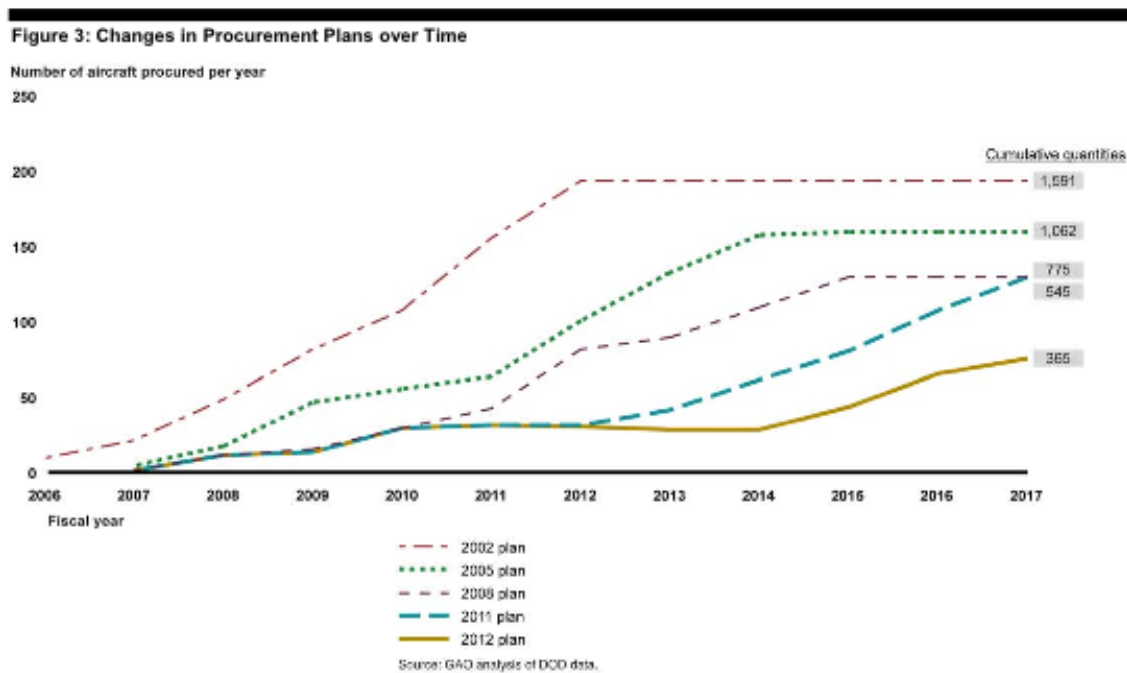
Incidentally, today's automobiles share this problem of a massive amount of on-board code and the bugginess that accompanies it. A few years ago, some Mercedes owners found that when they pushed a certain button on the navigation system the driver's seat moved.

The F-35 is stealthy. This means some combination of being invisible and confusing to radar. The confusion part is called “electronic warfare”. There is a kind of arms race between one side’s radars and the other side’s stealth planes. Making the stealth plane software controlled hastens its designers’ ability to modify it in response to radar changes.

Now it gets interesting. The F-35 has what is being called “total situational awareness”. This means that the pilot can see the total sphere around the airplane; there are no blind spots. Built into the exterior of the plane are six infrared detectors whose outputs are integrated into a single spherical image of the plane’s environment, thanks to a lot of onboard signal processing. This is another source of technological vulnerability, since the six inputs to the signal processors are by no means consistent with each other. Making this all work is late in the test schedule, still in the future.

Now let's talk about this test schedule. The Pentagon began its contract with Lockheed Martin, containing production and test plans in 2001. At the time, the total number of planes for US purchase was to be 2852, with initial full-rate production in 2012. The planned number of aircraft has since been reduced by a few hundred, but the first year of full-rate production at 200 planes per year has slipped from 2012 to 2019. I'll be giving you more detailed numbers after I finish. In one three-year period, the program slipped three years.

This chart from the latest GAO report shows that, even though the total number of planes to be delivered has barely changed, the deliveries will be slipping significantly. The top line shows initially planned delivery schedules, with production rate maxing out at 200 per year in 2012. The bottom line shows delivery schedules from the most recent replanning. In this latest plan, total deliveries through 2017 will be about one-quarter the number of deliveries projected in the original plan. The current plan now anticipates deliveries through 2037. *The budget for the whole F-35 program is 1½ Trillion dollars.*



The reason for this slippage is not that the planes can't be built; they are being built, at a current rate of about 30 per year. The problem is called "concurrency."

