Risks of Computers: What do we Do?

We Have Problems. . .

- Software is buggy
- It takes too long to develop
- It's generally over budget
- What do we do?

Custom Systems

- Even the big vendors of mass market software have such problems
- A lot of organizations rely on custom systems
- Custom system designs are frequently worse

Conversions

- Different programs
- Different file formats
- Different processes
- Different things to back up
- Different inputs and outputs
- Repeat for each program in your system

We're Dealing with *Systems*

- The hard part isn't one program on one machine
- Generally, it's many different programs
- They all interact
- "You can't eat just one peanut"

A Tale of Two Airlines. . .

- WestJet and JetBlue wanted to switch to new, more capable reservations systems
- They independently selected the same new system
- WestJet went first
- “Despite months of planning . . . its Web site crashed repeatedly and its call center was overwhelmed. It took months to resolve all the issues.” (WSJ, 12 Apr 2010)

Why?

- Massive data conversion: 840,000 files for existing reservations
- Too much complex hand-processing to convert files
- Too many passengers with reservations, despite the preemptive cancelation of some flights
- For competitive reasons, they didn't announce the conversion until that day

Recovery

- Apology letters
- Flight credits
- Outsourced temporary call center
- A “three- to six-month recovery process”

JetBlue Went Second

- They picked a light-traffic weekend
- They kept their planes abnormally empty
- They developed and deployed a backup web server (it was needed twice)
- They hired 500 temporary agents to handle routine calls, while their own, experienced staff handled complex situations
- The extra agents stayed for two months — “one of the wisest investments we made”
- There were still a few problems

The Differences

- Lighter load (though JetBlue is a bigger airline)
- Backup systems
- Personnel who were trained for conversion glitches

What Wasn't a Problem

- For the most part, this did not involve new software development
- They were switching to an existing, well-tested product
- (There was undoubtedly some custom software written, both for the conversion and for customization)
- JetBlue spent $40M — $25M in capital costs and $15M in one-time operating expenses
- The problem was *conversion*

Conversion Principles

- Conversions never go smoothly
- If possible, run old and new systems in parallel for a while
- Lighten load
- Automate as much of the data conversion as possible
- Train people for the effort
- Temporarily shed non-essential features

Why is the a Societal Issue?

- When large-scale — or governmental — systems are converted, the public at large is converted
- It costs money and poor service – and we all pay
- Too often, the conversions are postponed or mismanaged

The FBI

- The FBI's computer systems were *old*
- As of 2002, their network ran "bisync", a protocol that was obsolete by 1980
- (They had to buy spare parts on eBay!)
- The Trilogy project, the attempt to upgrade, was an unmitigated disaster

What the FBI Did Wrong

- No user involvement — they didn't know what they wanted the system to do
- Requirements change — post-9/11, they realized they needed much more support for intelligence work
- No prototyping
- Inadequate IT management ability
- Plans for a flash cut

The IRS Has Problems, Too

- Some of their systems are more than 40 years old, and are written in IBM mainframe assembler language
- They spent $3 *billion* trying — and failing — to upgrade their systems in the 1990s
- The problem: “inadequate management, ill-defined goals, repeated cost overruns, and failure to meet deadlines and expectations.” (CNET News)

Requirements for Successful Software

- Clear goals
- A clean architecture
- Very good management
- Enough time
- A deployment plan

Goals

- Meet the real needs of the organization
- Establish the goals up front, and don't keep changing them
- Have the political fight up front, before you spend money on software

Architecure

- All that stuff you've learned about software engineering really helps
- Remember that needs — and hence software — will change over time
- Clean system designs will smooth that process, too

Deployment Plans

- *Assume* that things will go wrong
- Plan for this — how will you cope?
- If you're running old and new systems in parallel, what if they disagree?
- How do you share data between different versions?

Process Matters

- People are part of the system, too
- In understanding the functions and security of a system, it is important to understand what people do
- People are also much more flexible in coping with mistakes — *if* the process lets them
- In some sense, “process” is the way you program the people. . .

Counting Votes

Photo by Dave Kopel; used by permission. (`http://volokh.com/2008/03/22/taiwan-presidential-election-results-and-process/`)

Why Show an Empty Ballot Box?

- Show that all of the votes have been counted
- (Do the same before the polls open)
- Many more aspects of process during voting

Voting Systems

- DRE machines illustrate many of these problems
- Counties generally don't have enough IT management expertise
- There's no manual backup
- Conflicting goals

The Really Scary Thing about DRE

- The big problem *isn't* security — it's ordinary buggy code
- We've already had many instances of this — and we'll never really know how the election should have come out

Programming Bugs and Elections

- Sarasota, FL, Congressional race, 2006 — large undervote in a heavily Democratic precinct
- Bernalillo County, NM, 2000 — bug tallying straight-ticket absentee ballots
- Voting machine tally tapes, NJ (several elections)
- Many more. . .
- And there's no manual backup!

Large Software Systems

- Be afraid. Be very afraid.
- It *never* works well at first
- But there's "bad" and "worse"
- Expect this and plan for it