Fault-Tolerant Computer System Design
ECE 60872/CS 590

Topic 1: Introduction to Fault Tolerant Design

Saurabh Bagchi
ECE/CS
Purdue University

Class Structure

- Grade allocation
  - Course project: 50%
  - Mid-term: 15%
  - Final: 20%
  - Homeworks: 15%
- Homework 0: Due on Blackboard by Aug 26 (Sat).
  - Turn in a file with the following information: Name, Department, MS/PhD, Year of study, Thesis topic and advisor (if any), Previous course experience in distributed systems/OS/probability theory, Expectation from course (what topics and techniques you want to learn), Recent picture

Topics to be Covered: First 4 weeks

- Introduction
  - Motivation
  - High-level view of high availability design
  - Definitions
- Probability review, distributions
  - Discrete distributions
  - Continuous distributions
  - Calculation of fault-tolerance measures
- Hardware redundancy
  - Basic approaches
  - Static & Dynamic
  - Voting
  - *Application*: FTMP

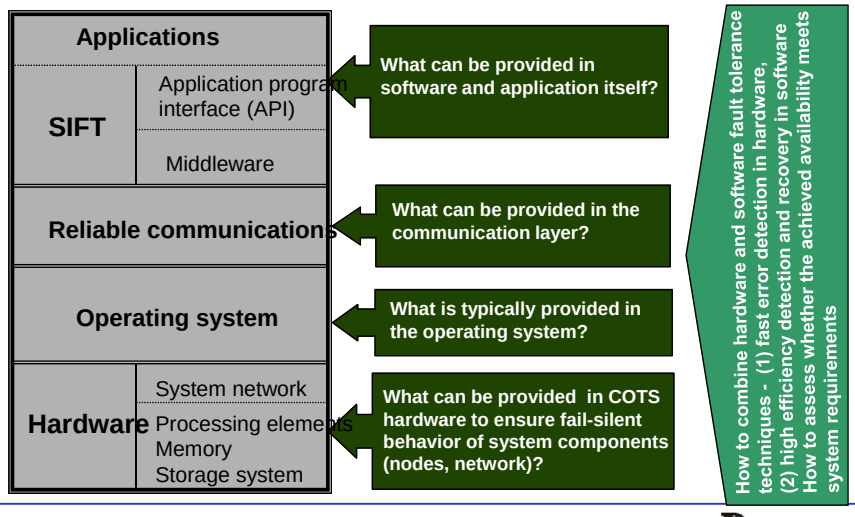
Why Study Reliable Computing!!!

- **Traditional needs**
  - Long-life applications (e.g., unmanned and manned space missions)
  - Life-critical, short-term applications (e.g., aircraft engine control, fly-by-wire)
  - Defense applications (e.g., aircraft, guidance & control)
  - Nuclear industry
- **Newer critical-computation applications**
  - Health industry
  - Automotive industry
  - Industrial control systems, production lines
  - Banking, reservations, switching, commerce

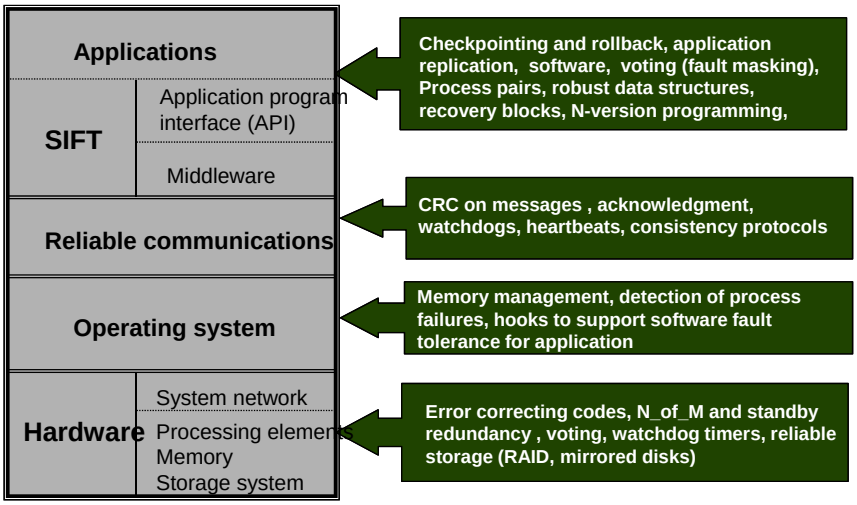
Why Study Reliable Computing!!! (cont.)

