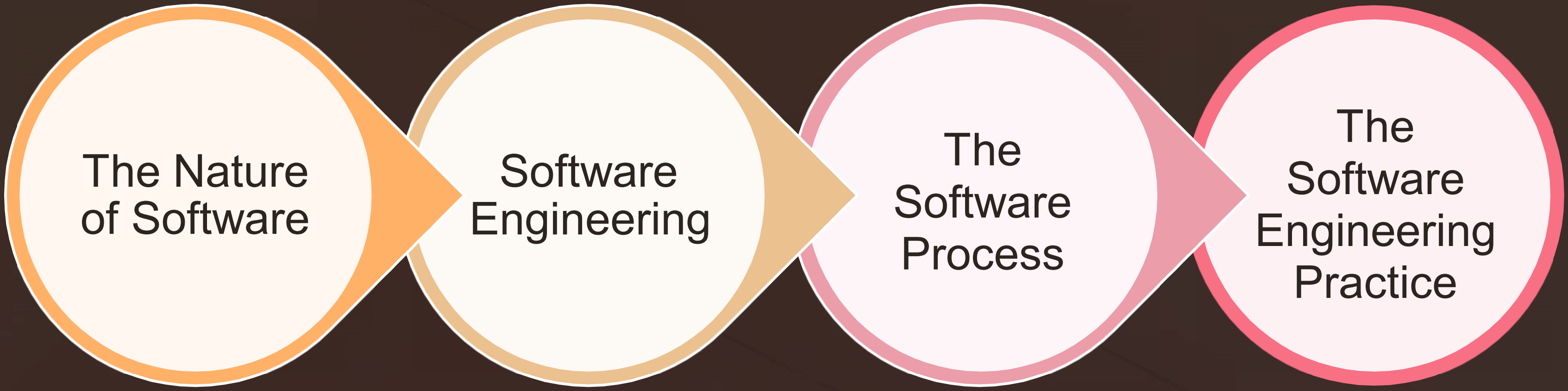
01
Software & Software Engineering

Today's Lesson



The Nature of Software

Software Engineering

The Software Process

The Software Engineering Practice

The Nature of Software

Product

- Transforming, producing, managing, acquiring, displaying or transmitting information

Vehicle for product delivery

- Basis for the control of the computer
- The communication of information
- Creation and control of other programs

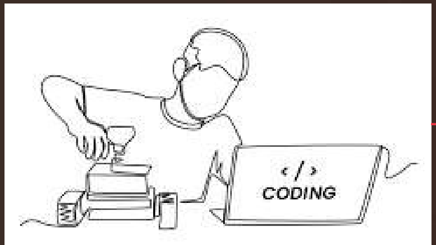
The Nature of Software

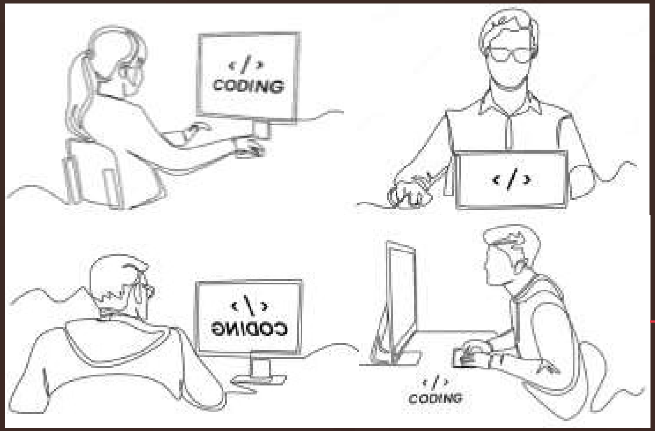
- Software delivers the most important product of our time—information.
  - It transforms personal data so that the data can be more useful in a local context
  - It manages business information to enhance competitiveness
  - It provides a gateway to worldwide information networks
  - It provides the means for acquiring information in all of its forms

The Nature of Software

- The role of computer software has undergone significant change over the last half-century:
  - Hardware performance
  - Computing architectures
  - Memory and storage capacity
  - input and output options
- All these have led to more sophisticated and complex computer-based systems.
- Sophistication and complexity can produce dazzling results when a system succeeds, but they can also pose huge problems for those who must build complex systems.

