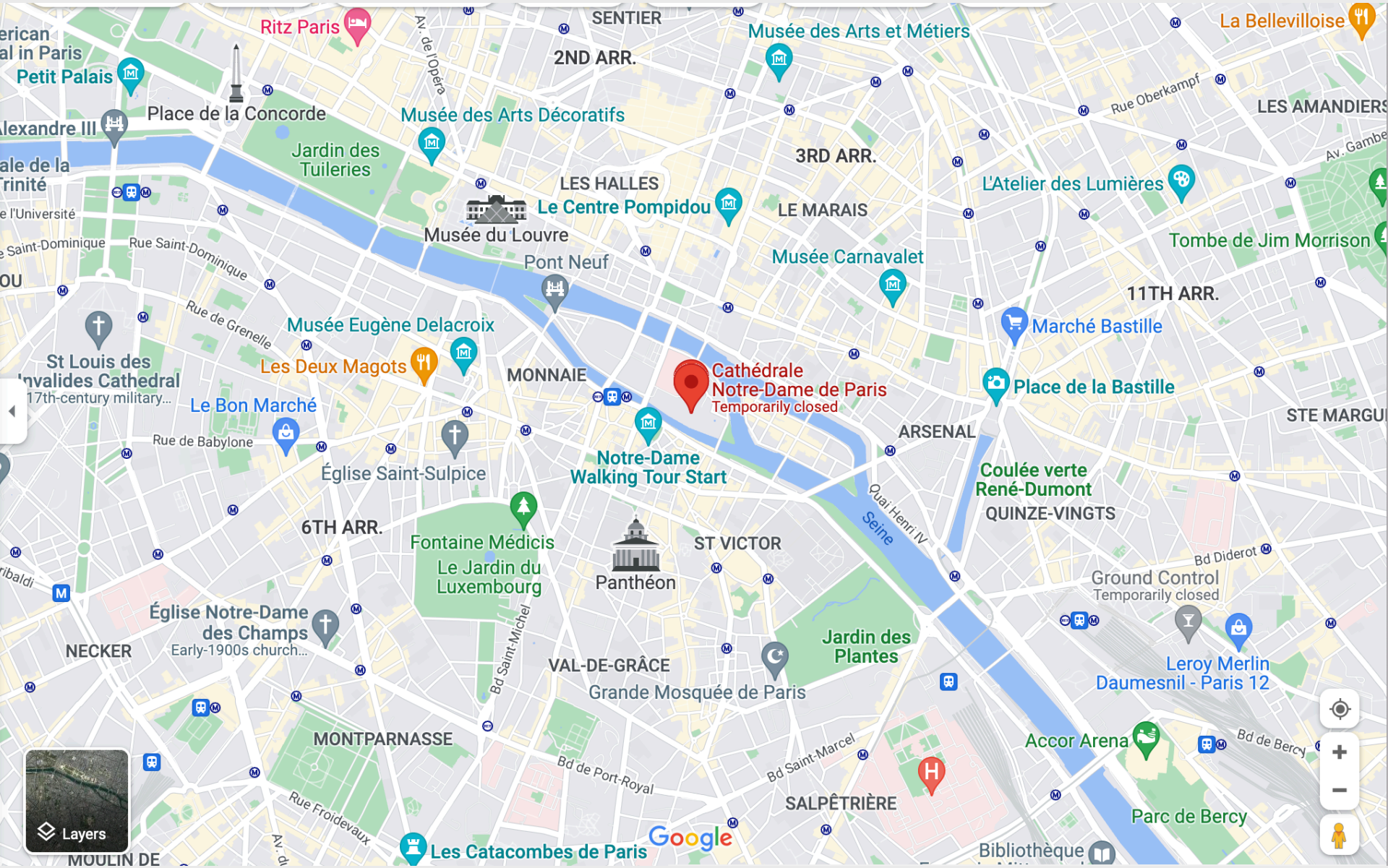
Software and Software Engineering

Chapter 1 (Pressman 9E)

