Software Reliability and Security

Module 4

Winter 2017

Presentation/Lecture Schedule and Report Due Dates

- Presentation 1
  - Related background paper
  - Jan 27, Feb 1, 3
- Presentation 2
  - Project proposal
  - March 1, 3, 8
- Presentation 3
  - Final project report
  - March 24, 29, 31

- Lectures

  Jan 13, 18, 20, 25, 27

  Feb 1, 3, 8, 10, 15, 17

  March 1, 3, 8, 10, 15, 17, 22, 24, 29, 31

- Project Proposal Due

  Tuesday, February 28

- Final Project Report Due

  Monday, April 10

- Final Exam

  Wednesday, April 12, 10:00am

Warm-Up Presentations

- Presentations will start on January 27
- Summary of at least two full conference/journal papers published in 2012-2016
- You can discuss with me for selecting papers after the lectures
- How the papers are related to each other?
- How the general concept of the papers related to the course topics?
- You can choose papers thinking about your project (But it is ok if it is not related to the project you are thinking of)

Warm-Up Presentation - contd.

- Answer the following in your presentation
  - Main motivation?
  - Problems/contributions?
  - Solution approach?
  - Conclusions /lessons learned?
  - Future work?
- General Advice
  - Provide the paper and the slides to me before your presentation
  - Try to use your own examples in the presentation
  - Think about the audience in the class so that they can understand
- An interesting reference
  - http://www.acsac.org/speakers.pdf

Outline

- Software Reliability vs. Hardware Reliability
- Software Reliability Terminology
- Software Reliability Engineering Process
- Software Reliability Modeling

Software Reliability Engineering

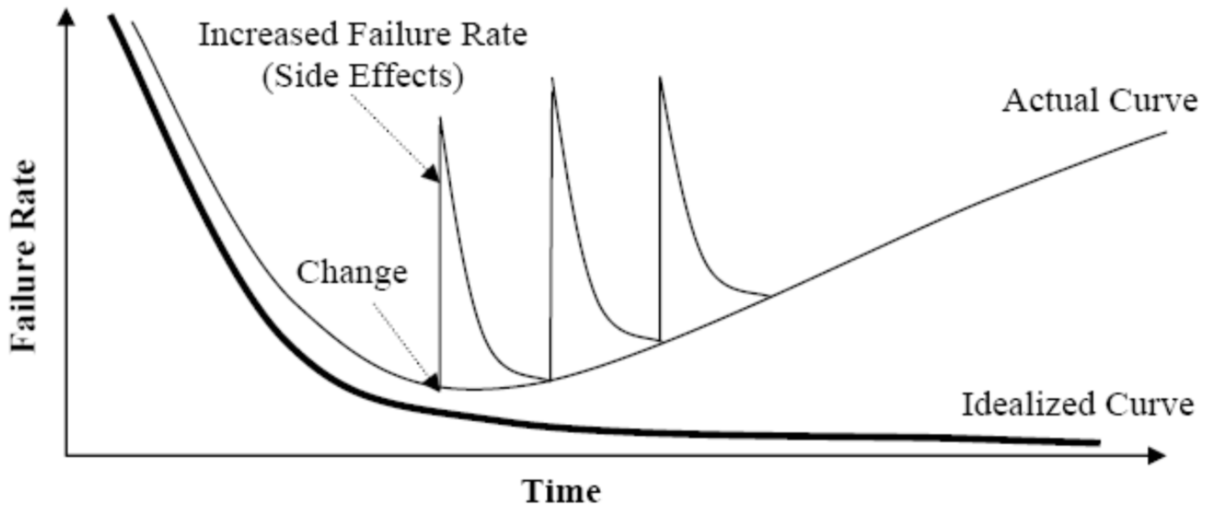
- Software Reliability
  - Probability of failure-free operation for a specific interval and environment
- Software Reliability Engineering (SRE)
  - The quantitative study of the operations of software systems with respect to users' reliability requirements IEEE95]
    - Software reliability measurement – reliability estimation and prediction
    - Attributes and metrics of software design, development, and the operation that affect reliability
    - The application of the above in the software development and maintenance phases

Software Reliability VS Hardware Reliability

- Failure Rate
  - SW: Failure rate is statistically non-increasing (without considering failure evolution)
  - HW: Failure rate has a bathtub curve

Software Failure Curve

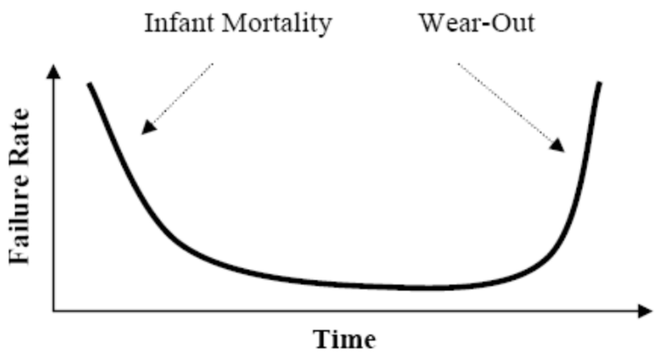
- Changes during maintenance introduce new defects that increase failure rate



Source: Pressman [13], Figure 1.3

Figure 3: Actual Software Failure Curve

Hardware Failure Curve



Source: Pressman [13], Figure 1.2

Figure 1: Hardware Failure Curve

- Three stages in a product life
  - a decreasing failure rate – due to defects and blunders
  - a low and relatively constant failure rate – random failures
  - an increasing failure rate - wear-out due to fatigue , material depletion

Software Reliability VS Hardware Reliability – contd.

