DEPENDABLE SOFTWARE: AN OXYMORON?

Daniel Jackson · RE'05 · Paris, September 2005

CSAIL
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

introduction: our study

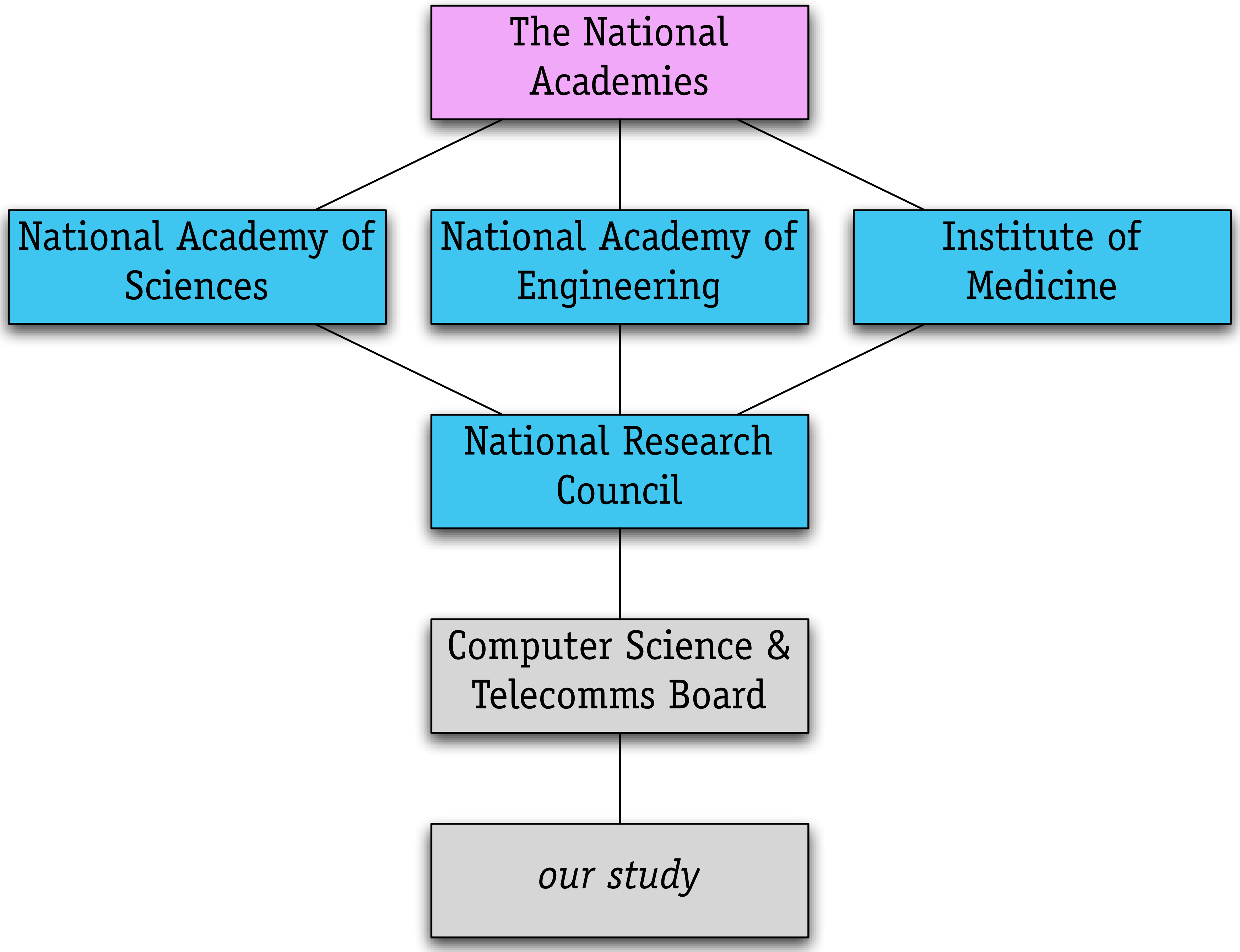
the study

Sufficient Evidence? Building Certifiably Dependable Systems

a study of the National Academies

› chartered by Act of Congress in 1863, signed by Lincoln
› gives advice to federal government and to the public

the national academies



The National Academies

National Academy of Sciences

National Academy of Engineering

Institute of Medicine

National Research Council

Computer Science & Telecomms Board

our study

committee

Joshua Bloch, Google
Michael DeWalt, Certification Systems, Inc.
Reed Gardner, University of Utah School of Medicine
Daniel Jackson, Massachusetts Institute of Technology, Chair
Peter Lee, Carnegie Mellon University
Steven Lipner, Microsoft Security Business and Technology Unit
Charles Perrow, Yale University
Jon Pincus, Microsoft Research
John Rushby, SRI International
Lui Sha, University of Illinois at Urbana-Champaign
Martyn Thomas, Martyn Thomas Associates
Scott Wallsten, American Enterprise Institute/Brookings Joint Center
David Woods, Ohio State University

Lynette Millett, CSTB, Study Director

sponsors & mandate

sponsors

- National Science Foundation
- National Security Agency
- Office of Naval Research
- Federal Aviation Administration
- informally: FDA, NASA, DARPA, AFRL too

mandate (very roughly)

how can we make software dependable?
in a timely and cost-effective way?
does certification help?

progress

schedule

- started in December 2003
- 5 three-day meetings so far
- public workshop in April 2004
- report expected early 2006

what we do

- open sessions: hear evidence & advice from experts
- closed sessions: think & argue
- homework: writing & researching

for workshop report, see:
http://www7.nationalacademies.org/cstb/pub_dependable_workshop.html

what is dependable software?

software that can justifiably be depended upon, in safety- and mission-critical settings

main concern:
prevent catastrophes

what this talk is (and isn’t)

some comments based on open materials

› workshop presentations

› evidence given in open sessions

› papers we’ve been referred to

some personal opinion

› not necessarily those of the committee!