CSCI-P465/565

Luddy School of
Informatics, Computing and Engineering


Birmingham
Cambridge
WALES
ENGLAND
Oxford
London
Bristol
Cardiff
Southampton
Brighton
Plymouth
English Channel
Amsterdam
The Hague
Netherlands
Rotterdam
Hanover
Magdeburg
Dortmund
Essen
Leipzig
Antwerp
Düsseldorf
Germany
Brussels
Cologne
Lille
Belgium
Frankfurt
Luxembourg
Mannheim
Nuremberg
Guernsey
Jersey
Rouen
Caen
Cathédrale
Notre-Dame de Paris
Temporarily closed
Karlsruhe
Stuttgart
Regensburg
Strasbourg
Brest
Rennes
Munich
Salzburg
Angers
Tours
Nantes
Basel
Zürich
Liechtenstein
Switzerland
France
Lausanne
La Rochelle
Geneva
Clermont-Ferrand
Lyon
Milan
Verona
Padua
Grenoble
Bay of Biscay
Bordeaux
Genoa
Bologna
Pisa
Layers
Google


Ritz Paris
SENTIER
Musée des Arts et Métiers
La Bellevilloise
Petit Palais
2ND ARR.
Place de la Concorde
Musée des Arts Décoratifs
LES AMANDIERS
Jardin des Tuileries
3RD ARR.
LES HALLES
Le Centre Pompidou
L'Atelier des Lumières
LE MARAIS
Musée du Louvre
Tombe de Jim Morrison
Pont Neuf
Musée Carnavalet
11TH ARR.
Marché Bastille
Musée Eugène Delacroix
St Louis des Invalides Cathedral
17th-century military...
Les Deux Magots
MONNAIE
Cathédrale Notre-Dame de Paris
Temporarily closed
Place de la Bastille
Le Bon Marché
ARSENAL
Église Saint-Sulpice
Notre-Dame Walking Tour Start
Coulée verte René-Dumont
QUINZE-VINGTS
6TH ARR.
Fontaine Médicis
Le Jardin du Luxembourg
ST VICTOR
Panthéon
Seine
Ground Control
Temporarily closed
Église Notre-Dame des Champs
Early-1900s church...
NECKER
Jardin des Plantes
Leroy Merlin Daumesnil - Paris 12
VAL-DE-GRÂCE
Grande Mosquée de Paris
MONTPARNASSE
Accor Arena
SALPÊTRIÈRE
Parc de Bercy
Les Catacombes de Paris
Bibliothèque
Google
Layers

Notre-Dame de Paris

- It is widely considered to be one of the finest examples of French Gothic architecture in the world.
- The cathedral was built in the 12th and 13th centuries.
- It is also home to many important religious and historical artifacts, including the Crown of Thorns, which is said to have been worn by Jesus Christ during his crucifixion.
- The cathedral is a popular tourist attraction, and it is visited by millions of people from all over the world each year.

Notre-Dame de Paris

- On April 15, 2019, a fire started at Notre-Dame cathedral.
- Within an hour, its famous 750 ton spire collapsed.
- The cause of the fire is still unknown.
- However, we do know something about the design of its fire alarm system…

The fire

- The New York Times: "The fire warning system at Notre-Dame took dozens of experts six years to put together, and in the end involved thousands of pages of diagrams, maps, spreadsheets and contracts."
- The fire alarm system was a hugely expensive, highly sophisticated, complex system. It was also badly designed.

The fire

- At 6:18 p.m. on the night of April 15, an inexperienced security employee, working an extra shift to cover for a co-worker, saw a warning on the fire safety system. It first told him what quadrant of the building might have a fire in it, ‘Attic Nave Sacristy’, and then a code:

- The code referred to a specific detection device, but there was no way the guard could have known how to use this code to locate the fire.

The fire

- The system wasn’t designed to make this easy to understand.
- And his job wasn’t designed with the training to close that knowledge gap.

The fire

- He did call the guard inside the church and asked him to investigate. The problem was that there were two attics in the cathedral, and the church guard went to the wrong one.
- It took 25 minutes, as the fire spread quietly hundreds of feet above their heads, for them to realize their mistake. By the time the church guard climbed the three hundred steps to the main attic, the fire was raging out of control.
- They finally called the fire department, but the damage had already been done.

The fire

- The tower collapsed in 60 minutes.

