2019 级本科

软件工程

Software Engineering

张昕

zhangxin@cust.edu.cn

计算机科学技术学院软件工程系

About course

- Total hours

  32 hours (theoretical) + 16 hours (experimental)

- Prerequisite

  Introduction to Computers, Programming Language,

  Data Structure, Software Architecture

- Further course

  Project Mangement, Software Process

  Management, Software Quality Assurance, etc.

- Class requirements

  Class note is essential, courseware is assistant

  Focus on training your capability

  Strengthen self-learning and knowledge mining

- ☐ Keep attendance
- ☐ Submit answered assignment on time
- ☐ Take experiments seriously

The nature of software

Questions to answer for software developers

- 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?

Defining Software

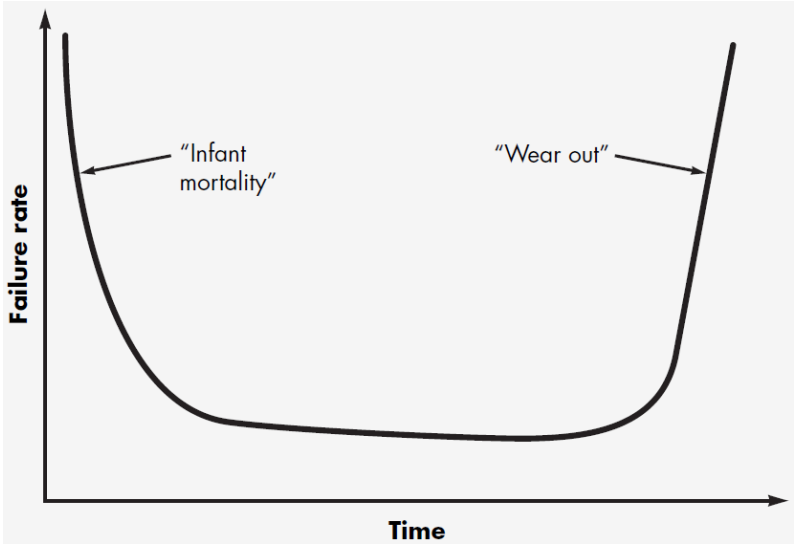
General definitions of Software

Software is:

(1) instructions (computer programs) that when executed provide desired features, function, and performance;

(2) data structures that enable the programs to adequately manipulate information, and

(3) descriptive information in both hard copy and virtual forms that describes the operation and use of the



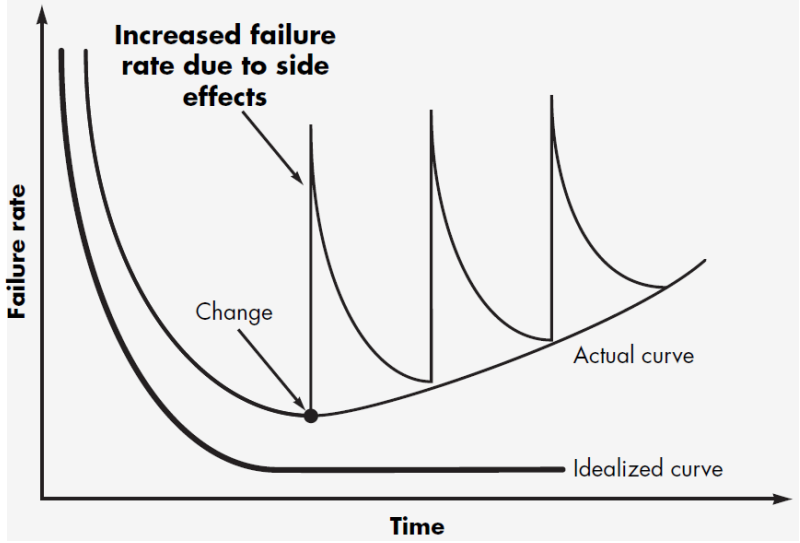
Failure curve for hardware

- Software is a logical rather than a physical system element.
- Software has one fundamental characteristic that makes it considerably different from hardware:
  - Software doesn't "wear out."

Defining Software

- Software is not susceptible to the environmental maladies that cause hardware to wear out.