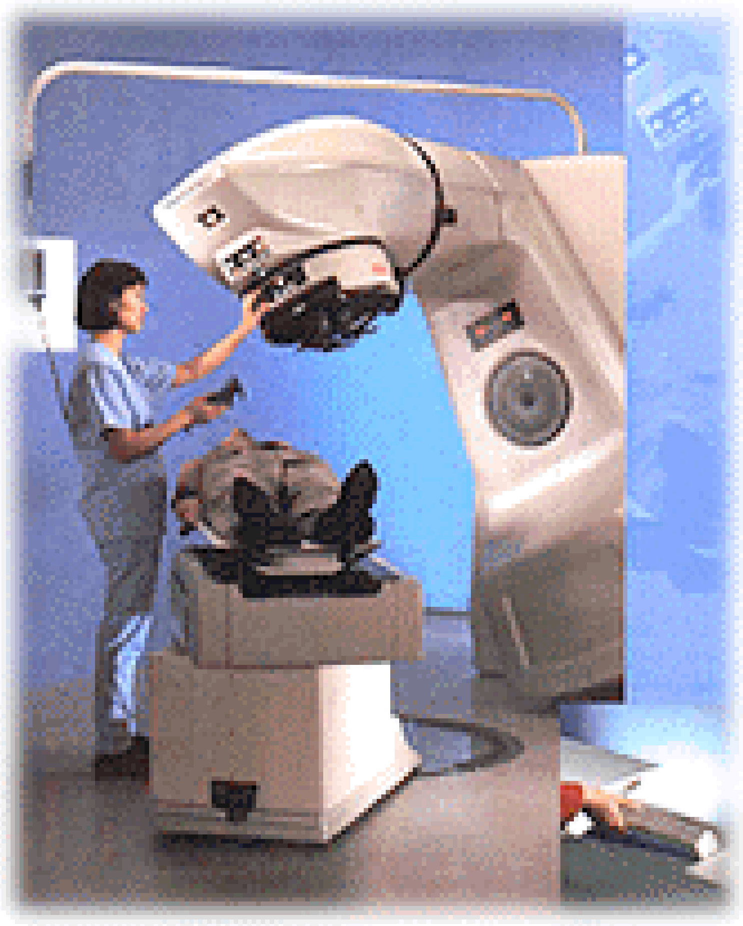
for our diagnosis & recommendations, see final report!

two medical catastrophes

panama radiation accident

Panama city public hospital, 2001

- therapy planning system by Multidata Systems
- Theratron-780 by Theratronics (maker of Therac-25)



shielding blocks

shielding blocks

- inserted into path of beam
- to protect healthy tissue

what software does

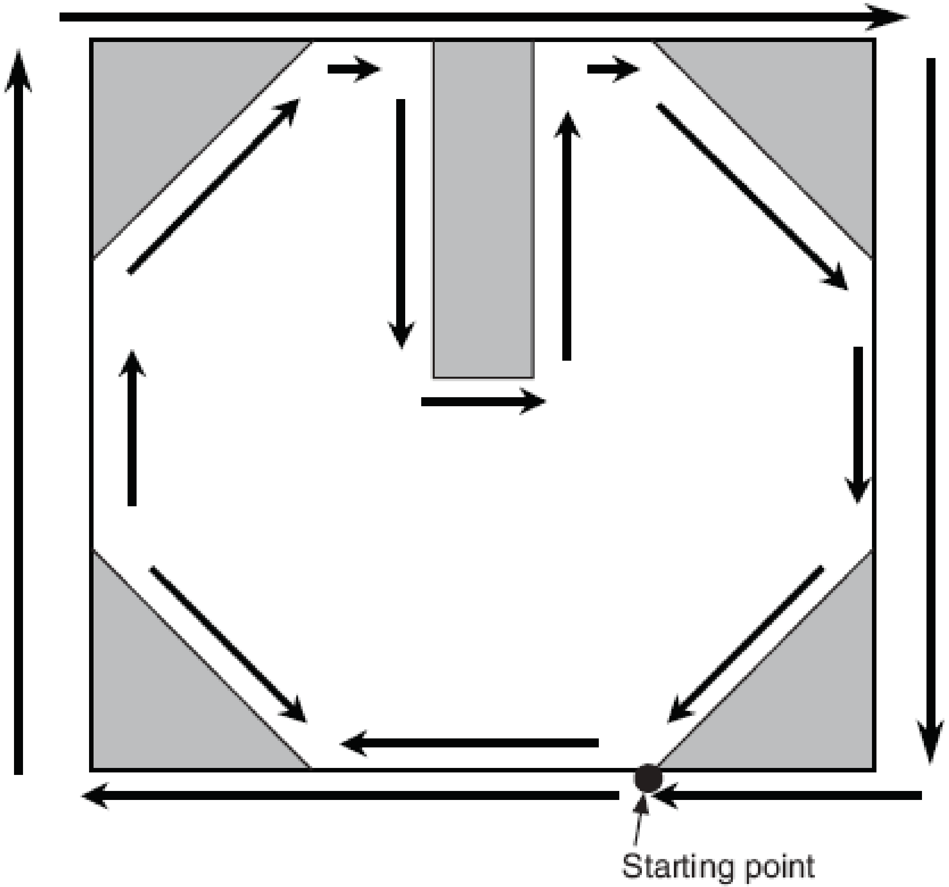
- shielding reduces scatter from nearby tissue, so must increase intensity to accommodate
- therapist draws blocks on GUI with mouse, and system computes new settings