The fire

- “We like to think that, here in the present day, with almost nine hundred years of technological progress from the time the Notre-Dame cathedral was designed, failures like this would be impossible. The truth is that **designing things well isn’t easy to do**.” (Scott Berkun more in his book *How Design Makes the World*)

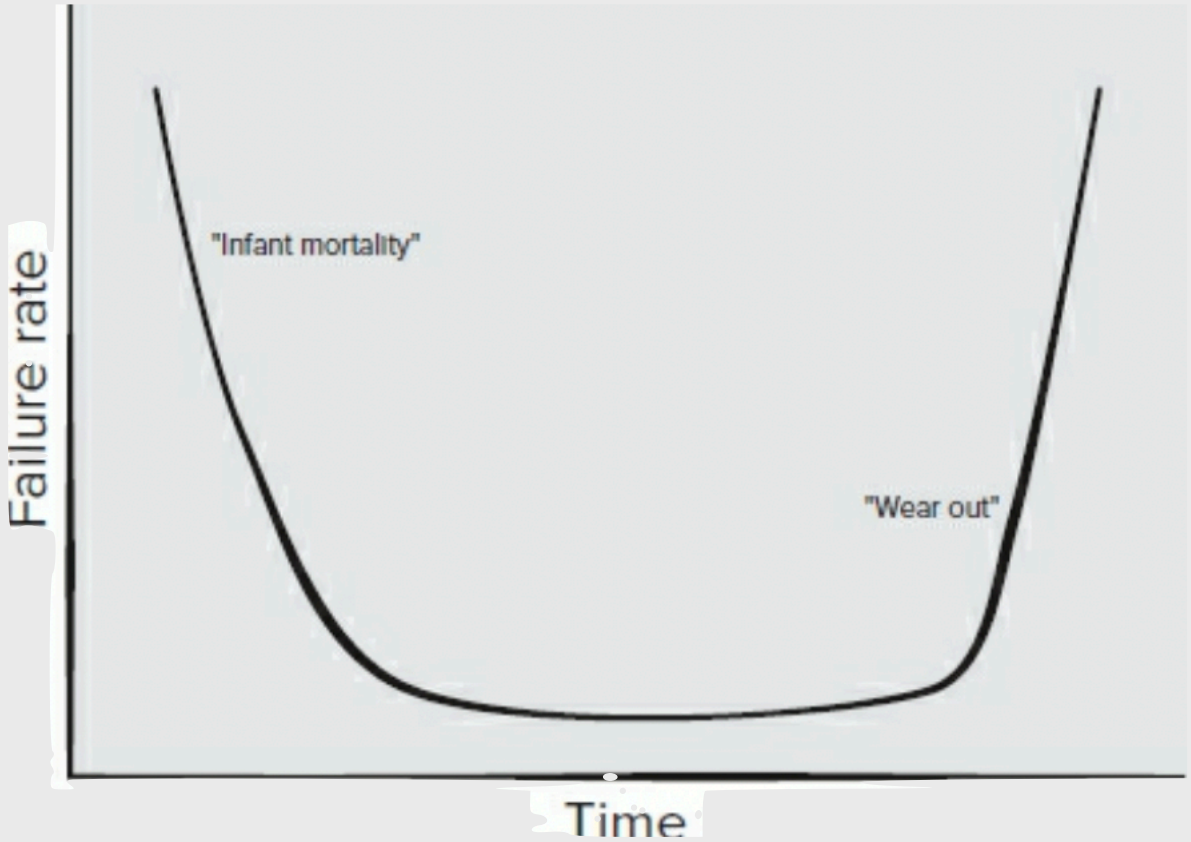
What is Software?

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) documentation that describes the operation and use of the programs.*

