Introduction of Software Engineering

Chapter 1:

Overview of Software Engineering

VŨ THỊ TRÀ

CONTENTS

The Nature of Software

1. What is it?
   - ✓ Programs that execute within a computer
2. Who does it?
   - ✓ Software engineer
3. Why is it important?
   - ✓ It affect every aspect of our lives from our commerce, our culture to our everyday activities.

The Nature of Software

4. What are steps?
   - ✓ Customers and stakeholders express the need for CS
   - ✓ Engineer build the software product
   - ✓ End users apply the software to solve a specific problem
5. What are the work product?
   - ✓ A computer program
6. How do I ensure that I've done it right?
   - ✓ If you're an end user, …
   - ✓ It you're a software engineer, …

The Nature of Software

- Computer software becomes an indispensable technology for business, science, and engineering.
- CS enables
  - ✓ The creation of new technologies
  - ✓ The extension of existing technologies
  - ✓ The radial change in older technologies (e.g. the media)
- CS is driving the personal computer revolution.

The Nature of Software

- "*Ideas and technological discoveries are the driving engines of economic growth.*"

  Wall Street Journal

- Software becomes embedded in systems of all kinds: transportation, medical, telecommunications, military, industrials, entertainment, office machines,…
- Millions of computer programs would have to be corrected, adapted, and enhanced as time passed that leads to the creation of new software.
- CS continually attempted to develop technologies that will make it easier, faster, and less expensive to build and maintain high-quality computer programs.

The Nature of Software

- One of these technologies are targeted at a specific application domain (e.g., website design and implementation); others focus on a technology domain (e.g., object-oriented systems or aspect-oriented programming); and still others are broad-based (e.g., operating systems such as Linux).