The Nature of Software

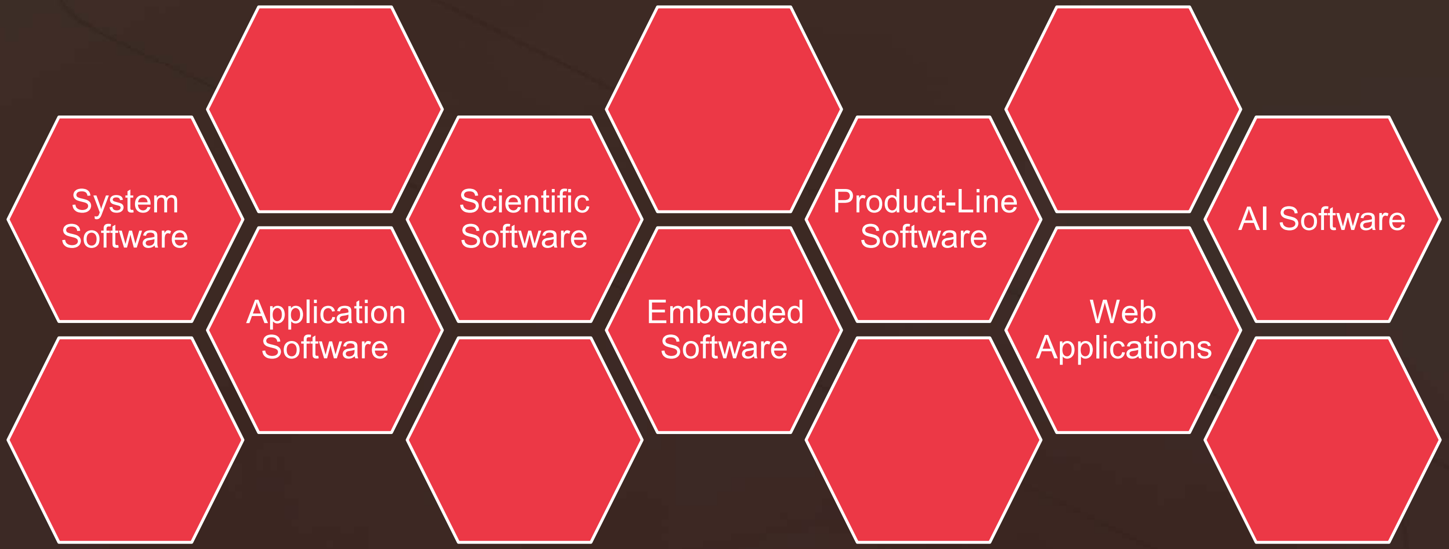




- Why does it take so long to get software finished?
- Why are development costs so high?
- Why can’t we find all errors before we give the software to our customers?
- Why do we spend so much time and effort maintaining existing programs?
- Why do we continue to have difficulty in measuring progress as software is being developed and maintained?

Software Application Domains

Today, seven broad categories of computer software present continuing challenges for software engineers:



System Software

Application Software

Scientific Software

Embedded Software

Product-Line Software

Web Applications

AI Software

Software Application Domains

- Millions of software engineers worldwide are hard at work on software projects in one or more of these categories.
- New systems are built or existing applications enhanced.
- Past generations of software people have left a legacy in each of the categories
- The goal is for legacy left behind by this generation to ease the burden of future software engineers.

New Challenges

Open-World Computing

- Wireless networking
  Challenge to develop software that will allow several devices to communicate across vast networks.

Net-sourcing

- The internet
  The challenge is to develop applications that provide a benefit to targeted end-user markets worldwide

Open-source

- Source-code distribution
  The challenge is to write code that is self-descriptive and develop techniques that will enable stakeholders to track changes