a snag

- Multidata's system only allowed 4 blocks

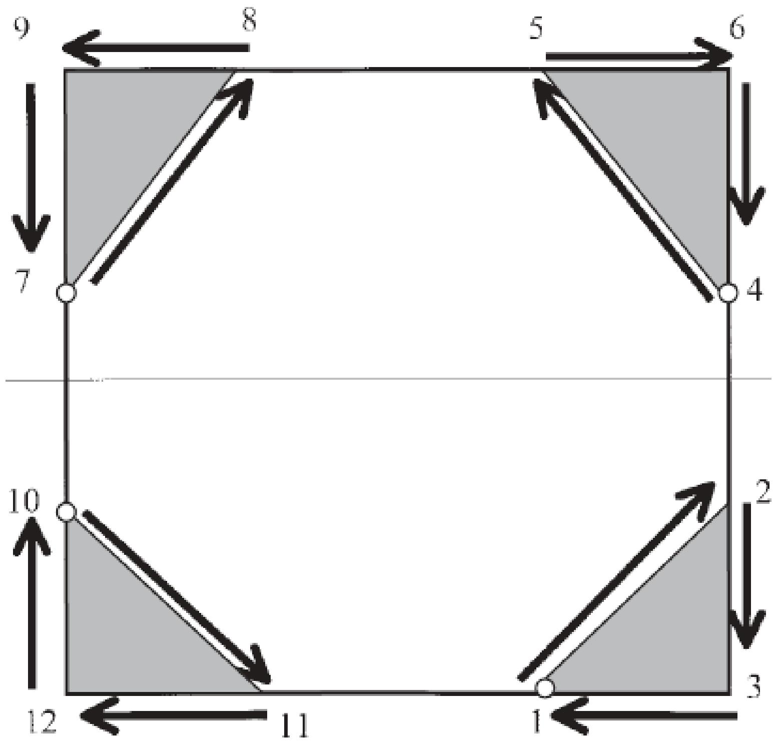
a workaround

draw multiple blocks as a single block

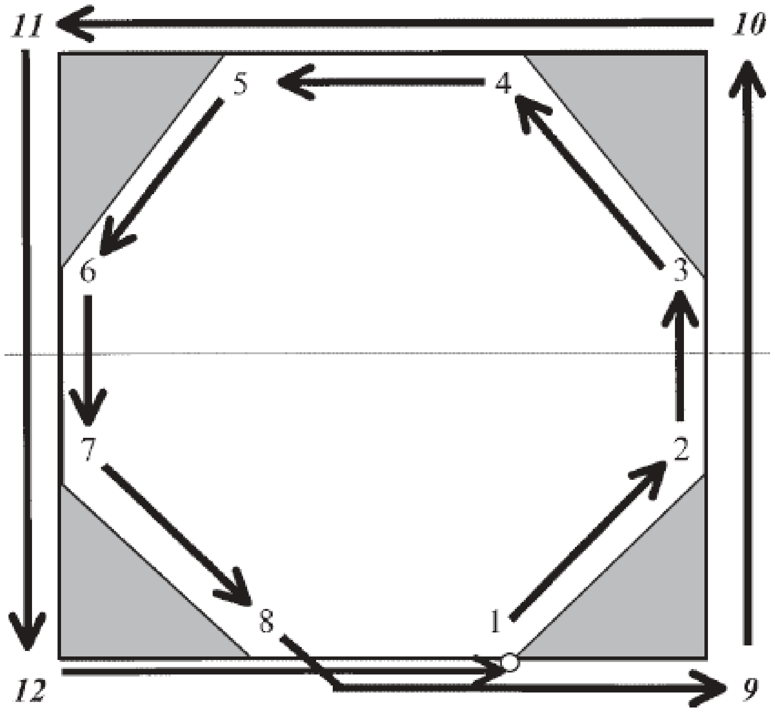


diagrams on this and subsequent slide from:
Investigation Of An Accidental Exposure Of Radiotherapy Patients In Panama.
Report Of A Team Of Experts, 26 May –1 June 2001.
International Atomic Energy Agency

but how drawn matters



dose = D



dose = $2D$

consequences

28 patients were overdosed over a period of several months

in May 2001, government calls in IAEA under
‘Convention on Assistance in the
Case of a Nuclear Accident or Radiological Emergency’

of 20 surviving patients

› 3/4 expected to suffer serious or fatal injury

MAR knockout

setting: a large, tertiary-care hospital

events: Friday night

- › nurse notices medication delivered to ward hasn't been ordered
- › but matched newly printed medicine admin record (MAR)
- › calls pharmacy; they check computer, all looks OK
- › more calls come in: problem is hospital-wide

Saturday morning:

- › crisis: pharmacy fill list is wrong, MARs can't be trusted
- › all medicines sent back to pharmacy

Richard Cook and Michael O'Connor
Thinking About Accidents And Systems
In K. Thompson, H. Manasse, eds. Improving Medication Safety Washington DC.
ASHP, to appear.

how the story ended

Saturday morning: mitigation

- senior pharmacist takes charge
- has each ward fax yesterday's MAR's to pharmacy
- filled gaps from handwritten physician records
- pharmacy fills prescriptions from these manual lists

by end of day

- database updated, system back online
- nobody injured

what happened?

sometime before

- devastating chemotherapy accident
- pharmacy software modified to check doses

Thursday night

- pharmacy software reported database integrity failure
- software engineer attempted fix, but failed
- so database was reloaded from most recent backup
- all appeared to work fine

Friday

- but backup tape was incomplete
- reloading corrupted database

some myths & misconceptions

myth: it’s the operator’s fault

Given [the input] that was given, our system calculated the correct amount, the correct dose. It was an unexpected result. And, if [the staff in Panama] had checked, they would have found an unexpected result. *-- Mick Conley, Multidata**

3 Panama physicists on trial for second-degree murder

- Saldaña paying for her own defence; earns $585/month

easy to blame the operator, especially when dead

- hindsight bias: accident was likely and anticipatable
- human behaviour always variable
- accidents rarely have a simple cause
- organizational & technical factors combine