- **Networks**
  - Wired and wireless networked applications
  - Data mining
  - Information processing on the Internet
  - Distributed, networked systems
  - Intranet - commercial computing
- **Scientific computing, education**
  - Lots of cores
  - Petaflop machines being used for reliable computation

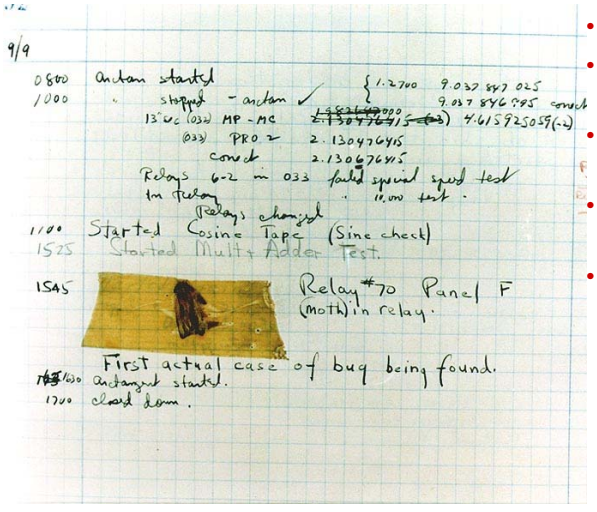
System Perspective on Design Issues in Reliable Computing



How do We Achieve the Objectives?



A Trip Back in History



- September 9, 1947
- Mark II computer at Harvard
- Popularly attributed to Grace Hopper
- Actually Bill Burke found the moth
- Diary with the taped moth can be seen at Smithsonian National Museum of American History

Costly Software Bugs in History

- Facebook IPO: May 2012
  - NASDAQ earmarked $62 million for reimbursing investors and says it expects to incur significant costs beyond that for system upgrades and legal battles. WSJ pegged the total loss to investors at $500M.
  - Investors were unsure how much of Facebook they'd bought. There were 12 million postponed share orders suddenly filled between 1:49 p.m. and 1:51 p.m. without being properly marked 'late sale', which exaggerated the impression that people were trying to dump Facebook shares.
  - **Diagnosis**: Computer systems used to establish the opening price were overwhelmed by order cancellations and updates during the "biggest IPO cross in the history of mankind," Nasdaq Chief Executive Officer Robert Greifeld said Sunday. Nasdaq's systems fell into a "loop" that prevented it from opening the shares on schedule

Costly Software Bugs in History

- US Power Grid Blackout: August 14, 2003
  - 50 million people lost power for up to two days in the biggest blackout in North American history. The event contributed to at least 11 deaths and cost an estimated $6 billion.
  - **Diagnosis**: The task force responsible for investigating the cause of the blackout concluded that a software failure at FirstEnergy Corp. "may have contributed significantly" to the outage.
  - FirstEnergy 's Alarm and Event Processing Routine (AEPR), a key software program that gives operators visual and audible indications of events occurring on their portion of the grid, began to malfunction. As a result, "key personnel may not have been aware of the need to take preventive measures at critical times".
  - Internet links to Supervisory Control and Data Acquisition (SCADA) software weren't properly secure and some operators lacked a system to view the status of electric systems outside their immediate control.

Costly Software Bugs in History

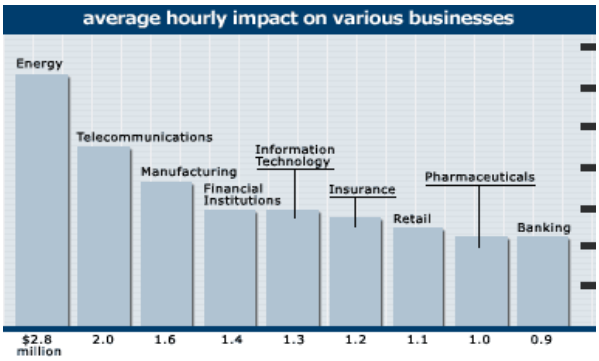
- 1996: European Space Agency Satellite Launcher crash