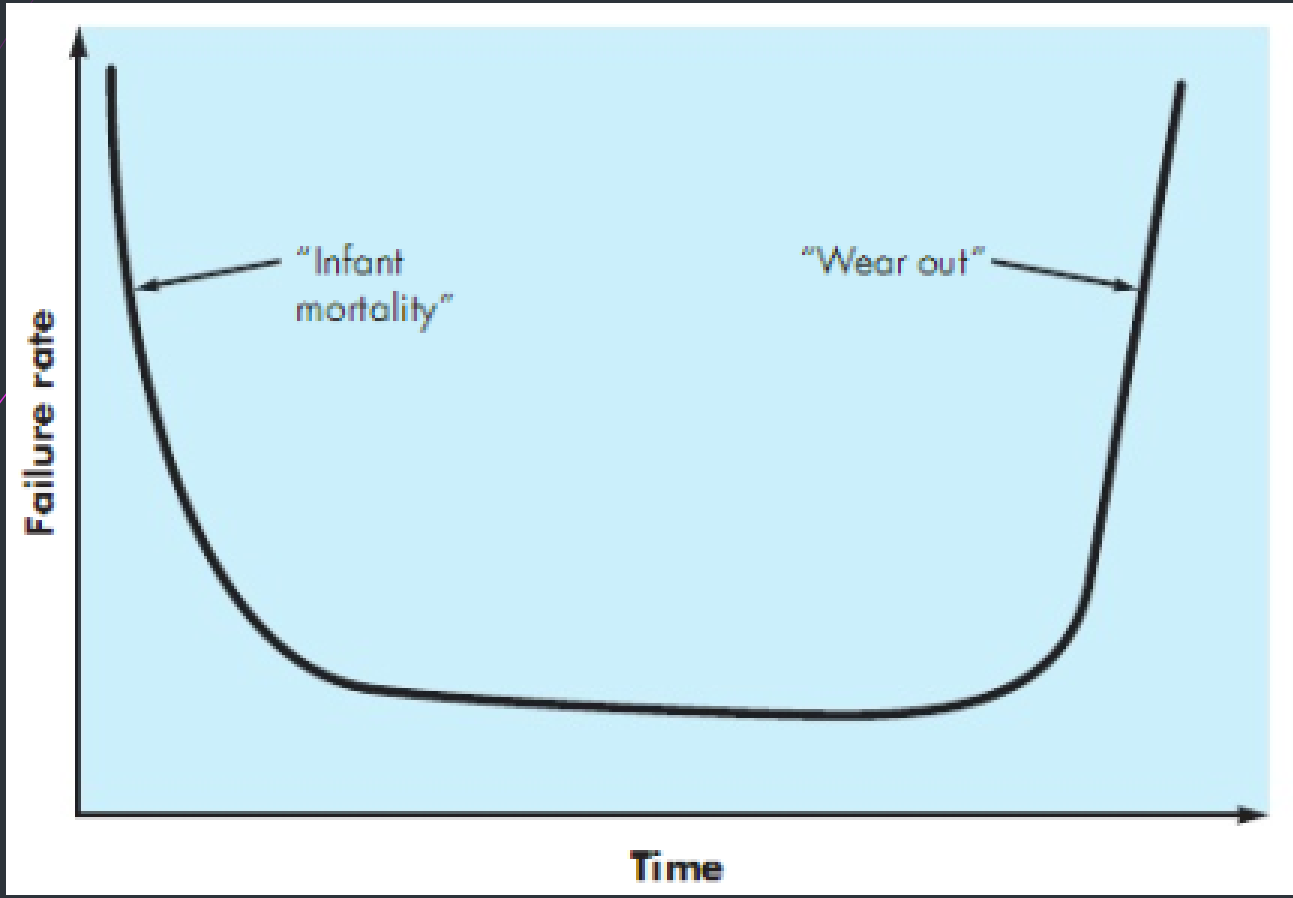
The Nature of Software

- Software is both a product and a vehicle.
- Typical questions are asked
  - ✓ Why does it take so long to get software finished?
  - ✓ Why are development costs so high?
  - ✓ Why can't we find all errors before we give the software to our customers?
  - ✓ Why do we spend so much time and effort maintaining existing programs?
  - ✓ Why do we continue to have difficulty in measuring progress as software is being developed and maintained?