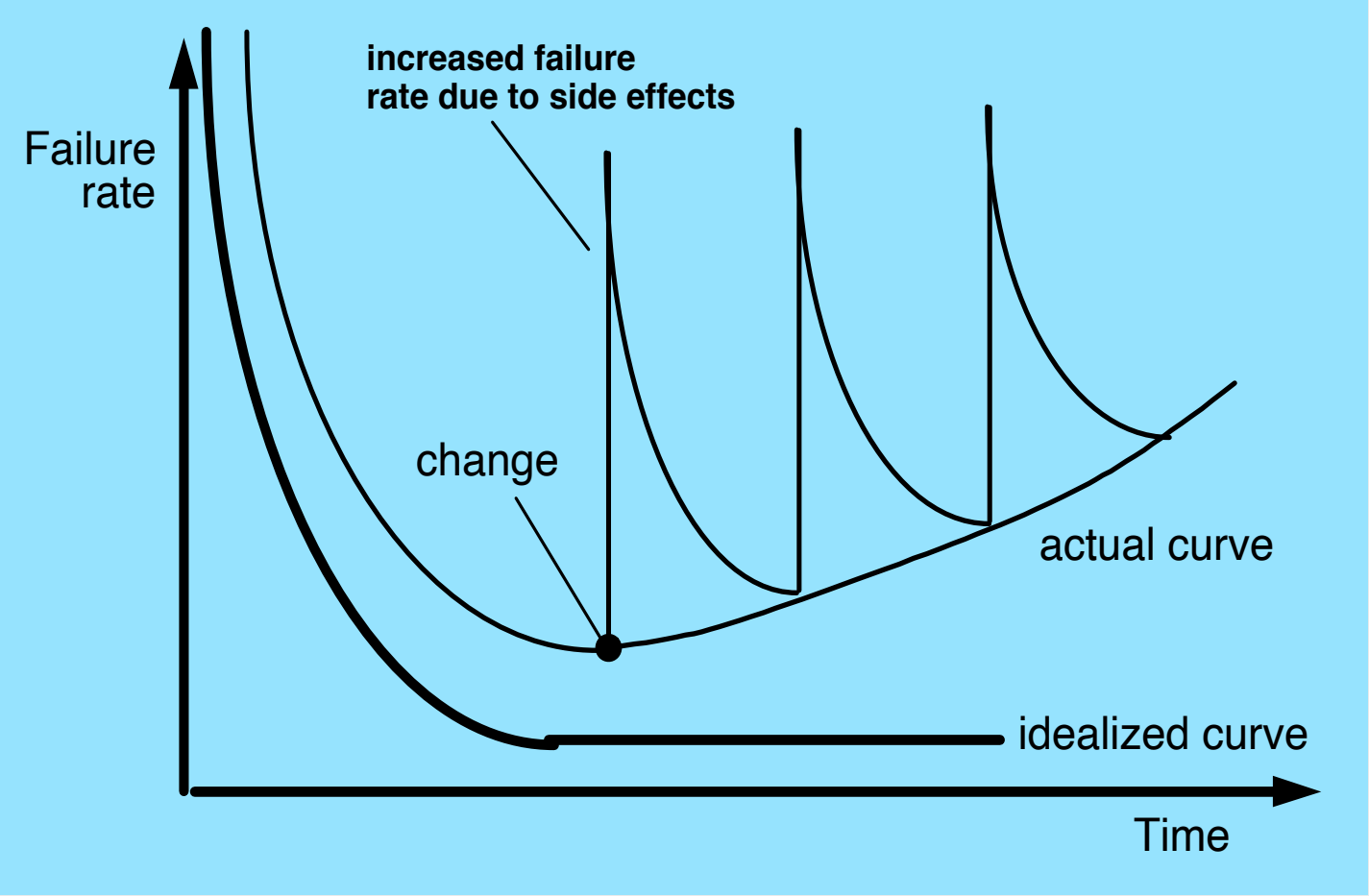
What is Software?

- Software is developed or engineered, it is not manufactured in the classical sense.
- Software is a logical rather than a physical system.
- Software doesn't "wear out".
- Although the industry is moving toward component-based construction, most software continues to be custom-built.

Failure curve for hardware



Failure curves for software



Software Application Domains

- System software
  - Programs written to service other programs (compilers, OS components and drivers, networking)
- Application software
  - Stand-alone programs that solve a specific business need

- Engineering/Scientific software
  - A broad array of “number-crunching” or data science programs that range from astronomy to volcanology, from automotive stress analysis to orbital dynamics, from computer-aided design to consumer spending habits, and from genetic analysis to meteorology.

Software Application Domains

- 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(key pad control for a microwave oven).

- Product-line software (mass production)
  - Composed of reusable components and designed to provide specific capabilities. It may focus on a limited marketplace (e.g., inventory control products) or attempt to address the mass consumer market.