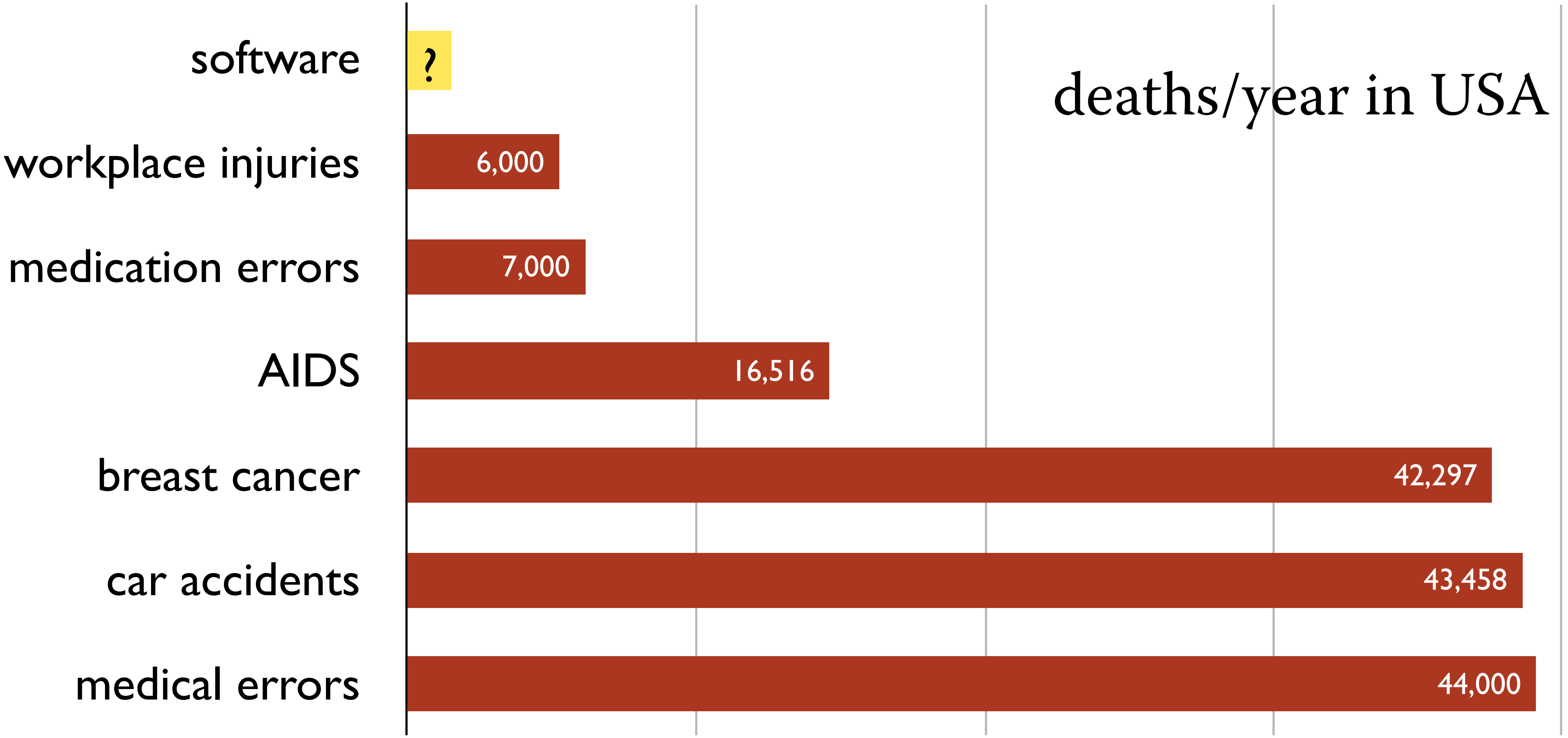
*Deborah Gage and John McCormick
Can Software Kill?
http://www.eweek.com/article2/0,1895,1544225,00.asp

myth: software kills many

some infamous accidents

- 1985: Therac-25; >3 killed
- 1992: London Ambulance System; 20 died, some allege
- 1995: American Airlines jet hit mountain; 159 killed
- 1997: Korean jet crash in Guam; 225 killed
- 2001: Panama radiation accident: 20 killed

what kills people?



from:
Linda T. Kohn, Janet M. Corrigan, and Molla S. Donaldson, Editors
To Err Is Human: Building a Safer Health System
Committee on Quality of Health Care in America, Institute of Medicine, 2000
http://www.nap.edu/openbook/0309068371/html/1.html

so does software matter?

software is a growing threat

‘accidents are signals sent from deep within the system about the vulnerability and potential for disaster that lie within’

software reduces the number of accidents …

› can eliminate many medical errors

… but increases their severity

› MAR more worrying than Panama?

Richard Cook and Michael O’Connor
Thinking About Accidents And Systems
In K. Thompson, H. Manasse, eds. Improving Medication Safety Washington DC.
ASHP, to appear.

