Haste makes Waste: The National Health Care Software Fiasco
Part One

In December of 2008, President Barack Obama declared his administration was pushing for a national effort to automate (digitize) the vast paper files of America's health care system. His commitment to digital medical records was part of an ambitious stimulus package for the U.S. to recover from the near-disastrous financial meltdown.

The effort would supposedly lead to electronic health records (EHR) throughout the nation. The president claimed; "We will make sure that every doctor's office and hospital in this country is using cutting edge technology and electronic medical records so that we can cut red tape, prevent medical mistakes, and help save billions of dollars each year."

Even a hostile Congress (to Obama's tenure) got on board. This body had been led by former House Speaker Newt Gingrich who stated it was easier to track a Fed Ex package than to track one's medical records.

In 2005, I listened to Gingrich speak at the National Press Club in Washington, DC, and later had a brief conversation with him.

Mr. Gingrich's emphasis was the health care system in America. The gist of his presentation and brief conversation with me was that the government programs were dangerously out of control and the continuance of present policies would lead to catastrophic problems for America in the future.

He said, "If we don't change our way of doing things, we will not succeed." He stressed the need to get rid of paper records and to digitize them. (It is likely that Newt did not receive much support from the paper industry lobbyists.)

Congress followed Newt's advice, even though he was no longer in Congress. In 2009, in spite of its disdain for Obama, Congress passed the HITECH act, which allocated billions of dollars for a national EHR initiative. Not millions, billions.

How many billions? Thirty six of them, as in $36,000,000,000. To what result? One can agree or disagree with the answer to this question but the following assertions are taken from comments of doctors about this issue (see the end of this article for the sources of these studies):

The EHRs they used had helped them, but 59% of the doctors stated their EHRs needed a complete overhaul. They also said, "The systems had detracted from their professional satisfaction (54%) as well as their clinical effectiveness (49%)."

One doctor offered, "Everything is so cumbersome. It's slow compared to a paper chart." One of the studies claimed a doctor (or an assistant) had to make several thousand mouse clicks during a shift. Even the most diligent key-boarder is going to make mistakes while doing thousands of entries.

The opponents of the current EHR situation state some entry errors led to severe consequences, such as erroneous treatment of a patient, even the death of a patient (admittedly, very rare).

From the perspective of this writer, a layman in the medical world, but somewhat versed in computer software, it appears Uncle Sam's thirty six billion dollars bought a mixed bag of effectiveness.

Is this result satisfactory? After all, large software systems are complex. They can strain the mental powers of even the most gifted humans. As explained in the next part of

this series, the U.S. government's EHR programs are not satisfactory, regardless of the $36,000,000,000 expenditure. It represents expenditure straight out of the wallets and purses of the taxpayers: you and me.

We shall see how this software debacle came about. As well, this series will examine the Boeing 737-M software disasters and a personal experience this writer had with software that was used to manage our nation's money supply and interest rates.

+++

These articles have used the following sources: khn.ort/ehr, fortune.com/longform/medical records (which cites the Stanford Medicine's 2018 National Physician Poll), *Fortune* magazine (April, 2019), U.S. Department of Health and Human Services.

Haste makes Waste: The National Health Care Software Fiasco
Part Two

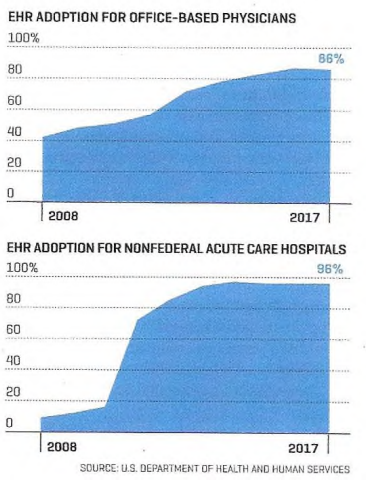
It is not unreasonable for taxpayers to ask Uncle Sam: How could you spend 36 billion dollars of our money on a software system that a huge number of its users do not want to use?

This writer believes the answer to this question is twofold: One, Obama's administration set in place the necessity to implement this software initiative on a time line beyond any semblance of reality. Two, regardless of the time line, the incompetence of its implementation staggers the imagination.

Given this introduction, hold on to your taxpayer ire. Here we go.