Concurrency is the overlap of production with design testing. In order for the system to come into service in a useful time frame, design testing is ongoing as production planes are being delivered into the inventory. It was asserted by Lockheed Martin that design techniques were now mature enough that the impact on the production schedule of design changes discovered in design testing would be minimal. This has turned out to be untrue, and a large portion of production planes will need to be retrofitted. The amount of concurrency in this program is unprecedented in the Defense Department.

Send in the Tech Reinforcements

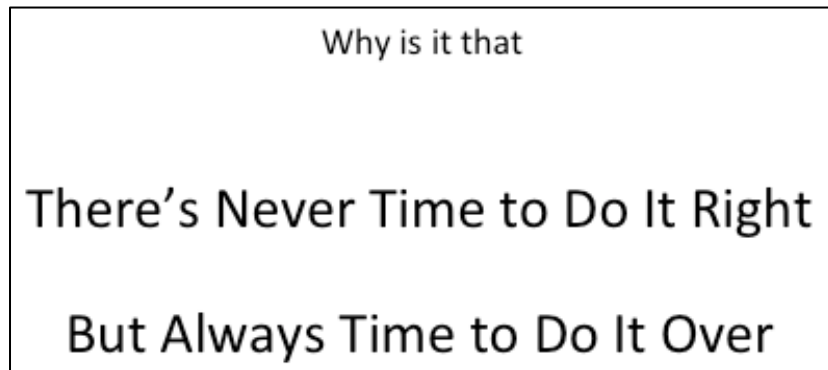
Last February the Wall Street Journal published an article that said that the Pentagon has not learned to build military hardware the way we are learning to build software, with tight feedback and correction between test and design. In my view, this is simplistic. Aircraft designers have a much harder problem than software engineers, because of the cost and schedule impact of errors discovered during test. Some of these errors might lead to, indeed, have led to, for example, cracks in a bulkhead requiring rebuilding of the airframe. The Pentagon's way of doing agile weapons development is called "concurrency", but agile software development is child's play compared to rushing out a high-performance aircraft that is pushing the technology envelope in every possible direction. Each engineering change raises total program cost and lengthens the delivery schedule: the planes will come later, and, because the total program cost must be allocated across all aircraft, each one

will cost more. Because of the general shortage of funds, higher unit cost means there will be order cancellations and fewer planes will be bought. This reduces purchase quantity and again raises the unit cost of each plane.

Concurrency is a big deal because, if it doesn't work as planned, military effectiveness is directly impaired.

Among the official reviews of the program, concurrency is viewed as its number one source of risk.

J. Presper Eckert, the chief engineer for the Univac I in the 1950s, had this sign on his wall. It will be familiar to everyone who must estimate a project under schedule pressure. The Pentagon puts this kind of pressure on its contractors. The consequence is concurrency and slippage.



The tests being executed now are still early in the total test plan, largely still dealing with the F-35's ability to meet its requirements as a high-performance aircraft. In other words, testing of the software in the operating aircraft related to its fighting mission, such as total situational awareness and electronic warfare, is still in the future.

Furthermore, as some early tests fail and result in redesign, the test schedule is rearranged by delaying failed tests until later in the test plan and swapping them with future tests that are easy enough to succeed. That's why the test plan is actually ahead of schedule but one can reasonably expect that it's going to bog down seriously, maybe even hit a wall, later.

The Pentagon has dealt with some failed tests by relaxing the F-35 specifications in minor ways. One wonders how far this relaxation might go when some of the major operational requirements are challenged by failed tests. In other words, we don't know the real schedule yet and the program is already about 6 years behind the original plan. This is creating consternation around the world among several of our allies.

Total situational awareness, a key part of the F-35 promise, is a massive technological challenge. The key to situational awareness is a display that is integrated with the pilot's helmet.



This is an innovation in fighter planes. The signals from the six detectors on the skin of the aircraft are somehow reconciled with each other, and a unified signal is projected onto the face of the helmet. This reconciliation of the external data is called “sensor fusion”. The faceplate of the mask, by the way, is not opaque as the picture suggests.

There's more. It turns out that each flying aircraft is one element of a distributed network containing other air and ground sources as well, and these sources are combined in the pilot's display. *In other words, each F-35 is a moving node of a distributed signal processing system that*

gathers data about its local environment and reconciles these data, and then provides these reconciled local data to the other nodes. As far as I know none of this has yet been tested on a flying aircraft.

This is obviously risky, and the risk is amplified by the fact that the helmet itself is not passing some of its key tests. For example, the update latency of the display is 300 milliseconds, too slow for pinpointing targets at supersonic speeds. An alternate helmet with fewer features is under development, but without the originally planned helmet, the workload of the pilot increases to the point that the airplane's full mission cannot be accomplished. If neither helmet works, the Major General in charge of the Pentagon's F-35 program office has said, "You don't fly this airplane without a helmet." This smells like a single point of vulnerability for the whole program. [Added after delivery of the address: I have learned that the alternative helmet contract was canceled.]

