RECOVERY-ORIENTED COMPUTING

Recovery Oriented Computing (ROC)

Dave Patterson and a cast of 1000s:

Aaron Brown, Pete Broadwell, George Candea†, Mike Chen, James Cutler†, Prof. Armando Fox†, Emre Kıcıman†, David Oppenheimer, and Jonathan Traupman

U.C. Berkeley, †Stanford University

April 2003

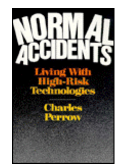
Outline

- **The past:** where we have been
- **The present:** new realities and challenges
- **A future:** Recovery-Oriented Computing (ROC)
- **ROC techniques and principles**

The past: research goals and assumptions of last 20 years

- ***Goal #1: Improve performance***
- ***Goal #2: Improve performance***
- ***Goal #3: Improve cost-performance***
- ***Simplifying Assumptions***
  - Humans are perfect (they don't make mistakes during installation, wiring, upgrade, maintenance or repair)
  - Software will eventually be bug free (Hire better programmers!)
  - Hardware MTBF is already very large (~100 years between failures), and will continue to increase
  - Maintenance costs irrelevant vs. Purchase price (maintenance a function of price, so cheaper helps)

Learning from other fields: disasters

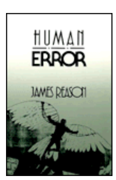


Common threads in accidents ~3 Mile Island

1. **More multiple failures than you believe possible, because latent errors accumulate**
2. **Operators cannot fully understand system because errors in implementation, measurement system, warning systems. Also complex, hard to predict interactions**
3. **Tendency to blame operators afterwards (60-80%), but they must operate with missing, wrong information**
4. **The systems are never all working fully properly: bad warning lights, sensors out, things in repair**
5. **Emergency Systems are often flawed. At 3 Mile Island, 2 valves in wrong position; parts of a redundant system used only in an emergency. Facility running under normal operation masks errors in error handling**

Source: Charles Perrow, *Normal Accidents: Living with High Risk Technologies*, Perseus Books, 1990

Learning from other fields: human error

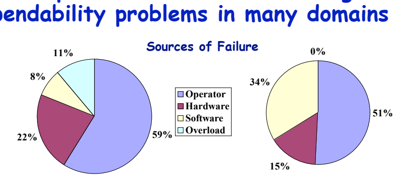


- **Two kinds of human error**
  - **1) slips/lapses:** errors in execution
  - **2) mistakes:** errors in planning
  - errors can be **active** (operator error) or **latent** (design error, management error)
- **Human errors are inevitable**
  - "humans are furious pattern-matchers"
    - » sometimes the match is wrong
  - cognitive strain leads brain to think up least-effort solutions first, even if wrong
- **Humans can self-detect errors**
  - about 75% of errors are immediately detected

Source: J. Reason, Human Error, Cambridge, 1990.

Human error

- **Human operator error is the leading cause of dependability problems in many domains**



- **Operator error cannot be eliminated**
  - humans inevitably make mistakes: "to err is human"
  - *automation irony* tells us we can't eliminate the human

Source: D. Patterson et al. Recovery Oriented Computing (ROC): Motivation, Definition, Techniques, and Case Studies, UC Berkeley Technical Report UCB//CSD-02-1175, March 2002.

The ironies of automation

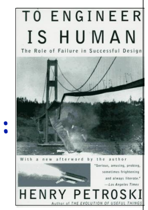
- **Automation doesn't remove human influence**
  - shifts the burden from operator to designer
    - » designers are human too, and make mistakes
    - » unless designer is perfect, human operator still needed
- **Automation can make operator's job harder**
  - reduces operator's understanding of the system
    - » automation increases complexity, decreases visibility
    - » no opportunity to learn without day-to-day interaction
  - uninformed operator still has to solve exceptional scenarios missed by (imperfect) designers
    - » exceptional situations are already the most error-prone
- **Need tools to help, not replace, operator**

Source: J. Reason, Human Error, Cambridge University Press, 1990.

Slide 7

Learning from others: Bridges

- **1800s: 1/4 iron truss railroad bridges failed!**
- **Safety is now part of Civil Engineering DNA**
- **Techniques invented since 1800s:**
  - **Learn from failures vs. successes**
  - **Redundancy to survive some failures**
  - **Margin of safety 3X-6X vs. calculated load**
  - **(CS&E version of safety margin?)**
- **What will people of future think of our computers?**



Slide 8

Where we are today

- **MAD TV, "Antiques Roadshow, 3005 AD"**

VALTREX:
"Ah ha. You paid 7 million Rubex too much. My suggestion: beam it directly into the disposal cube. These pieces of crap crashed and froze so frequently that people became violent!
Hargh!"

"Worthless Piece of Crap: 0 Rubex"

Slide 9

Recovery-Oriented Computing Philosophy

"If a problem has no solution, it may not be a problem, but a fact, not to be solved, but to be coped with over time"
— Shimon Peres ("Peres's Law")

- **People/HW/SW failures are facts, not problems**
- **Recovery/repair is how we cope with them**
- **Improving recovery/repair improves availability**
  - UnAvailability = $\frac{MTTR}{MTTF}$ *(assuming MTTR much less than MTTF)*
  - 1/10th MTTR just as valuable as 10X MTBF
- **ROC also helps with maintenance/TCO**
  - since major Sys Admin job is recovery after failure
- **Since TCO is 5-10X HW/SW \$, if necessary spend disk/DRAM/CPU resources for recovery**

Slide 10

ROC Summary

- **21st Century Research challenge is Synergy with Humanity, Dependability, Security/Privacy**
- **2002: Peres's Law greater than Moore's Law?**
  - Must cope with fact that people, SW, HW fail
  - Industry may soon compete on recovery time v. SPEC
- **Recovery Oriented Computing is one path for operator synergy, dependability for servers**
  - Failure data collection + Benchmarks to evaluate
  - Partitioning, Redundandy, Diagnosis, Partial Restart, Input/Fault Insertion, Undo, Margin of Safety
- **Significantly reducing MTTR (people/SW/HW) => better Dependability & Cost of Ownership**

Slide 11

Interested in ROCing?

- **More research opportunities than 2 university projects can cover. Many could help with:**
  - Failure data collection, analysis, and publication
  - Create/Run Recovery benchmarks: compare (by vendor) databases, files systems, routers, …
  - Invent, evaluate techniques to reduce MTTR and TCO in computation, storage, and network systems
  - (Lots of low hanging fruit)

"If it's important,
how can you say it's impossible if you don't try?"
Jean Monnet, a founder of European Union

http://ROC.cs.berkeley.edu

Slide 12