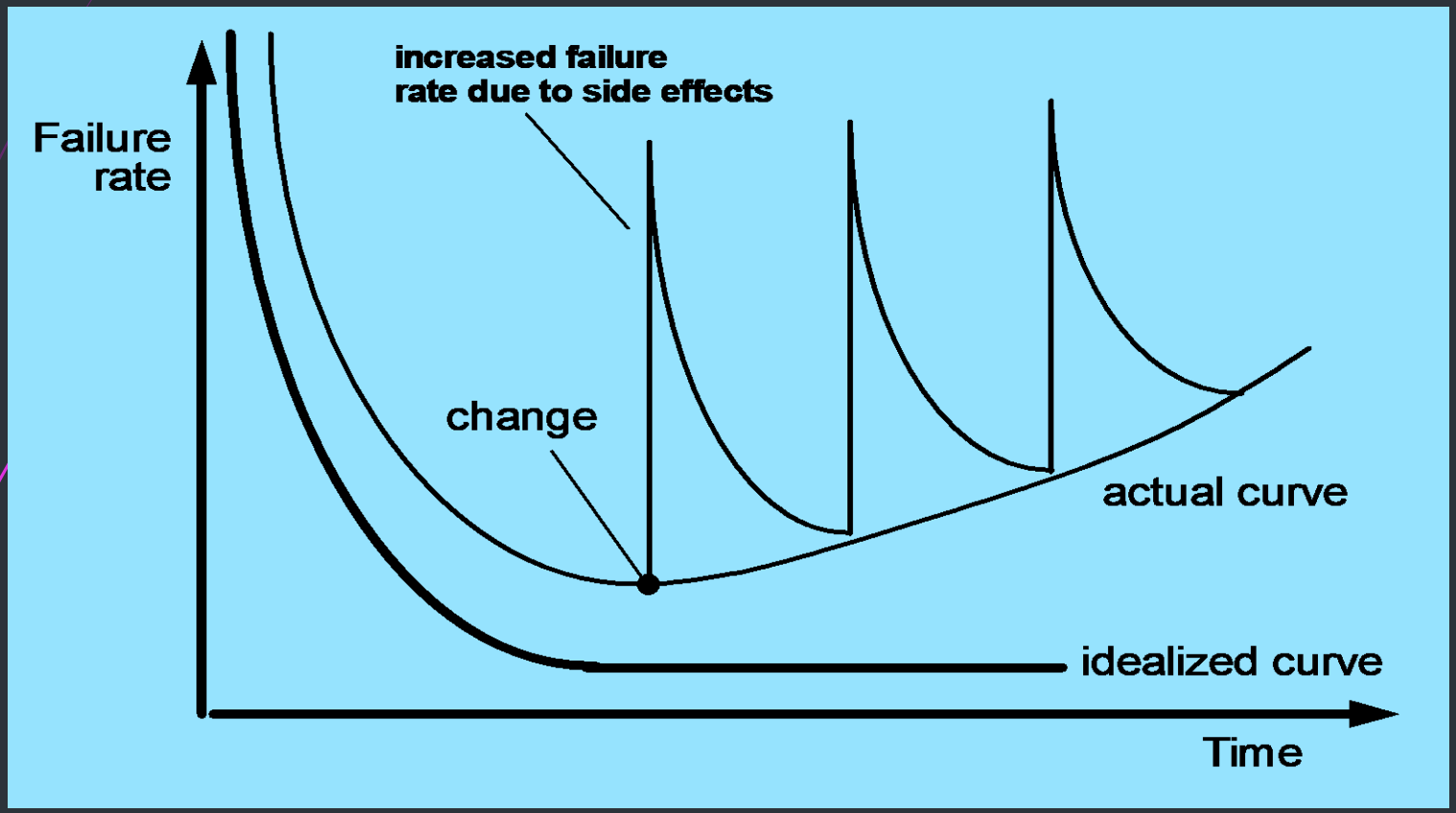
The Definition of Software

- Software is
  - ✓ 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 other programs.

Failure curve for hardware



Software Failure Curve



Software engineering methods strike to reduce the magnitude of the spikes and the slope of the actual curve.

The Nature of Software

- If you want to reduce software deterioration, you'll have to do better software design.

Software Application Domains

1. System software
   - ✓ programs written to service other programs
   - ✓ compilers, editors, file management utilities,…
   - ✓ operating system components, drivers, networking software, telecommunications processors,…
2. Application software
   - ✓ standard-alone programs that solve a specific business need
3. Engineering/ scientific software
   - ✓ number-crunching programs that range from astronomy to volcanology, from automotive stress analysis to orbital dynamics, and from computer-aided design to molecular biology, from genetic analysis to meteorology.

Software Application Domains (cont.)

4. Embedded software
   - ✓ resides within a product or system
   - ✓ is used to implement and control features and functions for the end user and for the system itself.
5. Product-line software
   - ✓ designed to provide a specific capability for use by many different customers.
6. Web/ Mobile application
   - ✓ network-centric software category spans a wide array of applications and encompasses both browser-based apps and software that resides on mobile devices.
7. Artificial intelligent software
   - ✓ makes use of nonnumerical algorithms to solve complex problems that are not amenable to computation or straightforward analysis.

Legacy Software

- Developed decades ago and have been continually modified to meet changes in business requirements and computing platforms.
- Costly to maintain and risky to evolve.
- Often evolve for one or more of the following reasons:
  - ✓ The software must be adapted to meet the needs of new computing environments or technology.
  - ✓ The software must be enhanced to implement new business requirements.
  - ✓ The software must be extended to make it interoperable with other more modern systems or databases.
  - ✓ The software must be re-architected to make it viable within a evolving computing environment.

Legacy Software

- Software engineering must recognize that change is natural. Don't try to fight it.