Costly Software Bugs in History

- Medical Devices: Ongoing
  - It has been shown to be possible for a heart defibrillator and pacemaker to reprogram it to shut down and to deliver jolts of electricity that could be fatal. (For a device called Maximo from industry #1 company called Medtronic)
  - Also possible to glean personal patient data by eavesdropping on signals from the tiny wireless radio embedded in the implant as a way to let doctors monitor and adjust it without surgery.
  - 1983-1997: There were 2,792 quality problems that resulted in recalls of medical devices, 383 of which were related to computer software (14%), according to a 2001 study analyzing FDA reports of the medical devices that were voluntarily recalled by manufacturers.
  - 2014: FDA had 43 recalls so far, categorized as ones where there is "reasonable probability that use of these products will cause serious adverse health consequences or death." At least 6 of the recalls were likely caused by software defects.

Effect of major network outages on large business customers

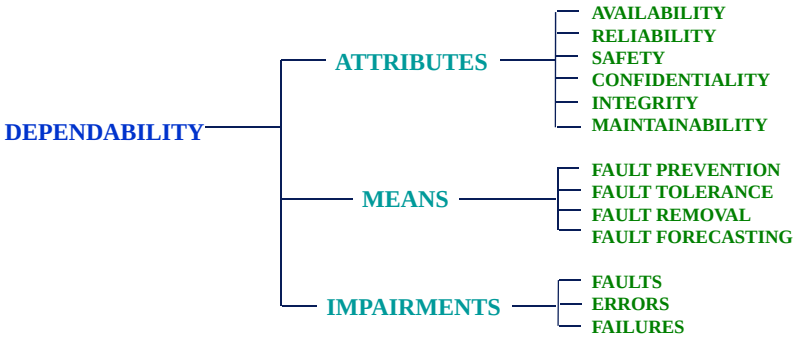
October 2000 data



- Survey by Meta Group Inc. of 21 industrial sectors in 2000 found the mean loss of revenue due to computer system downtime was $1.01M/hour

Dependable Computing

- Dependability is property of computer system that allows reliance to be placed justifiably on service it delivers. The service delivered by a system is its behavior as it is perceptible by its user



Fault Classes

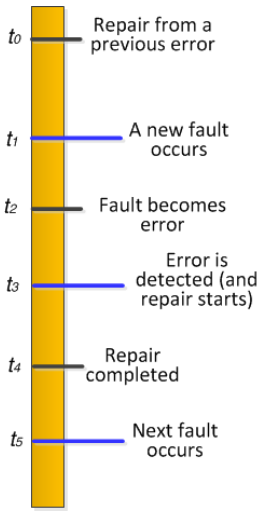
Based on the temporal persistence

- **Permanent** faults, whose presence is continuous and stable.
- **Intermittent** faults, whose presence is only occasional due to unstable hardware or varying hardware and software states (e.g., as a function of load or activity).
- **Transient** faults, resulting from temporary environmental conditions.

Based on the origin

- **Physical** faults, stemming from physical phenomena internal to the system, such as threshold change, shorts, opens, etc., or from external changes, such as environmental, electromagnetic, vibration, etc.
- **Human-made** faults, which may be either design faults, introduced during system design, modification, or establishment of operating procedures, or interaction faults, which are violation of operating or maintenance procedures.

Fault Cycle & Dependability Measures



Reliability:
a measure of the continuous delivery of service;
R(t) is the probability that the system survives (does not fail) throughout [0, t];
expected value: ***MTTF(Mean Time To Failure)***

Maintainability:
a measure of the service interruption
M(t) is the probability that the system will be repaired within a time less than t;
expected value: ***MTTR (Mean Time To Repair)***

Availability:
a measure of the service delivery with respect to the alternation of the delivery and interruptions
A(t) is the probability that the system delivers a proper (conforming to specification)service at a given time t.
expected value: ***EA = ???***

Safety:
a measure of the time to catastrophic failure
S(t) is the probability that no catastrophic failures occur during [0, t];
expected value:
MTTCF(Mean Time To Catastrophic Failure)

Good Sources of Information

- The Risks Digest. At: http://catless.ncl.ac.uk/Risks - Weekly to every two weeks; news of risks to the public in computers and related systems. Started by (and moderated by) Peter Neumann.
- Reading: Siewiorek and Swarz, Chapter 2 “Faults and their Manifestations” Pages 22-49
- A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004.

Presentation available on:
Course web page
engineering.purdue.edu/ee695b