The Uniqueness of Web Apps

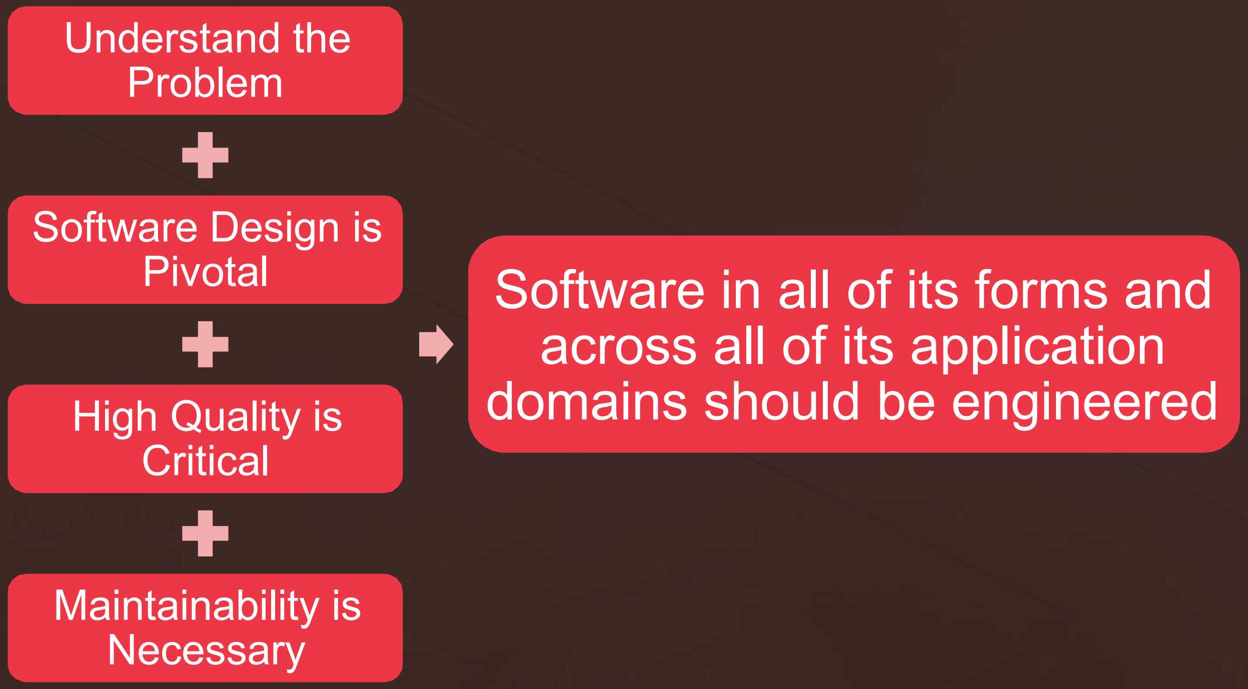
- In the early days of the web, websites consisted of set of linked hypertext files that presented information using text and limited graphics.
- As time passed, the augmentation of HTML by development tools (e.g., XML, Java) enabled Web engineers to provide computing capability along with informational content.
- Web-based systems and applications (WebApps) were born.
- 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.

Common Attributes of Web Apps

- Network intensiveness
- Concurrency
- Unpredictable Loads
- Performance
- Availability
- Data-driven
- Content-sensitive
- Continuous Evolution
- Immediacy
- Security
- Aesthetics

Software Engineering

In order to build software that is ready to meet today’s challenges, one must recognize a few simple realities:



Understand the Problem

+

Software Design is Pivotal

+

High Quality is Critical

+

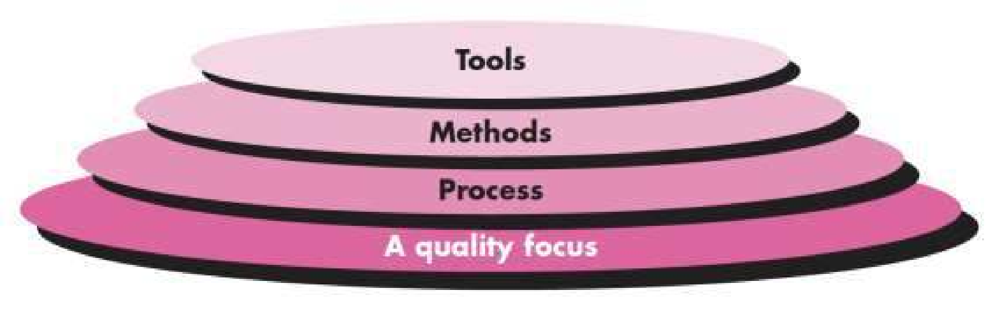
Maintainability is Necessary

Software in all of its forms and across all of its application domains should be engineered

Software Engineering

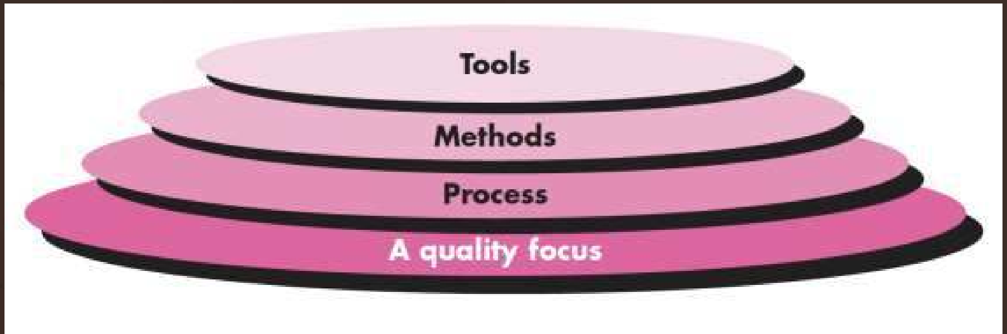
- Software Engineering can be described as the application of a systematic, disciplined and quantifiable approach to the development, operation, and maintenance of software.
  - Simply put, it is, the application of engineering principles to software
- Because these may differ from team to team, adaptability and agility are also required