- Failure in Idle State
  - SW: Failures do not occur when the software is not in use
  - HW: Material deterioration can lead to failures even though the system has not been used
- Reliability Models
  - SW: Most models are analytically derived based on assumptions
  - HW: Failure data are used to fit to some distributions

Software Reliability VS Hardware Reliability – contd.

- Cause of Failures
  - SW: incorrect logic, statements, or input data (from the environment) - similar to design errors of a complex hardware system
  - HW: material deterioration, random failures, design errors, misuse, and environmental change
- Reliability Improvement
  - SW: fix the detected defects through testing (reliability may change during testing due to defects in the new code or the removal of defects from the old code)
  - HW: by better design and material and by applying redundancy

Software Reliability VS Hardware Reliability – contd.

- Repair
  - SW: make a new software
  - HW: restore the original condition
- Warnings
  - SW: warning before failures - rare
  - HW: warning before failures - usually
- Component Standardization
  - SW: components are usually not standardized
  - HW: components are standardized
- Testing
  - SW: may need infinite testing
  - HW: exhaustive testing may be possible

- Lesson Learned: Achieving software reliability is much more difficult than achieving hardware reliability

Software Reliability - Terminology

- Software Fault, Error, Failure
  - Fault : a defect or bug in the code, cause of an error
  - Error : part of the system state which may lead to failure
  - Failure : unexpected (unspecified) externally observable behavior
- More discussions on error, fault, and failure in the next lecture

Software Reliability – More Terminology

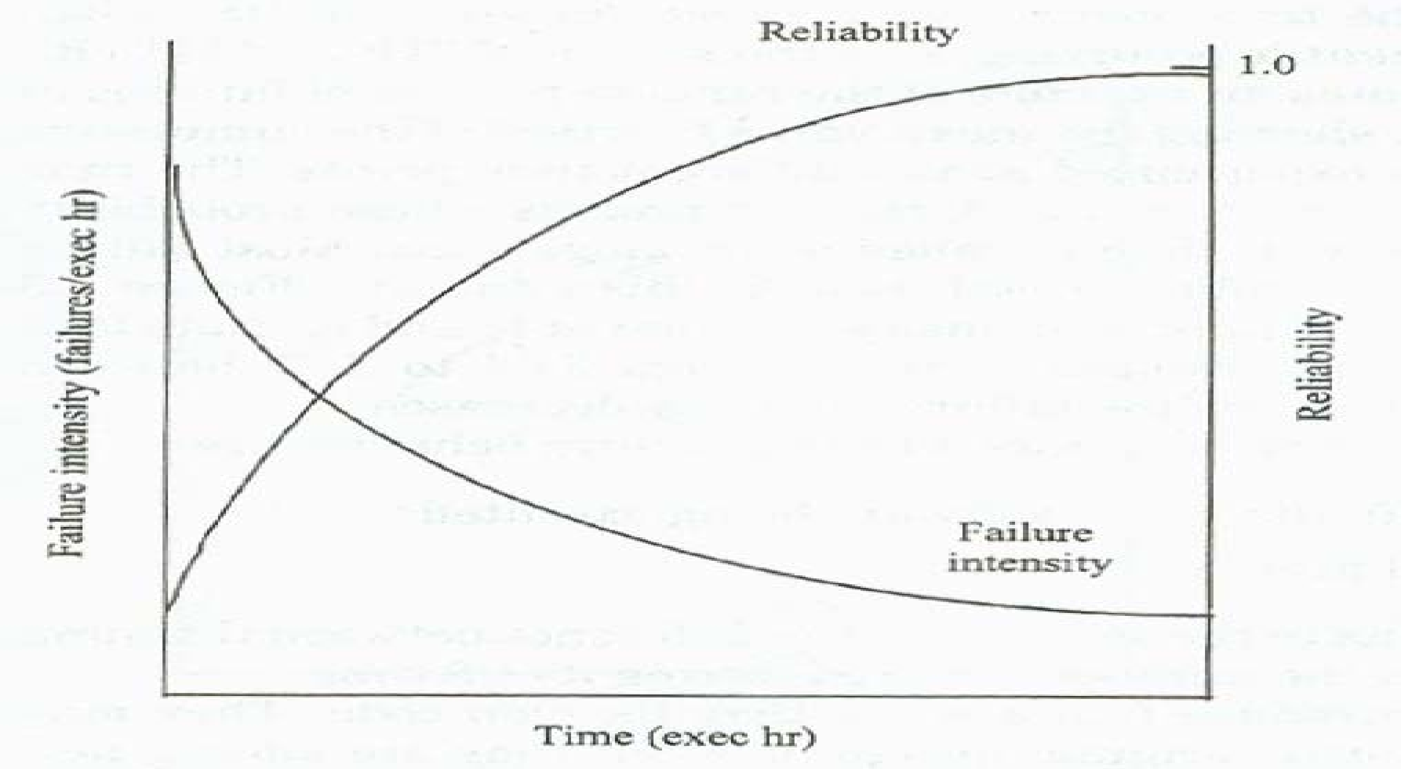
- Failure Functions
  - Cumulative failure function (mean value function)
    - Average cumulative failures at each point of time
  - Failure intensity function
    - The rate of change of the cumulative failure function or the number of failures per unit of time
  - Failure rate function (rate of occurrence of failures)
    - Probability that a failure/unit time occurs in an interval when a failure has not occurred before the beginning of that interval