risky trends

software encourages

- complexity: no barriers
- coupling: integration

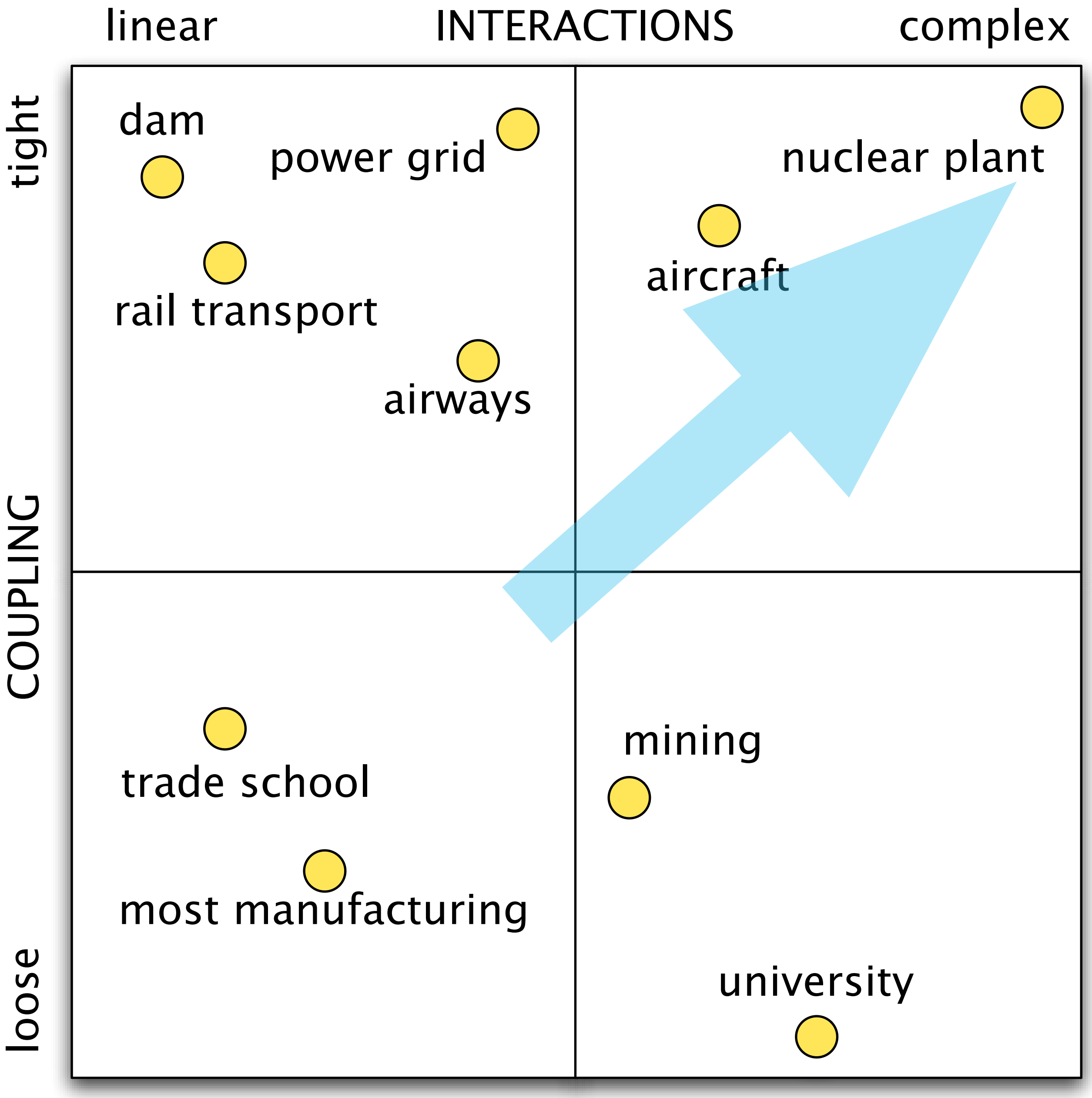
automation

- eliminates human oversight
- makes mitigation harder

examples

- TSAFE: can air-traffic controller handle conflict in 10 seconds?
- MAR: what if patients & drug records had been integrated?

where software is headed?



from: Charles Perrow; *Normal Accidents: Living with High-Risk Technologies*; Basic Books, NY, 1984.

tip of the iceberg?

GAO report (1986)

› sample of 1175 medical device failures

› 9% caused injury, 37% potential for death or serious injury

› less than 1% reported to FDA

Medical Devices: Early Warnings of Problems Is Hampered By Severe Underreporting
General Accounting Office, Washington DC, 1986
GAO/T-PEMD-87-1
http://archive.gao.gov/t2pbat22/132822.pdf

so, in short …

yes, software matters

- few disasters so far, but underreported
- software can save lives, but increases risk of catastrophe

myth: most software not critical

‘accidental systems’ [**John Rushby**]

› enterprises introduce software
without realizing they form a system

commercial COTS components in critical systems

› a time-bomb?

examples

› MAR knockout
pharmacy database was critical
FDA regards as ‘medical device’, but not subject to 510k

› viruses in radiotherapy machines
2 incidents this summer: in Merseyside & Boston

what the FDA says

Databases, software, and computers that only process manually input data and is not used in the diagnosis, treatment, etc. of the patient is usually exempt from 510(k) requirements. However, it is a medical device and should be properly labeled and meet its claims. Examples include; patient data management programs or record keeping programs.

