Introduction to Software Engineering

What is software engineering?

- Software engineering is an engineering discipline that is concerned with all aspects of software production.

- That is from the early stages of system specification to maintaining the system after it has gone into use.

Software Engineering -Definition

- **General Definition**
- Software engineering is the **establishment and use of engineering principles** in order to obtain economically feasible software that is **reliable and works efficiently** on real machines.

- **IEEE Definition**
- The application of a **systematic, disciplined, quantifiable** approach to the **development, operation, and maintenance of software**; that is, the application of engineering to software.

WHY IS SOFTWARE ENGINEERING IMPORTANT?

- Producing a software application is relatively simple in concept: Take an idea and turn it into a useful program.
- Unfortunately for projects of any real scope, there are countless ways that a simple concept can go wrong.
- Programmers may not understand what users want or need so they build the wrong application.
- The program might be full of bugs that it’s frustrating to use, impossible to fix, and can’t be enhanced over time.

- **Software engineering includes techniques for avoiding the many pitfalls.**

- It ensures the final application is effective, usable, and maintainable.

- It helps to meet milestones on schedule and produce a finished project on time and within budget.

- Perhaps most important, software engineering gives us the flexibility to make changes to meet unexpected demands without completely affecting our schedule and budget constraints.

Software characteristics

- Software is developed or engineered; it is not manufactured.
- Software does not “wear out” but it does deteriorate.
- Software continues to be custom built, as industry is moving toward component based construction.

Software characteristics

- What is “software reliability”?
- The system “stands the test of time.”
- There is an absence of known catastrophic errors (those that disable or destroy the system).
- The system recovers “gracefully” from errors.

- How do we measure reliability?
- Downtime is below a certain threshold.
- The accuracy of the system is within a certain tolerance.
- Real-time performance requirements are met consistently

Software doesn't wear out, so why would it conform to the bathtub curve?

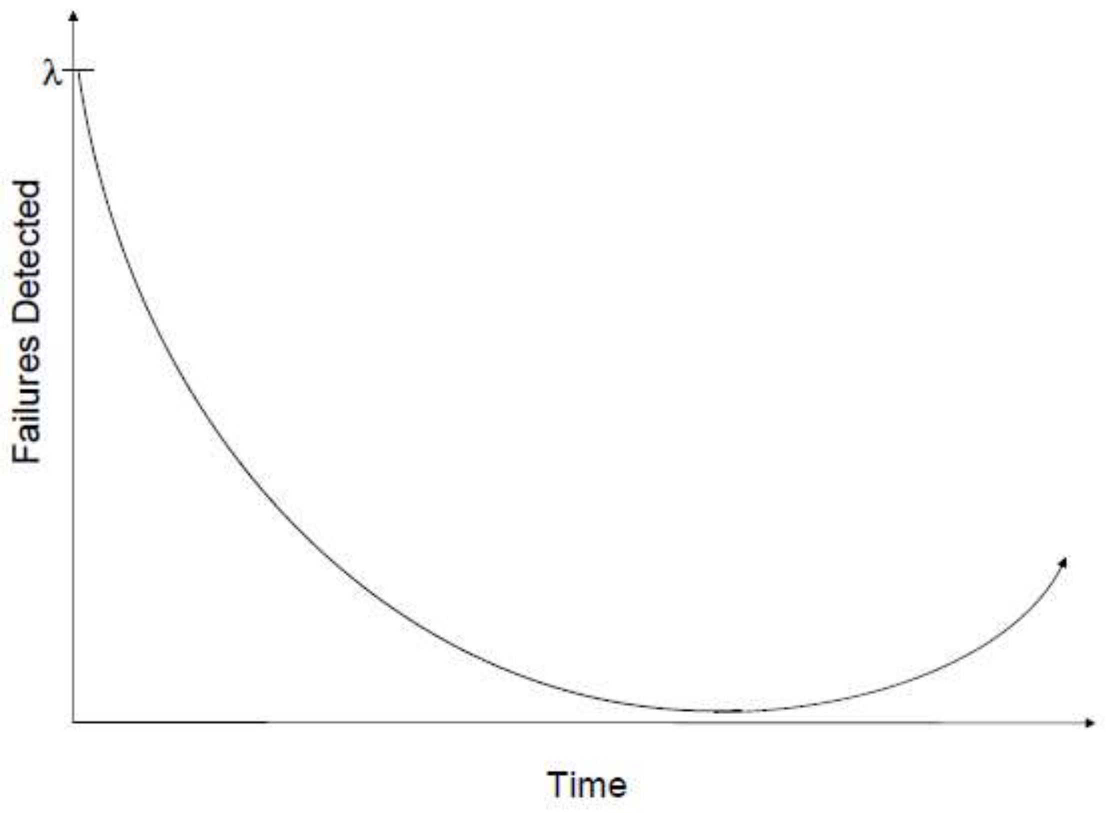

FIGURE 2.2
A software failure function represented by the bathtub curve.

Software characteristics

What is meant by the “correctness” of software?

Correctness can be measured in terms of the number of failures detected over time.

What is software “performance”?
Performance is a measure of some required behavior

How do we characterize software usability?
Usability is a measure of how easy the software is for humans to use. Software usability is synonymous with ease-of-use, or user-friendliness.

Software characteristics

What is interoperability?

- This quality refers to the ability of the software system to coexist and cooperate with other systems.

What is software “maintainability, evolvability, and repairability”?

- A software system in which changes are relatively easy to make has a high level of maintainability.
- Evolvability is a measure of how easily the system can be changed to accommodate new features or modification of existing features.
- Repairability is the ability of a software defect to be easily repaired.

Software characteristics

What is meant by “portability”?
Software is portable if it can run easily in different environments.

What is “verifiability”?
A software system is verifiable if its properties, including all of those previously introduced, can be verified easily.

What is “traceability” in software systems?
Traceability is concerned with the relationships between requirements, their sources, and the system