Software Application Domains

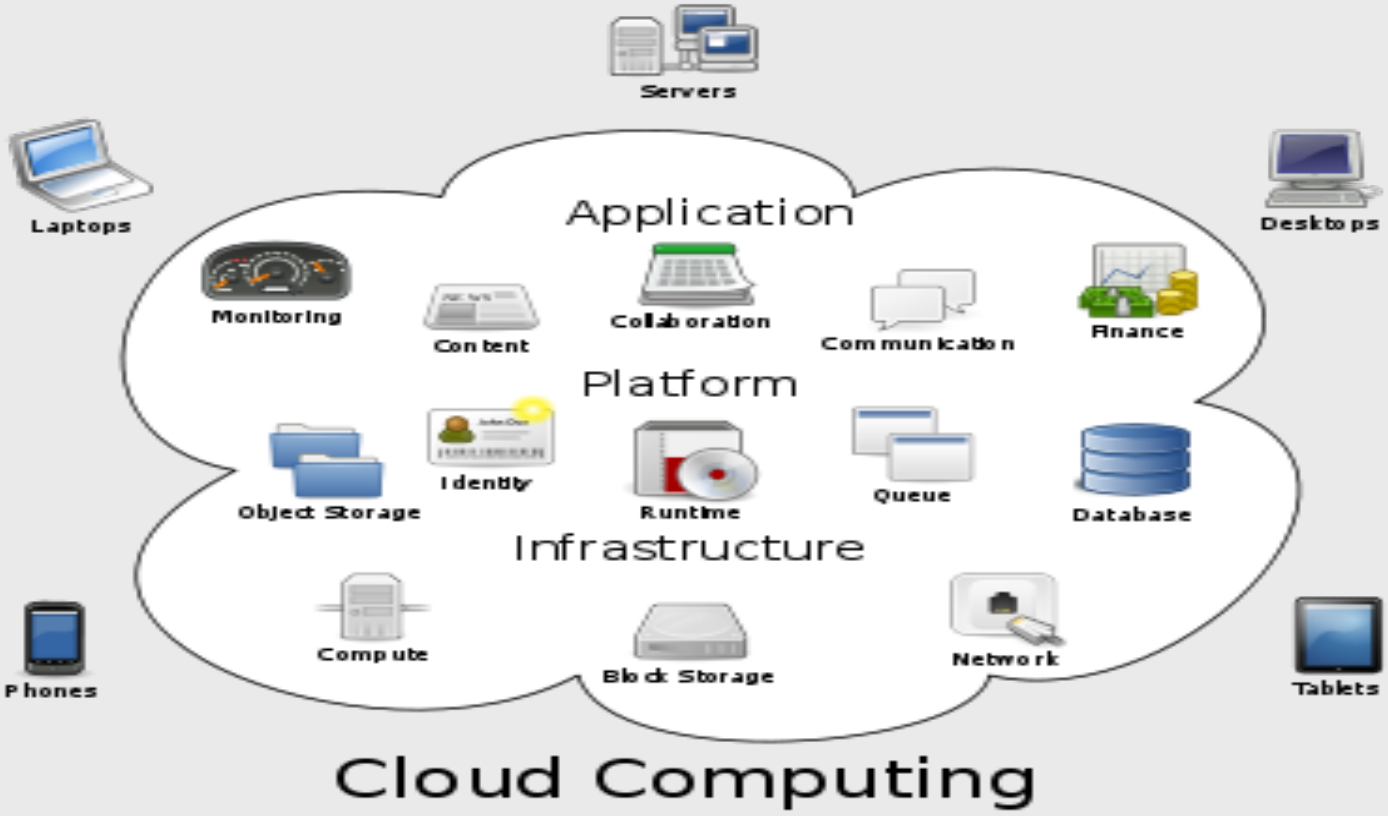
- Web/Mobile applications
  - Spans a wide array of applications and encompasses browser-based apps, cloud computing, service-based computing, and software that resides on mobile devices.
- AI software
  - Makes use of heuristics to solve complex problems that are not amenable to regular computation or straightforward analysis(robotics, decision-making systems, pattern recognition (image and voice), machine learning, theorem proving, and game playing).

Legacy Software

Why must it change?

- software must be adapted to meet the needs of new computing environments or technology.
- software must be enhanced to implement new business requirements.
- software must be extended to make it interoperable with other more modern systems or databases.
- software must be re-architected to make it viable within a network environment.