Software Reliability – More Terminology

- Failure Functions – contd.
  - Mean Time to Failure Function (MTTF) or Mean Time Between Failure (MTBF)
  - Mean Time to Repair (MTTR)
  - Availability
    - Probability that a system is ready to use when needed = MTTF/(MTTF+MTTR)

Reliability and Failure Intensity

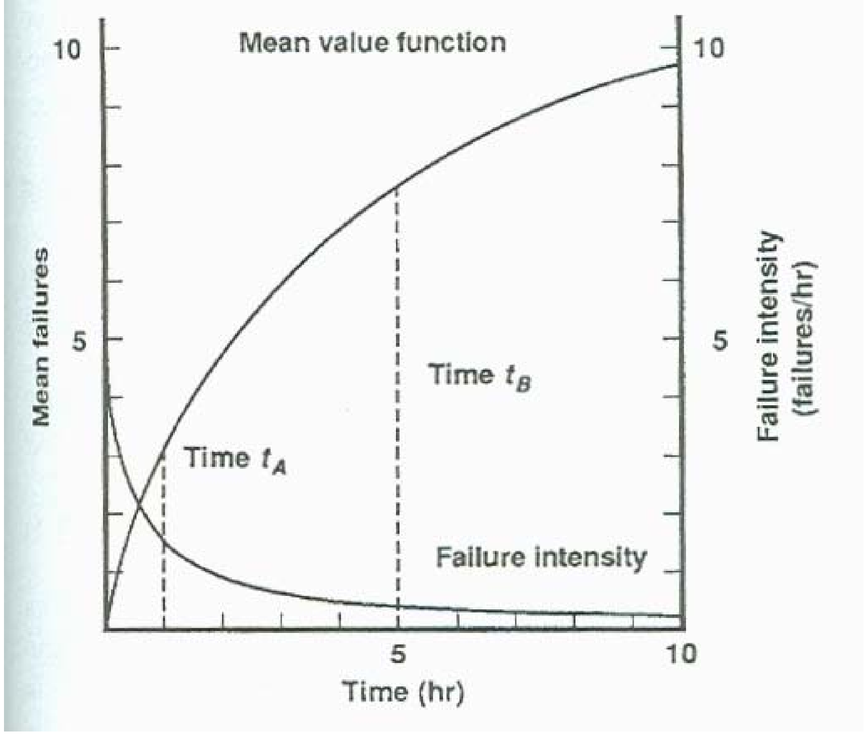
- Variation of reliability and failure intensity during a test period, as faults are removed



Mean Value and Failure Intensity

Mean Value Function (a.k.a. Cumulative failure function) - average cumulative failures associated with each point of time

Failure Intensity - The rate of change of mean value function or the number of failures per unit of time



Software Reliability: Some More Terminology – contd.

- Operational Profile
  - The set of operations (and the corresponding probabilities) that a software system will execute
  - An operation is a unique run (sequence f steps)
- Failure Data Collection
  - Failure-count data
    - a.k.a. failures per time period, e.g., failures/hour
  - Time-between-failures data
    - Mainly collected as mean time to failure
- Software Reliability Models
  - Predict future failures based on historically observed failures and a mathematical model

Summary

- Software reliability vs. hardware reliability
  - Achieving software reliability is much more difficult than achieving hardware reliability
- Software Reliability Terminology
  - Error, fault, and failure
  - Failure functions
  - Operational profile
  - Failure data collection
  - Software reliability models
- Software Reliability Engineering Process
- Software Reliability Growth Modeling

Lecture Sources

- Hoang Pham, Software Reliability, Springer, 2000.
- Pressman, Roger S., Software Engineering: A Practitioner's Approach, 4th ed., McGraw-Hill, 1997.
- John Musa, Software Reliability Engineering, McGraw-Hill, 1999.
- Paul Rook (editor), Software Reliability Handbook, Kluwer Academic Publishers, 2002.
- Michael R. Lyu (Editor), Handbook of Software Reliability Engineering, McGraw Hill Text, 1996.