Software and Software Engineering

Lecturer: Adel Vahdati

Software engineering

- IEEE Definition:
  - The application of a **systematic**, **disciplined**, **quantifiable** approach to the **development**, **operation**, and **maintenance** of software
  - Requires organizational commitment to quality
- Process, methods, and tools for building high-quality software.
- Enables timely and high-quality complex system development.
- Imposes discipline on potentially chaotic work.
- Allows adaptation to suit developers' needs.

Software Engineering - Process

- **Process:**
  - Basis for management control,
  - context for technical methods,
  - production of work products,
  - establishment of milestones,
  - quality assurance,
  - change management.
- **Collection of activities, actions, and tasks for creating work products.**
- **Adaptable approach allowing the software team to choose appropriate actions and tasks.**

Software Engineering – Methods & Tools

- **Methods:**
  - communication,
  - requirements analysis,
  - design modeling,
  - program construction,
  - testing,
  - support.
- **Tools:**
  - Provide automated or semi-automated support for processes and methods.

Work product perspectives

- **Engineer's view:**
  - Programs, data, and supporting work products.
- **User's view:**
  - Tools or products that improve their world.

Key realities for 21st-century software

- **Embedded in all aspects of life:**
  - understanding problems is crucial.
- **Increasingly complex IT requirements:**
  - design is pivotal.
- **Essential for decision-making and operations:**
  - must be high quality.
- **Growing demands for adaptation and enhancement:**
  - must be maintainable.
- **Conclusion:**
  - Software should be engineered across all domains.

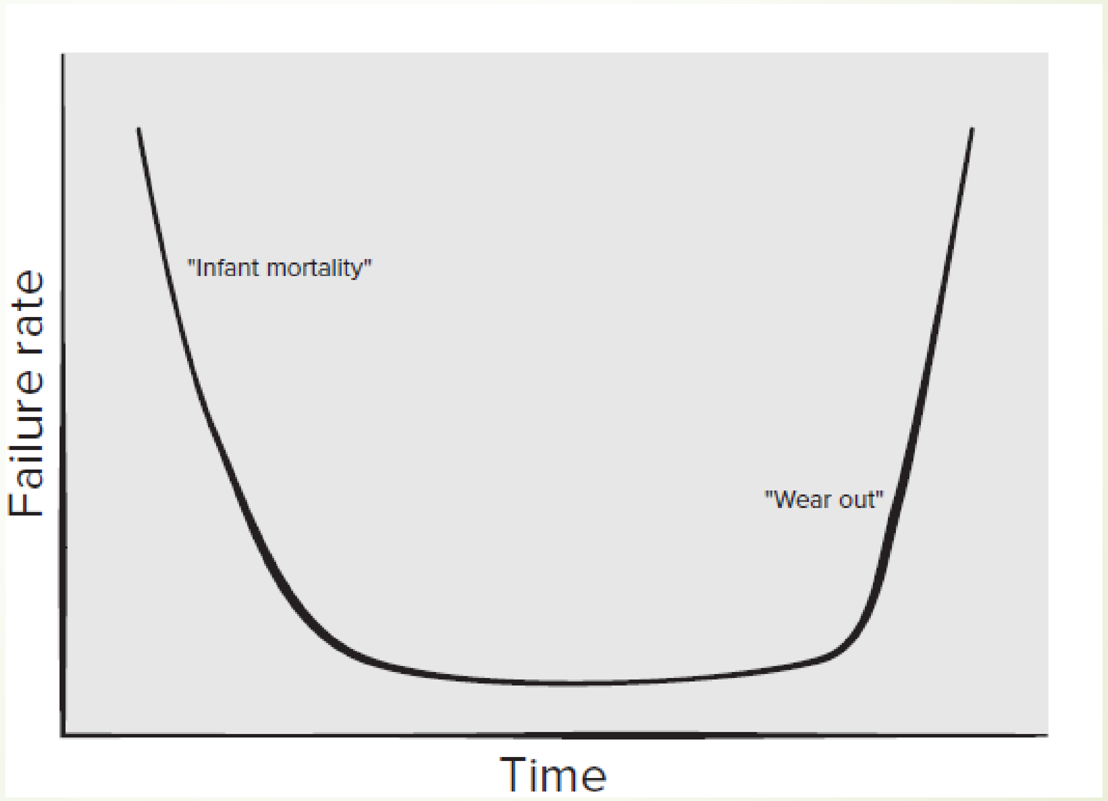
Defining software

- instructions (computer programs) that when executed provide desired features, function, and performance
- data structures that enable the programs to adequately manipulate information
- descriptive information in both hard copy and virtual forms that describes the operation and use of the programs.