All these technologies are state-of-the-art, but I doubt that they have been made to work together like this. Now

add to this mix of risks the need to feed the outputs of this sensor fusion and distributed signal processing system into targeting, and I suspect we are in new territory.

I am reminded of the first documented example of technological overreach: the Tower of Babel story in Genesis. After the Flood, the overconfident people came together to build a tower that would reach to Heaven. God, seeing that they had not acquired wisdom from the experience of the Flood, scattered them and gave them different languages so they could not cooperate.



Pieter Breugel the Elder

Does the overreach analogy apply to the F-35? I would not sell the combination of the Pentagon and Lockheed Martin short. From their extensive experience with complex military systems they have built an institutional machine for pulling rabbits out of this kind of hat. However, I wonder if Congress would not already have canceled the program if Lockheed Martin did not have subcontractor work going on in 45 states. You will find in the readings I'll be giving you a Bloomberg article that sums this up nicely.

In Design, Complexity is Toxic

I'll close now with a comment based on the lessons I have drawn from my experience and this case study.

The big lesson is that complexity is toxic, leading to design errors and system disintegration.



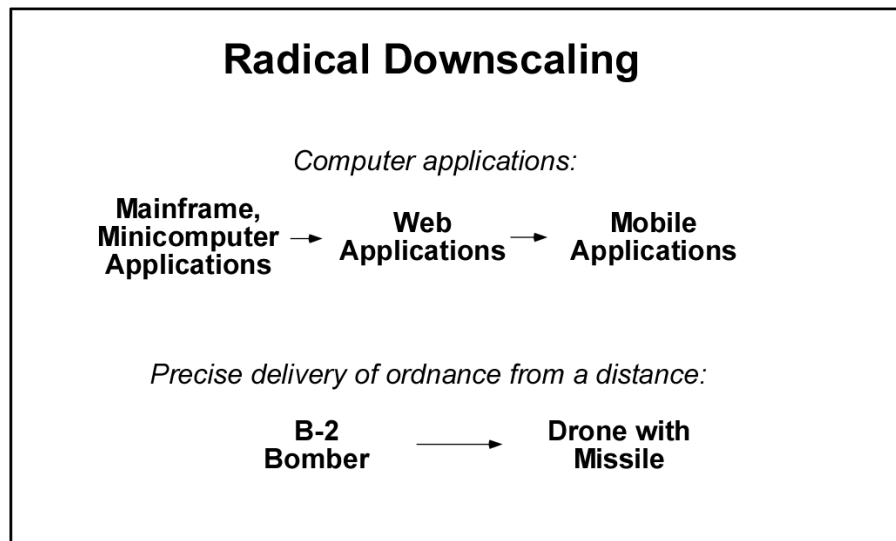
After writing the 1968 paper I took as a personal research goal for the next thirty years thinking about *abstractions that will permit a single person to be able to wrap his mind around a system of any size. I am coming to believe that there is no limit to the power of abstractions we can invent except for the limit of human ingenuity.* Fred Brooks made a reference to this idea when he said, “Representation is the essence of programming.”

“Representation is the essence of Programming”

Fred Brooks

In my 30 years of research I went through the mind-bending process of radically changing my conceptual model of event-driven applications with graphical user interfaces, the applications we use every day with our mobile phones and computers. I was looking for a non-sequential, graphical conceptual model that would permit a school child to understand and build any such application. I pushed my brain away from the input-process-output model of the UNIX shell to the transform-in-place model of the potter's wheel. The epiphany occurred after I spent hours watching a baby try to grasp a grape and put it in his mouth. I can attest to the feeling of triumph after diving into a conceptual morass and coming out the other side with something a little different. This experience has led me to the belief that with enough time and effort we will find more powerful abstractions for large systems. Among the resources I'll be giving you is a brief presentation on the conceptual shifts I had to make and the design principles I adopted.

The F-35 experience suggests to me that our methods for designing large systems have hit a wall. Are complex systems pushing the comprehension limits of our brains?



I have one suggestion: Radical down-scaling of our system models. We are actually moving in this direction, as you see in the examples on the screen, not necessarily as a general approach to simplification but as a set of specific responses to technological opportunities such as miniaturization.

I'll close with this suggestion: let us commit to a new research program with radical down-scaling as a general strategy for designing large systems with less complexity.

Thank you.

http://melconway.com/keynote <i>conwaydotmelatgmaildotcom</i>