  Undiscovered defects will cause high failure rates early in the life of a program.



Failure curve for software

The software maintenance tasks that accommodate requests for change involve considerably more complexity than hardware maintenance.

- The idealized curve is a gross oversimplification of actual failure models for software.
- However, the implication is clear—software doesn't wear out.
  - But it does deteriorate!
- During its life, software will undergo change. As changes are made, it is likely that errors will be introduced, causing the failure rate curve to spike.
- Before the curve can return to the original steady-state failure rate, another change is requested, causing the curve to spike again.
- Slowly, the minimum failure rate level begins to rise—the software is deteriorating due to change.

Software Application Domains

- Seven broad categories of computer software present continuing challenges for software engineers
  - System software: collection of programs written to service other programs.
    - e.g., compilers, editors, and file management utilities to processes complex, but determinate, information structures.
    - e.g., operating system components, drivers, networking software, telecommunications processors to process largely indeterminate data.
  - Application software: stand-alone programs that solve a specifi c business need.
    - To process business or technical data in a way that facilitates business operations or management/technical decision.
  - Engineering/scientific software: a broad array of "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.
  - Embedded software: resides within a product or system and is used to implement and control features and functions for the end user and for the system itself.
    - e.g., key pad control for a microwave oven, digital functions in an automobile such as fuel control, dashboard displays, and braking systems.

Software Application Domains

- Seven broad categories of computer software present continuing challenges for software engineers
  - Product line software: designed to provide a specific capability for use by many different customers.
    - e.g., inventory control products.
  - Web/Mobile applications: this network-centric software category spans a wide array of applications and encompasses both browser-based apps and software that resides on mobile devices.
  - Artificial intelligence software: makes use of nonnumerical algorithms to solve complex problems that are not amenable to computation or straightforward analysis..
    - e.g., robotics, expert systems, pattern recognition (image and voice), artificial neural networks, theorem proving, and game playing.

Legacy Software

- These older programs—often referred to as legacy software —have been the focus of continuous attention and concern since the 1960s.
  - Legacy software systems . . . were developed decades ago and have been continually modified to meet changes in business requirements and computing platforms. The proliferation of such systems is causing headaches for large organizations who find them costly to maintain and risky to evolve. [Day99]
  - "many legacy systems remain supportive to core business functions and are 'indispensable' to the business." [Liu98]
  - Legacy software systems always have one characteristic: poor quality
    - These systems support "core business functions and are indispensable to the business."
    - What to do?
      - The only reasonable answer may be: Do nothing, at least until the legacy system must undergo some significant change.
      - If the legacy software meets the needs of its users and runs reliably, it isn't broken and does not need to be fixed.

Legacy Software

- As time passes, legacy systems 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.
- When these modes of evolution occur, a legacy system must be reengineered so that it remains viable into the future.
  - The goal of modern software engineering is to "devise methodologies that are founded on the notion of evolution;"
  - i.e., the notion that software systems continually change, new software systems are built from the old ones, and . . . all must interoperate and cooperate with each other.