The Tail Wags the Dog

Because this enormous infusion of money was tied to the 2008 financial meltdown, it was to be used as an immediate stimulus to the economy. Thus, the adoption of electronic health records (EHR) took place rapidly, as shown in the figure.



Show me the money! It was already being shown. Physicians could "qualify for federal subsidies…[but] only if they [used] a government-certified system." As mentioned, time was important. Because the EHR initiative was part of the nation's strategy to stimulate the economy, the funds had to be distributed as quickly as possible.

As the figure shows, the EHR industry was small potatoes in 2008, valued at about two billion dollars. A thirty six billion dollar infusion into this industry was akin to a sudden gold rush, but only if a company could ramp up its operations to produce and sell its software quickly.

As a consequence of this sudden software god-send, 700 EHR firms sprang into existence! Seven hundred different systems were created that were islands of isolation from communicating with one another. It was as if 700 different languages were created within a few short years.

All well and good, *if* the systems (using the language metaphor) were not to speak with one another. But they should have interworked. After all, that was one of the major reasons for their creation, to be able exchange health care records between different hospitals and physicians.

In an astounding display of ignorance and ineptitude, the head of the organization responsible for managing the federal EHR program offered these ideas: "[Seema Verma, chief of the Centers for Medicare and Medicaid Services] shutters at the thought of the billions of dollars spent building software that doesn't share data---an electronic bridge to nowhere."

She told her interviewers (*Fortune* magazine and KHN analysts): "…we didn't think about how all these systems connect with one another. That was the real missing piece."

Fantastic.

Perhaps Ms.Verma was not in charge of national EHR operations in 2008/09. Regardless of the boss, how could the requirement for having different health care records to communicate with one another be "missing?" With this mentality, it is no wonder the national EHR program has floundered.

A software programmer's dream is to be able to automate a manual system; to create the software from scratch. At the other end of the desirability spectrum is making changes to an existing software system. This effort requires the programmer to learn about the code that was generated by another programmer. The result is often the addition of code that introduces errors or leads to klutzy systems and unfriendly user interfaces.

Granted, in 2008 hospitals and physicians had their own way of creating and maintaining their records. But the national EHR initiative provided a golden opportunity to start with no existing (or little) software to manipulate.

Take another look at the bottom picture in the figure accompanying this article. An opportunity was lost because the managers of this program did not have the "missing piece."

The 36 billion dollar windfall had to be exploited, and exploited soon. Get that software out the door! As one EHR company executive said, "This is a one-time opportunity to expand our share, focus everything there, and then we'll go back and fix it."

Fixing software is not like fixing an automobile carburetor. Fixing faulty software sometimes ends-up making the software even more error-prone.

One of the readers of my work sent these thoughts to me: "So all the government needed to do to make this work was specify at the start that the software had to be compatible to a certain standard. Is that right?" My response: "Yes, and especially to make the software physician-friendly. They did neither."

My reader continued, "I'm certain none of the competing companies wanted their software to be compatible with others." My response: "Why would they want compatibility? Such an action would diminish their potential market share."

The next article in this series examines the EHR industry, with a focus on the major vendors of EHR software.

+++

These articles have used the following sources: khn.ort/ehr, fortune.com/longform/medical records (which cites the Stanford Medicine's 2018 National Physician Poll), *Fortune* magazine (April, 2019), U.S. Department of Health and Human Services.

Haste makes Waste: The National Health Care Software Fiasco
Part Three

Software failures are rare in major commercial systems. But like any human creation, they happen. It's part of the reality of our living on this earth. None of us, including software engineers, are infallible.

But a software failure to a system installed in millions of computers has considerably more consequences than a software failure---as tragic as it may be---to two Boeing airplanes that crashed (discussed in part four of this series).

As stated in part two, about 700 electronic health records (EHR) software companies sprang up after President Obama and Congress allotted 36 billion dollars for the program. Today, the big players in the marketplace are:

EHR in Hospitals:

Epic	12.8 %
Cerner	12.8%
Meditech	12.0%

EHR in Other Installations (for example, primary care physicians):

Epic	23.5%
Allscripts	8.4%
Nextgen	5.9%

Epic Systems leads the pack. *Fortune* magazine calls Epic Systems, "the Cadillac of EHR systems." Nonetheless, the company is not immune from problems and associated law suits. A large company writing complex code, such as Epic, will likely encounter lawsuits. The other major EHR firms have also been sued. Many have settled out of court.