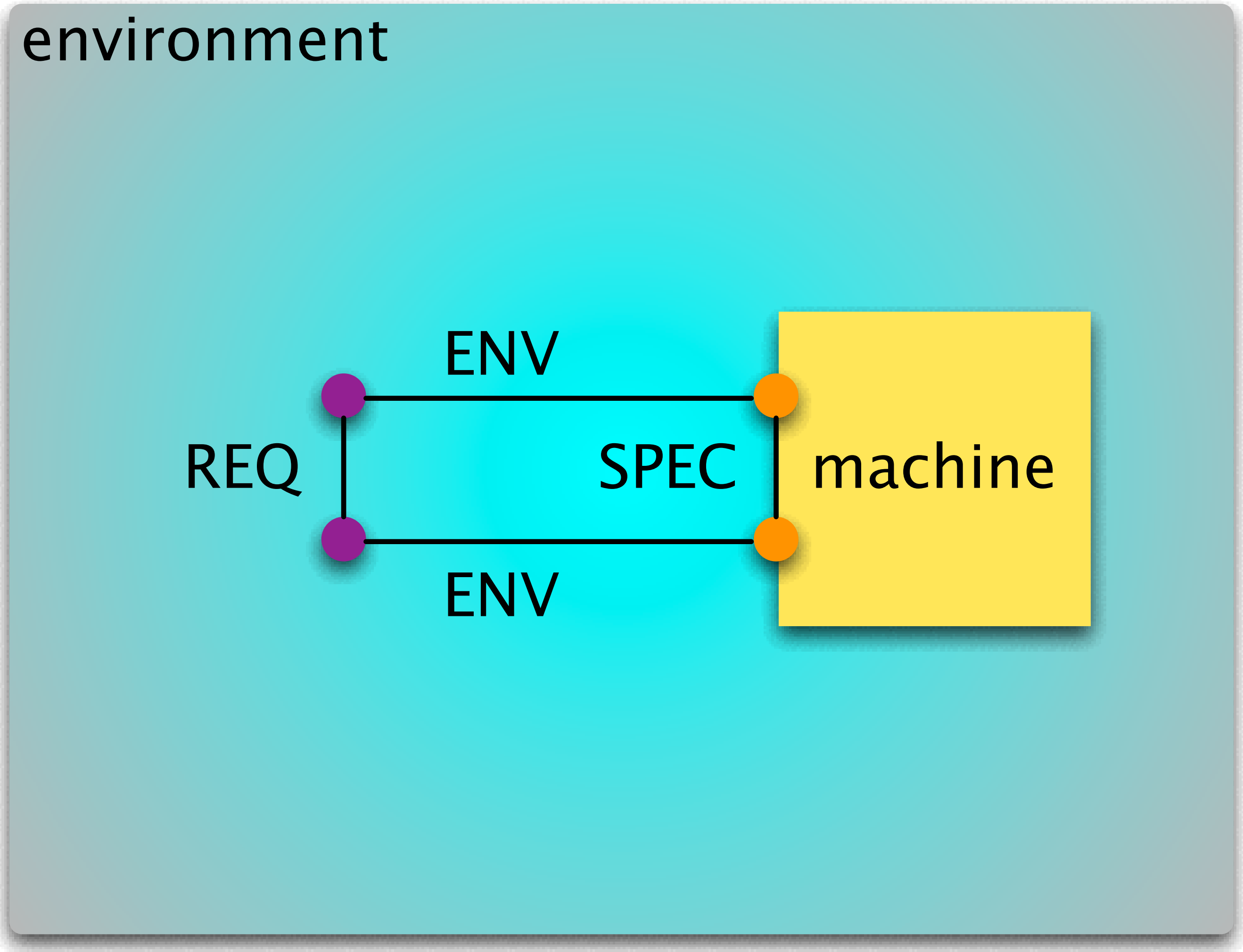
Premarket Notification 510(K): Regulatory Requirements For Medical Devices
HHS Publication FDA 95-4158, August 1995
http://www.fda.gov/cdrh/manual/510kprt1.html

USS Yorktown, 1998

dead in water for hours because divide-by-zero crashed network

Gregory Slabodkin
Software glitches leave Navy Smart Ship dead in the water
Government Computing News
July 13, 1998; Vol. 17 No. 17
http://www.gcn.com/17_17/news/33727-1.html