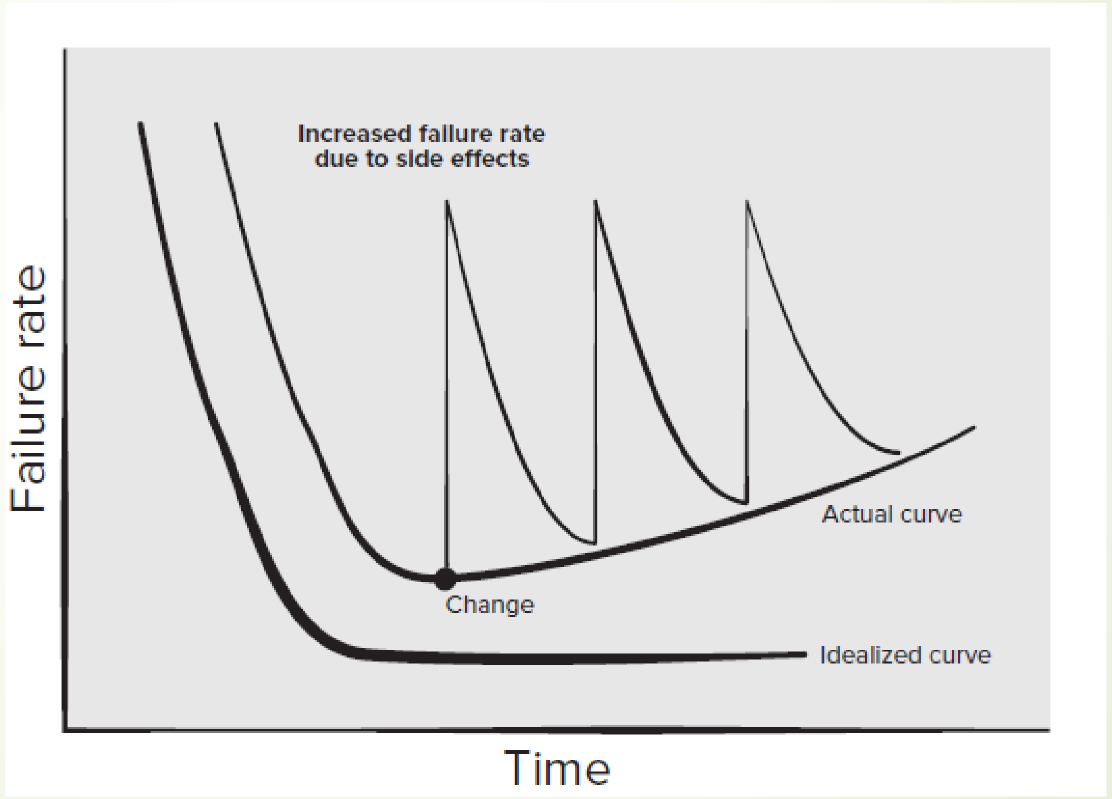
Software vs. hardware

- Software doesn't wear out but deteriorates with changes.
- No spare parts for software; failures indicate design or process errors.
- Software maintenance is more complex than hardware maintenance.

Hardware Failure Curve



Software Failure Curve



Categories of computer software

- System software: Services other programs.
- Application software: Solves specific business needs.
- Engineering/scientific software.
- Embedded software: Controls product/system features.
- Product-line software: Reusable components for various customers.
- Web/mobile applications.
- Artificial intelligence software: Uses heuristics for complex problems.

Generic Process Framework Activities

- **Communication:**
  - Understand stakeholders' objectives and gather requirements.
- **Planning:**
  - Create a software project plan outlining tasks, risks, resources, work products, and schedule.
- **Modeling:**
  - Create models to understand requirements and design.
- **Construction:**
  - Build the design through code generation and testing.
- **Deployment:**
  - Deliver the product to the customer for evaluation and feedback.

Umbrella Activities

- Applied throughout a software project to manage and control progress, quality, change, and risk.
- **Software project tracking and control:**
  - Assess progress and maintain schedule.
- **Risk management:**
  - Assess risks affecting project outcome or product quality.
- **Software quality assurance:**
  - Ensure software quality through defined activities.
- **Technical reviews:**
  - Uncover and remove errors in work products.

Umbrella Activities (2)

- **Measurement:**
  - Collect process, project, and product measures to meet stakeholders' needs.
- **Software configuration management:**
  - Manage change effects throughout the process.
- **Reusability management:**
  - Define criteria and mechanisms for work product reuse.
- **Work product preparation and production:**
  - Create models, documents, logs, forms, and lists.

Essence of Software Engineering Practice

- **Understand the problem** (communication and analysis).
- **Plan a solution** (modeling and software design).
- **Carry out the plan** (code generation).
- **Examine the result for accuracy** (testing and quality assurance).

General Principles

- **The Reason It All Exists:** Provide value to users.
- **KISS:** Keep designs as simple as possible.
- **Maintain the Vision:** Ensure conceptual integrity.
- **What You Produce, Others Will Consume:** Make work understandable for others.
- **Be Open to the Future:** Adapt to changes.
- **Plan Ahead for Reuse:** Save time and effort through reuse.
- **Think!:** Complete thought before action.

References

- R. S. Pressman and B. R. Maxim. Software Engineering: A Practitioner's Approach. 9th Edition, McGraw-Hill, 2019.