Part of the problem stems from the fact that the original EHRs were designed to support book keeping and billing operations. Less thought was given to patient care and physician interfaces.

As indicated in part two of this series, government officials' lack of oversight of the industry led hundreds of software firms rushing to get their product onto the market. A sophisticated and reliable system, such as Epic, costs millions of dollars and considerable time to design, code and test the software. Logically and naturally, the user of the software is going to pay a hefty fee for its services, usually many millions of dollars.

However, it is an almost laughable situation that a doctor, shortly after Obama's declaration, could go to Walmart Sam's Club or Costco and purchase an off-the-shelf EHR package for $11,925. Why would a physician want such a system? Because, if it were federally approved, the physician was eligible to dine at Uncle Sam's EHR feeding trough. Haste makes waste.

If you think Sam's Club's software will interwork with say, an Epic system, I've a bridge in Brooklyn to sell and you would eagerly buy it. It is almost as good a deal as the Costco health care package. In software systems, it is safe to rely on that old adage, "You usually get what you pay for." (With an exception the $36 billion EHR expenditure.)

Fixing these systems that sprang from near-nowhere has been quite difficult, partly because "EHR vendors often impose contractual 'gag clauses' that discourage buyers from speaking out about safety issues and disastrous software installations."

Out of sight, out of mind. One cannot fix a system if one does not know its faults. I find it disturbing (and I hope you do, too) that taxpayers pony-up 36 billion dollars for EHR yet the users of these systems (namely, physicians) are often forbidden by their hospital or an EHR contract from discussing or writing about their problems.

One last point about electronic health records: If the software is not user friendly, if a physician dreads interacting with it, the doctor should have the right to speak up. If that right is denied, the system will likely continue to numb the senses and motivation of its users.

If a hospital is going to purchase an EHR system, its doctors must be allowed to test it hands-on and be the final arbiter of using it or looking to another vendor.

The next part of this series examines another software problem that affects many people: the software that supposedly pilots airplanes.

Haste makes Waste: The Boeing 737-M Software Failure
Part Four

Imagine the terror experienced by the passengers of the two Boeing 737-MAX airplanes that recently crashed, killing everyone aboard both planes. Imagine the frustrated terror experienced by the occupants in the flight cabin, as they tried unsuccessfully to pull the airplanes' noses up.

Without help from the pilots, and courtesy of faulty software, on October 29, 2018, one of the planes turned itself downward and plunged into the sea. The other plane ploughed into the earth on March 10, 2019.

In the three previous articles to this series, faulty health care software was cited as being responsible for lost productivity and user dissatisfaction. On occasion, the klutzy software created dangerous situations for a patient, but the result was rarely the loss of life. That cannot be said for the software in these airplanes. Three hundred forty six people lost their lives.

Software failure was been identified as the *sole cause* of the crashes. Taking control of the aircraft, the software did not allow the pilots to override the automated programs, to take command of the situation and correct the path of the planes.

I trust you do not think I am writing these articles as a "the sky is falling" fear monger. However, it must be said that we humans are becoming more and more dependent on software.

As for the health care and aircraft industries, we expect software to take care of our lives. It is not unreasonable to ask software managers and their programmers to produce code that does not place our lives in jeopardy for using it.

I know just enough about software programming to be ill-at-ease in understanding the brakes on my car are controlled by two parties: me and some unknown programmers, who are constantly working under deadlines to get their code "up-and-running." I also try not to feel uneasy when I am flying on commercial planes, knowing the ailerons on the airplane's wings may be beyond the control of the pilot. I still drive and I still fly, as I have no choice.

In the meantime, all airlines around the world have grounded their 737-M airplanes. With the absence of these aircraft in their inventory, they are cancelling flights, suffering significant revenue losses and announcing an increase in ticket prices.

What to do to fix the problem? Write more software and patch it onto the ailing system. We learn from the news media: "Boeing crews have made 96 flights to test a software update for its troubled 737 MAX jet. More test flights are planned in the coming weeks as the company attempts to convince regulators that the plane is safe."