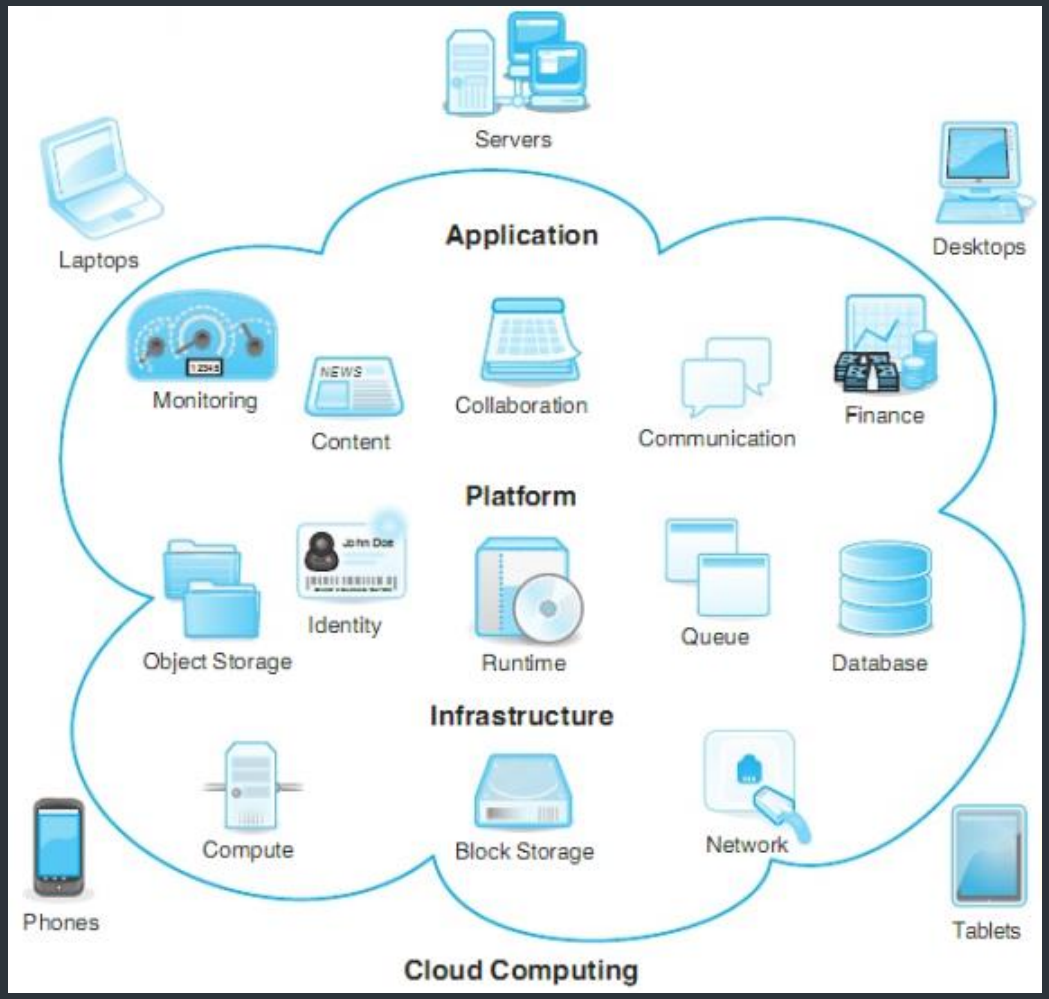
The Changing Nature of Software

1. WebApps
   - ✓ 1990 – 1995: Word Wide Web … Web-based systems and application (known as WebApps).
   - ✓ Over the past decade: Web 3.0 - Sematic Web technologies, Sophisticated relational data structures will lead to entirely new WebApps that allow access to disparate information in ways never before possible.
2. Mobile Applications
   - ✓ A mobile web app allows a mobile device to gain access to web-based content via a browser.
   - ✓ A mobile app can gain direct access to the hardware characteristics of the device.

The Changing Nature of Software (cont.)

3. Cloud Computing
   - ✓ An infrastructure or "ecosystem" that enables any user anywhere, to use a computing device to share computing resources on a broad scale.
   - ✓ The development of an architecture that encompasses front-end and back-end services.
   - ✓ A variety of different levels from full public access to private cloud architectures.
4. Product Line Software
   - ✓ A set of software-intensive systems that share a common, managed sec of features satisfying the specific needs of a particular marker segment or mission.
   - ✓ That are developed from a common set of core assets in a prescribed way.

Cloud computing logical architecture



CONTENTS

- **The Nature of Software**
- **Software Engineering**

Software Engineering

1. What is it?
   - ✓ a process, a collection of methods and an array of tools that allow professionals to build high-quality CS.
2. Who does it?
   - ✓ Software engineer
3. Why is it important?
   - ✓ It enables us ta build complex systems in a timely manner and with high quality.
   - ✓ It can become quite chaotic, but it also allows the people who build CS to adapt their needs.

Software Engineering

4. What are steps?
   - ✓ Build CS by applying an agile, adaptable process that leads to a high-quality result that meets the needs
5. What are the work product?
   - ✓ The software engineer's view: computer program
   - ✓ The user's view: the resultant information
6. How do I ensure that I've done it right?
   - ✓ Following the lessons, select those ideas that are applicable to the software that you build

Software Engineering

- **Tips:**
  - ✓ *Understand the problem before you build a solution.*
  - ✓ *Both quality and maintainability are on outgrowth of good design.*

The Definition of Software Engineering

1. The application of a systematic, disciplined. quantifiable approach to the development, operation, and maintenance of software: that is, the application of engineering to software
2. The study of approaches as in (1)

- *Software engineering encompasses a process. methods for managing and engineering software and tools.*

Software Engineering Layers

- ✓ *methods* provide the technical how-to's for building software.
- ✓ *tools* provide automated or semi-automated support for the process and the methods.
- ✓ *a process* is a collection of work activities, actions, and tasks that are performed when some work product is to be created.

The Software Process

- *"A process defines who is doing what when and how to reach a certain goal."*

  Ivan Jacobson, Grady Booch, and James Rumbaugh

The Five Generic Process Activities

1. Communication
   - ✓ gather requirements that help define software
2. Planning
   - ✓ define scopes, the technical task, the risks, the resources, the work product, the work schedule
3. Modelling
   - ✓ analysis and design
4. Construction
   - ✓ code and test
5. Deployment
   - ✓ complete and deliver product

Software Engineering

- Software Engineering Practices
- Software Development Myths