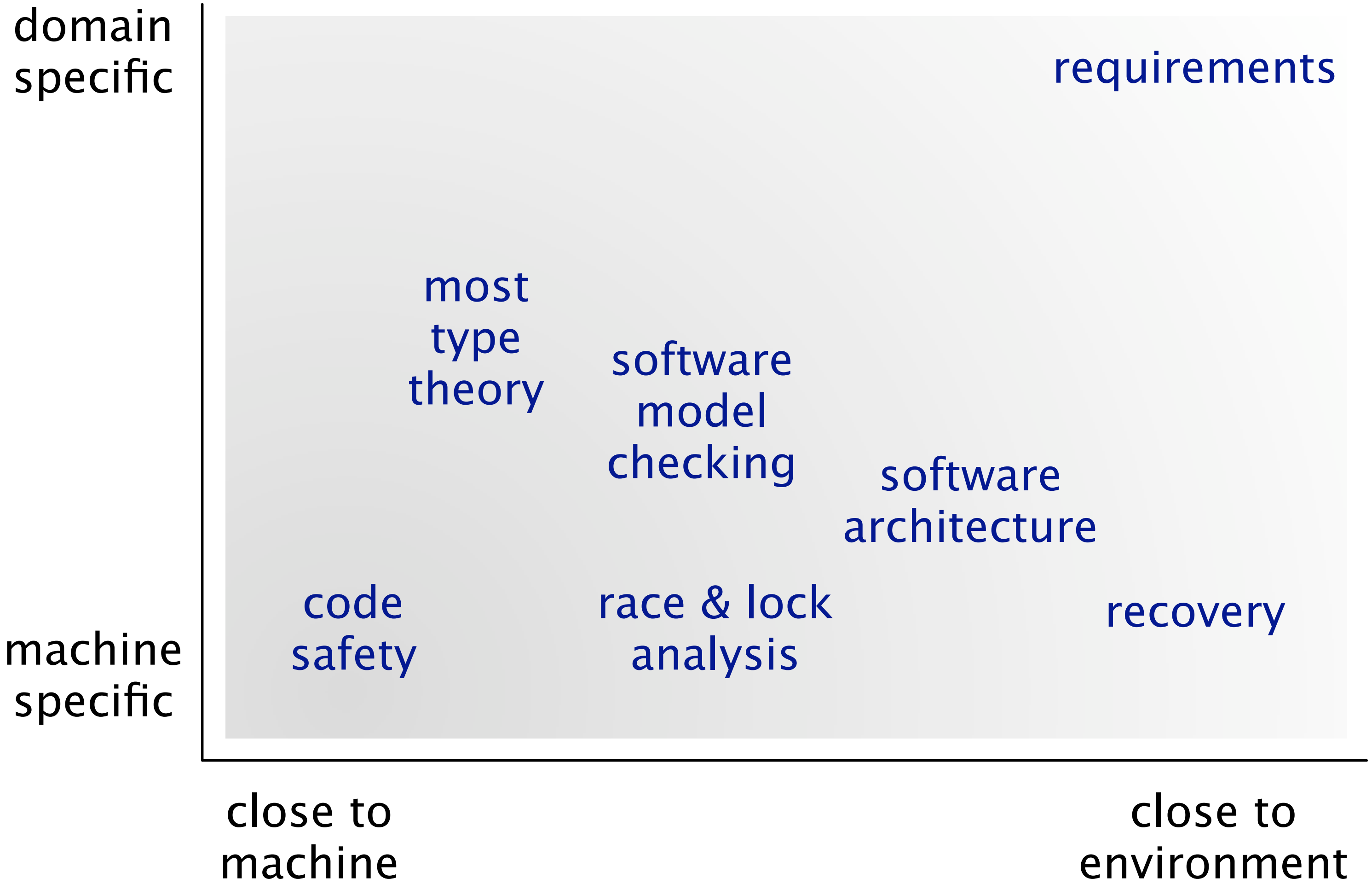
myth: correct ⇒ dependable



dependability argument: ENV ∧ SPEC ⇒ REQ

Michael Jackson. *Problem Frames.* Addison Wesley, 2001.

just lip service?



need research in all areas, but unhealthy bias to lower left

where are security vulnerabilities?

myth: tested ⇒ dependable

There is no evidence to suggest that the full system software, when commissioned, will not prove reliable.*
-- London Ambulance System Chief Executive

from 510k manual:

Software. If the device is computer controlled, software and/or hardware, validation and verification information must be included in the 510(k). The test data must support all performance and safety claims.

*Report of the Inquiry Into The London Ambulance Service, Feb 1993
The Communications Directorate, South West
Thames Regional Health Authority. Original ISBN No: 0 905133 70 6
Electronic Version prepared by Anthony Finkelstein
http://www.cs.ucl.ac.uk/staff/A.Finkelstein/las/lascase0.9.pdf
quoted in: Anthony Finkelstein, Unit 3: Case Study London Ambulance Service CAD System

myth: all requirements equal

prioritization: a hallmark of dependable systems

- some properties matter more than others

examples

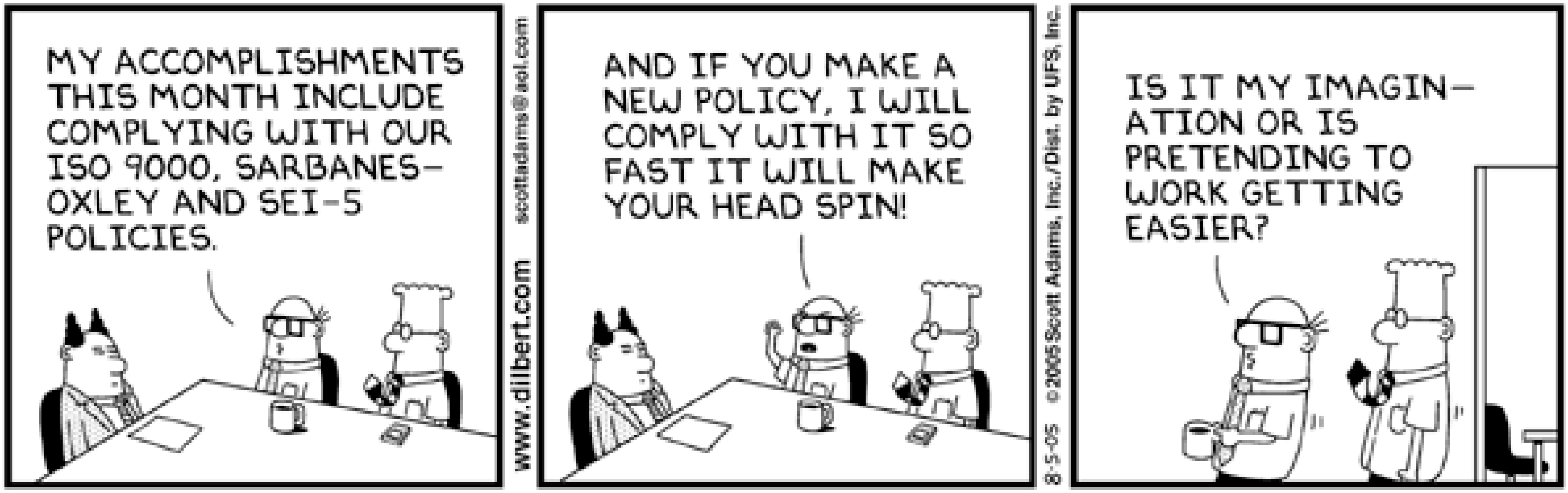
- radiotherapy
  1. dose delivered ≤ dose prescribed
  2. dose delivered = dose reported
- in-flight entertainment
  1. don’t interfere with flight (eg, by catching fire)

after 1998 Swiss Air crash

- FAA ordered switch to turn off power to entertainment system
- before, in 767-300’s, could only turn off with avionics too!

Gary Stoller. Game systems linked to scores of jet ‘difficulties’; USA Today, 07/08/2003;
http://www.usatoday.com/travel/news/2003/07/09-plane-games.htm

does certification help?


MY ACCOMPLISHMENTS THIS MONTH INCLUDE COMPLYING WITH OUR ISO 9000, SARBANES-OXLEY AND SEI-5 POLICIES.
AND IF YOU MAKE A NEW POLICY, I WILL COMPLY WITH IT SO FAST IT WILL MAKE YOUR HEAD SPIN!
IS IT MY IMAGIN-ATION OR IS PRETENDING TO WORK GETTING EASIER?
www.dilbert.com
scottadams@aol.com
8-5-05
© 2005 Scott Adams, Inc./Dist. by UFS, Inc.