Boeing crews and later, the crews of scores of airlines, made thousands of flights using the faulty software. Everything went along just fine…until it didn't.

Recently, Southwest airlines, the biggest buyer of Boeing planes, has announced it is extending its grounding of the 737-M planes because the software to fix the software is still under test.

Tremendous pressure is being placed on Boeing, from its executives down the chain of command to its programmers. The company has to make sure the fixing of the software is fulfilling its job description, that of repairing flawed software.

According to an AP press release (See *CDA Press*, April 14, 2019, page M6): "Other airlines are likely to follow Southwest's example, *putting pressure* [my italics] on Boeing to finish fixing software on an anti-stall system implicated in two deadly crashes."

Let's hope their haste does not result in the waste of yet more lives. The situation does not make me eager to fly again. My fear is not that the sky is falling, but my airplane might be. (I cannot avoid thinking: Why not let a human actually fly the plane? It has worked just fine for many decades.)

The next article in this series recounts this writer's experience with a software system used to manage the money supply and interest rates of America. This fifth article also offers some ideas that might compensate for the bad news portioned out in the first four pieces of this series.

Nonetheless, go to this website to view the problems associated with the Boeing 737. Your habit of remaining sober on your next airplane flight might take a turn-around. https://en.wikipedia.org/wiki/List_of_accidents_and_incidents_involving_the_Boeing_737

Haste makes Waste: If it Ain't Broke, Don't Fix it!
Part Five

In 1972, I was a rookie software programmer at the Federal Reserve Board in Washington, DC. I was hired by the Board because of my experience with writing simulation software. While in the Navy, I was on a team that coded a package simulating warfare between Chinese submarines and U.S. Navy warships. We team members joked that the coding was an easy job as the Chinese had no submarines.

Anyway, my first project for the Board was to design and write code that simulated the money supply of the nation. At that time, the Board used scores of clerks and economists to crunch numbers (on desk top size calculators) for the Federal Open Market Committee (FOMC) to determine the money supply and interest rates for America.

My job was to automate these manual calculations. So off I went to confer with some brilliant economists and shortly thereafter, to write the code to ease their tasks. It was a dream job: No software that I had to patch into; no other programmers involved. In hindsight, it was a privilege to have written this software.

The software worked. Everyone was happy, and I migrated to other projects. My software was taken over by programmers in the economists' department.

In the fall of 1973 several Middle East nations imposed an oil embargo on its petroleum hungry customers, including the United States. Queues at gas stations formed. Gasoline prices skyrocketed. The FOMC and the Board economists worked into the nights coping with a financial crisis in America.

The software I wrote had to be modified to account for this situation---and fast. I was no longer directly involved in this software, but I was consulted on the changes the economists needed to have made---and soon.

I offered that the software was written as a stand-alone package, but the current crisis seemingly necessitated using parts of it anyway.

The changes were made in a few days, in time for the next FOMC meeting. As recounted in previous reports, the software worked just fine…until it didn't.

The stand alone software I built did not interface well with the new software patches. However, the error did not manifest itself for a while---just like the Boeing 737-M software. A combination of factors that rarely occurred…occurred.

The Board economists had to resort to the original code and amplify the output of this system by resorting to their desk top calculators. They were not happy users.

The phenomenon described in these articles is known in the programming software world as Black Swan Software. Rarely does a black swan swim by; so rare, it is not even coded in the software. But when the black swan does swim by, it might have sad, even devastating consequences (the EHR and Boeing software respectively).

What to do? I've made the case that for creating software, haste makes waste. Does that mean that software programmers should be the tail that wags the dog? That they dictate to President Obama and the CEO of Boeing airlines? Of course not, we all must work with deadlines.

Software folks should be subject to a deadline for completing their coding, but their opinion about this deadline should be taken into account. Some latitude is needed in setting these deadlines.

As an example, suppose you decide to write a biography about your life. The publisher establishes three months for you to complete the writing of the manuscript. You ask for a year. Nope, the publisher needs to get so many books published in the upcoming quarter. Unless you write a sloppy book, the deadline is impossible to meet.

This example is not too far off the mark in what happened with the national health care system. Let us hope the software programmers at Boeing are given some breathing room for making their patches to the 737-M software. As for the Federal Reserve software, I understand it is stable, functioning, and saving the policy makers from a tiresome session with their calculators.