Software Engineering



- The foundation for software engineering is the process layer.
- The software engineering process is the glue that holds the technology layers together and enables rational and timely development of computer software.
- The bedrock that supports software engineering is a quality focus. There must be total commitment to quality

Software Engineering



- Process defines a framework that must be established for effective delivery of software engineering technology.
- Methods provide the technical how-to's for building software.
- Software engineering tools provide support for the process and the methods

The Software Process

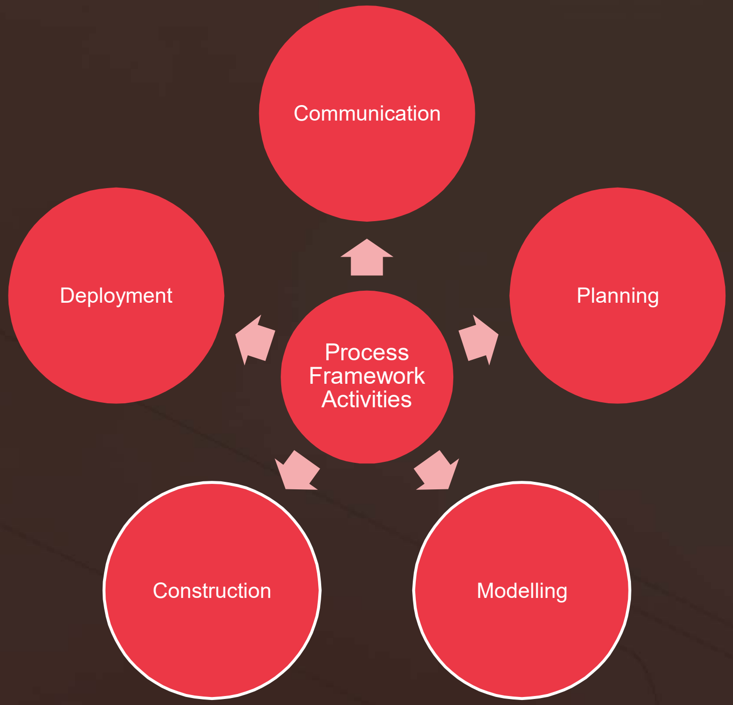
- A process is a collection of activities, actions, and tasks that are performed when some work product is to be created.
  - An activity strives to achieve a broad objective (e.g., communication with stakeholders)
  - An action (e.g., architectural design) encompasses a set of tasks that produce a major work product (e.g., an architectural design model)
  - A task focuses on a small, but well-defined objective that produces a tangible outcome (e.g., conducting a unit test)

The Software Process

- In the context of software engineering, a process is not a rigid prescription for how to build computer software.
- Rather, it is an adaptable approach that enables the software team to pick and choose the appropriate set of work actions and tasks.
- The intent is always to deliver software in a timely manner and with sufficient quality to satisfy the clients and users.

The Process Framework

- It establishes the foundation for a complete software engineering process by identifying a small number of framework activities that are applicable to all software projects, regardless of their size or complexity.



The Process Framework

- The process framework activities are complemented by a number of umbrella activities.
- In general, umbrella activities are applied throughout a software project
- They help a software team manage and control progress, quality, change, and risk.

The Process Framework

Typical umbrella activities include:

- Project tracking and control
- Risk management
- Quality assurance
- Technical reviews
- Measurement
- Configuration management
- Reusability management
- Work product preparation and production

The Software Engineering Practice

The essence of practice

- The essence of problem solving, and consequently, the essence of software engineering practice:
  1. Understand the problem (communication and analysis).
  2. Plan a solution (modelling and software design).
  3. Carry out the plan (code generation).
  4. Examine the result for accuracy (testing and quality assurance).

The Essence of Practice

Understand the Problem

Who has a stake in the solution to the problem?

What are the unknowns?

Can the problem be compartmentalized?

Can the problem be represented graphically?

The Essence of Practice

Plan the Solution

Have you seen similar problems before?

Has a similar problem been solved?

Can subproblems be defined?

Can you represent a solution in a manner that leads to effective implementation?

The Essence of Practice

Carry out the plan

Does the solution conform to the plan?

Is each component part of the solution provably correct?

The Essence of Practice

Examine the Result

Is it possible to test each component part of the solution?

Does the solution produce results that conform to the data, functions, and features that are required?

General Principles

1	2	3	4	5	6	7
• The reason it all exists - value for the users	• Keep it simple	• Maintain the vision	• Others will use what you produce	• Be open to the future	• Plan ahead for reuse	• Think!

Exercise

1. Provide a number of examples (both positive and negative) that indicate the impact of software on our society.
2. As software becomes more pervasive, risks to the public (due to faulty programs) become an increasingly significant concern. Develop a doomsday but realistic scenario in which the failure of a computer program could do great harm (either economic or human).

Any Questions?

The End