The Changing Nature of Software

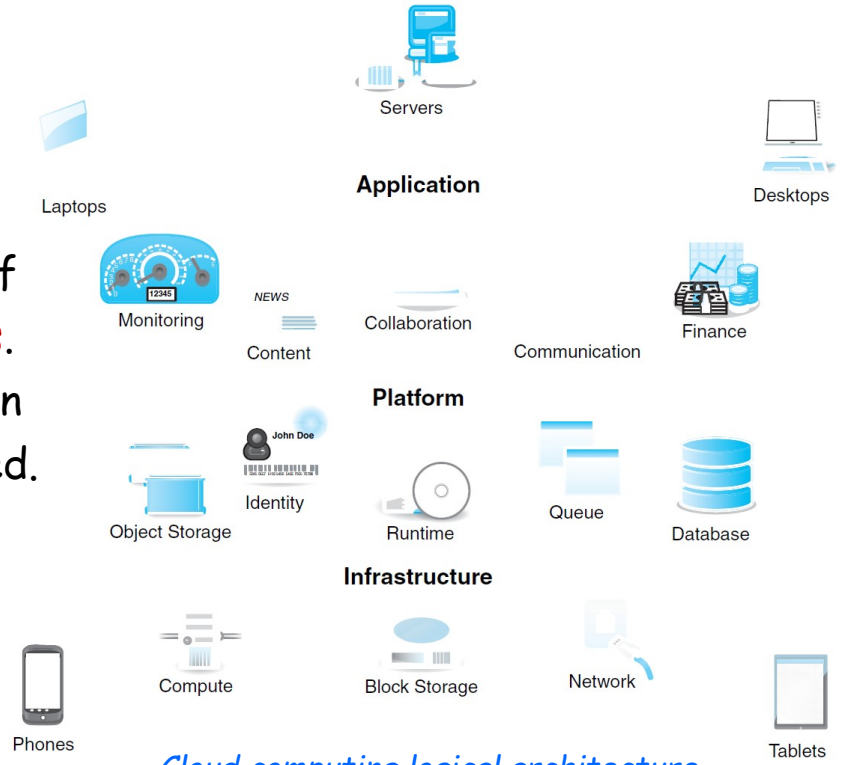
- Four broad categories of software are evolving to dominate the industry
  - WebApps
    - Web-based systems and applications
    - WebApps have evolved into sophisticated computing tools that not only provide stand-alone function to the end user, but also have been integrated with corporate databases and business applications.
  - Mobile Applications
    - The term app has evolved to connote software that has been specifically designed to reside on a mobile platform (e.g., iOS, Android, or Windows Mobile).
    - Mobile applications encompass a user interface that takes advantage of the unique interaction mechanisms provided by the mobile platform, interoperability with Web-based resources that provide access to a wide array of information that is relevant to the app, and local processing capabilities that collect, analyze, and format information in a manner that is best suited to the mobile platform.
    - A mobile app provides persistent storage capabilities within the platform.

The Changing Nature of Software

- Four broad categories of software are evolving to dominate the industry
  - Difference between WebApps and Mobile Applications
    - A mobile web application (WebApp) allows a mobile device to gain access to web-based content via a browser that has been specifically designed to accommodate the strengths and weaknesses of the mobile platform.
    - A mobile app can gain direct access to the hardware characteristics of the device (e.g., accelerometer or GPS location) and then provide the local processing and storage capabilities noted earlier.
    - As time passes, the distinction between mobile WebApps and mobile apps will blur as mobile browsers become more sophisticated and gain access to device level hardware and information.

The Changing Nature of Software

- Four broad categories of software are evolving to dominate the industry
  - *Cloud Computing*
    - Cloud computing encompasses an infrastructure or "ecosystem" that enables any user, anywhere, to use a computing device to share computing resources on a broad scale.
    - The cloud architecture can be segmented to provide access at a variety of different levels from full public access to private cloud architectures accessible only to those with authorization.
    - The implementation of cloud computing requires the development of an architecture that encompasses front-end and back-end services.
      - The front-end includes the client (user) device and the application software (e.g., a browser) that allows the back-end to be accessed.
      - The back-end includes servers and related computing resources, data storage systems (e.g., databases), server-resident applications, and administrative servers that use middleware to coordinate and monitor traffic by establishing a set of protocols for access to the cloud and its resident resources.



Cloud computing logical architecture

The Changing Nature of Software

- Four broad categories of software are evolving to dominate the industry
  - Product Line Software
    - A software product line is defined as "*a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.*" [Software Engineering Institute]
    - The idea that a line of software products, all developed using the same underlying application and data architectures, and all implemented using a set of reusable software components that can be reused across the product line provides significant engineering leverage.
    - A software product line shares a set of assets that include requirements, architecture, design patterns, reusable components, test cases, and other software engineering work products.
    - In essence, a software product line results in the development of many products that are engineered by capitalizing on the commonality among all the products within the product line.

Short Summary

- Four broad categories of software are evolving to dominate the industry
  - Software is the key element in the evolution of computer-based systems and products and one of the most important technologies on the world stage.
  - Over the past 50 years, software has evolved from a specialized problem solving and information analysis tool to an industry in itself.
  - Yet we still have trouble developing high-quality software on time and within budget.