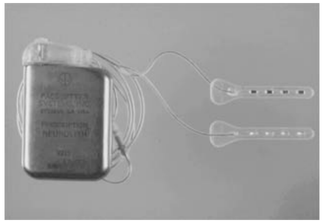
© Scott Adams, Inc./Dist. by UFS, Inc.

FDA device regulation

devices subject to regulation since 1976

Class 1: label, register, etc

Class 2: premarket notification (~$3k)



Class 3: premarket approval (~$260k)

premarket notification: 510k

safety demonstrated by

- ‘substantial equivalence’ to ‘predicate device’
- ‘consensus standards’ (eg, IEC 60601, UL 1998)

  typically include ‘verification’ (ie, testing)

timeliness

- 510k processed in weeks!

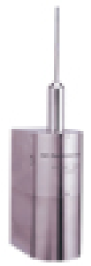
FDA also inspects device companies every 2-3 years

- 1993 inspection of Multidata found many problems
- lack of documentation, product shipped with known bugs
- overdoses of 20% had been noted

example

Carl Zeiss Surgical GmbH

- 510k application for INTRABEAM® System
- applied April 20, 05; granted May 23, 05



substantial equivalence argument

The INTRABEAM® System described in this Special 510(k) is a modification and enhancement of the PRS400 System cleared under K980526 and K992577. The INTRABEAM® System consists of the PRS500 Control Console, the XRS 4, the Workstation Software, the User Terminal, the verification accessories, and the treatment accessories. The PRS500 Control Console provides all low-level operational control and safety functions of this System. **The User Terminal with Workstation Software is the primary interface between the System and the user. The Software provides the high level control and display functions of the System. With the INTRABEAM® System, the radiologist, physicist, or technologist uses the Workstation Software on the User Terminal to set up the PRS500 Control Console (perform the pre-treatment verification and treatment planning), perform the treatment to completion, log all procedure variables and events, and after completion of the treatment, save and/or print treatment and performance data.**

safety argument

6. TECHNOLOGICAL CHARACTERISTICS AND SUBSTANTIAL EQUIVALENCE

Equivalence of the INTRABEAM® System with the PRS 400 System is based on intended use, indications for use, technological characteristics, system specifications, and operational characteristics.

7. TESTING

The INTRABEAM® System was systematically tested to demonstrate that the modifications did not adversely affect safety and effectiveness. Testing described in this premarket notification included functionality (integration testing of hardware, firmware, and accessories), safety, electrical safety, electromagnetic compatibility, environmental testing, mechanical stress testing, and validation testing.

other industries

in aviation

- DO178b, de facto standard developed by avionics industry

in process control (“SCADA”)

- IEC 61508 (safety of electronic & programmable systems)
- IEC 61511 (safety for process sector)
- ISA S84.01 (safety instrumented systems for process sector)

in aerospace

- NASA-STD-8719.13B (software safety)
- NASA-STD-8739.8 (software assurance process)

safety integrity levels

- 178B’s Level A (top) requires MCDC testing
- 61508’s SIL 1 (lowest): $< 10^{-5}$ failures/hour

certification: good or bad?

problems

- expensive and burdensome, often after the fact
  - eg, Common Criteria
- more to cover liability than prevent catastrophe?
  - eg, 510k?
- often customers and users are kept in the dark
  - eg, certification of electronic voting systems
- little evidence of efficacy
  - claimed integrity levels rarely even measurable

but

- may have collateral effects on software quality?
- help managers justify investment in dependability?

personal opinions

failures as basis for engineering

engineers learn from failure [Petroski, Salvadori, etc]

› failure is the norm, and the engineer's main challenge

› but software engineers still think in terms of 'correctness'

what do we have for software?

› RISKS Forum (anecdotal)

› occasional investigations (Ariane-5, Panama, etc.)

› a few academic studies (Ladkin, Johnson, Cook, Leveson, etc)

› mention in incident databases (FDA, FAA, etc)

maybe we need

› an NTSB for software?

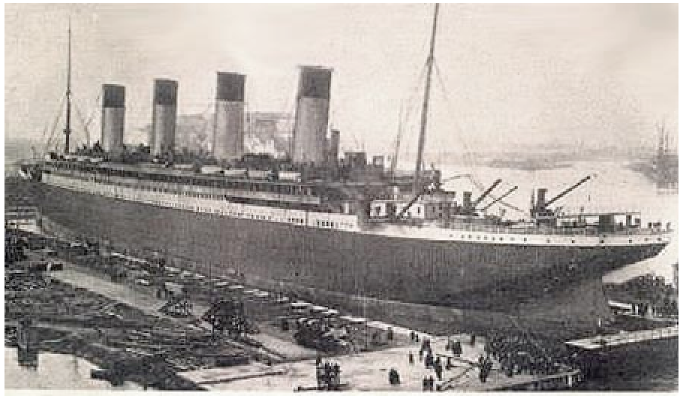
› incident reporting as for air-traffic controllers?

› mandatory black boxes in critical software?

elements of an approach

humility: our best effort may not be good enough

- › don’t build it
- › retain human in loop



THE NEW WHITE STAR LINER "TITANIC" (45,000 TONS) NEARING COMPLETION: DOCKED IN THE LARGEST GRAVING DOCK IN THE WORLD. BELFAST, FEBRUARY 1912.

software is only a part

- › organizational factors

explicit claims of dependability

- › should be demanded from developers
- › properties claimed, and levels of confidence

dependability justified by an end-to-end case

- › from requirements in domain down to code

where can RE contribute?

methods, languages & tools to

› support partiality, not completeness!

› make end-to-end arguments

    where does X fit in?

    $X \in$ {type checking, architectural compatibility, etc}

    what confidence from checking in finite scope?

› handle inconsistency due to redundancy

    requirement R is most important

    if R is not met, then meet R'

bridge the gap

› between technical requirements & organizational factors

the end

for more information

- http://cstb.org/project_dependable

opinions and pointers welcome!

- dnj@mit.edu