Cloud Computing



Software Engineering

- Some realities:
  - *a concerted effort should be made to understand the problem before a software solution is developed*
  - *design becomes a pivotal activity*
  - *software should exhibit high quality*
  - *software should be maintainable*

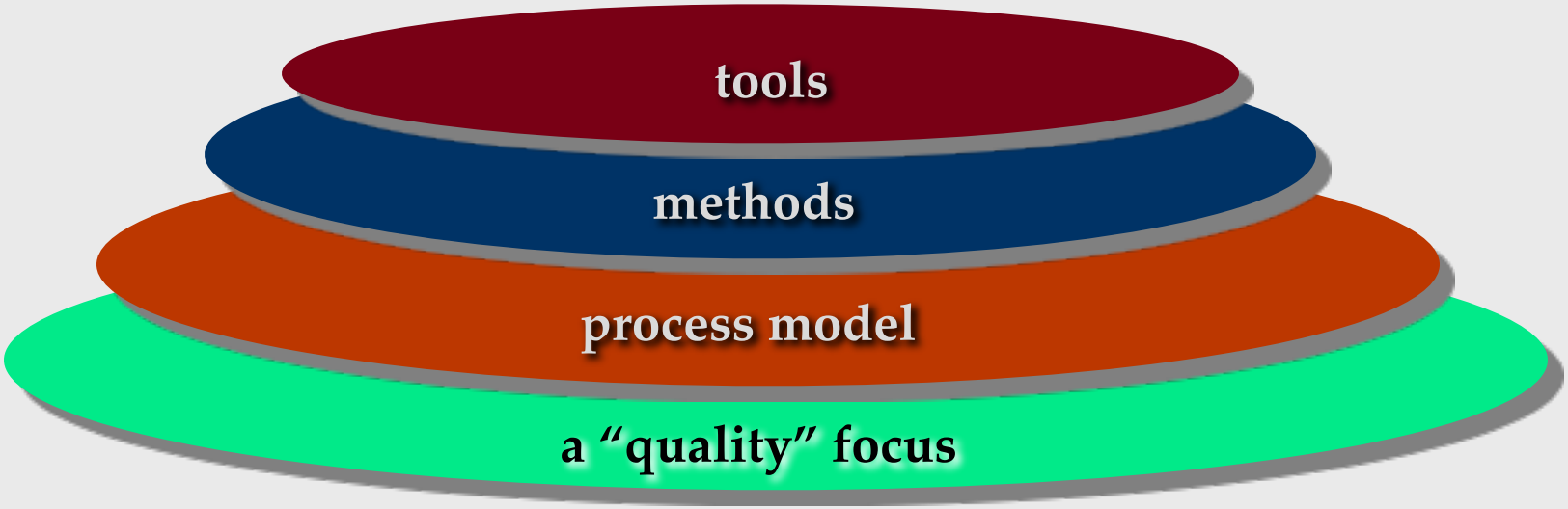
Software Engineering

- The seminal definition:
  - *[Software engineering is] the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines.*

DEFINING THE DISCIPLINE

- The IEEE definition:
  - *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).*

A Layered Technology



Software Engineering

A Layered Technology

- Any engineering approach (including software engineering) must rest on an organizational commitment to quality. Total quality management (TQM) or Six Sigma, and similar philosophies foster a culture of continuous process improvement.
- The software process establishes the context in which technical methods are applied, work products (models, documents, data, reports, forms, etc.) are produced, milestones are established, quality is ensured, and change is properly managed.

The Software Process

- A process is a collection of *activities*, *actions*, and *tasks* to create a product.
- Activity: strives to achieve a broad objective
  - Example: communication with stakeholders
- Action: aims to produce a major work product
  - Example: architectural design
- Task: focuses on a small, well-defined objective with a tangible outcome
  - Example: conducting a unit test

A Process Framework

Software Engineering Process

Framework activities

work tasks

work products

milestones & deliverables

QA checkpoints

Umbrella Activities

Framework Activities

- Communication
- Planning
- Modeling
  - Analysis of requirements
  - Design
- Construction
  - Code generation
  - Testing
- Deployment

- Software Cost:
  - 60% development, 40% Testing

Framework Activities

- Communication
  - To understand stakeholders' objectives for the project and to gather requirements that help define software features and functions.
- Planning
  - A software project is a complicated journey, and the planning activity creates a "map" that helps guide the team as it makes the journey.

Stake and Stakeholder

- A stake means an amount of money or something else of value that is placed in a bet.
- A stakeholder is anyone who has a stake in the successful outcome of the project: business managers, end users, support people, et cetera.

Framework Activities

- Modeling
  - To better understand software requirements and the design that will achieve those requirements.
- Construction
  - Code generation
  - Testing
- Deployment

- Software Cost:
  - 60% development, 40% Testing

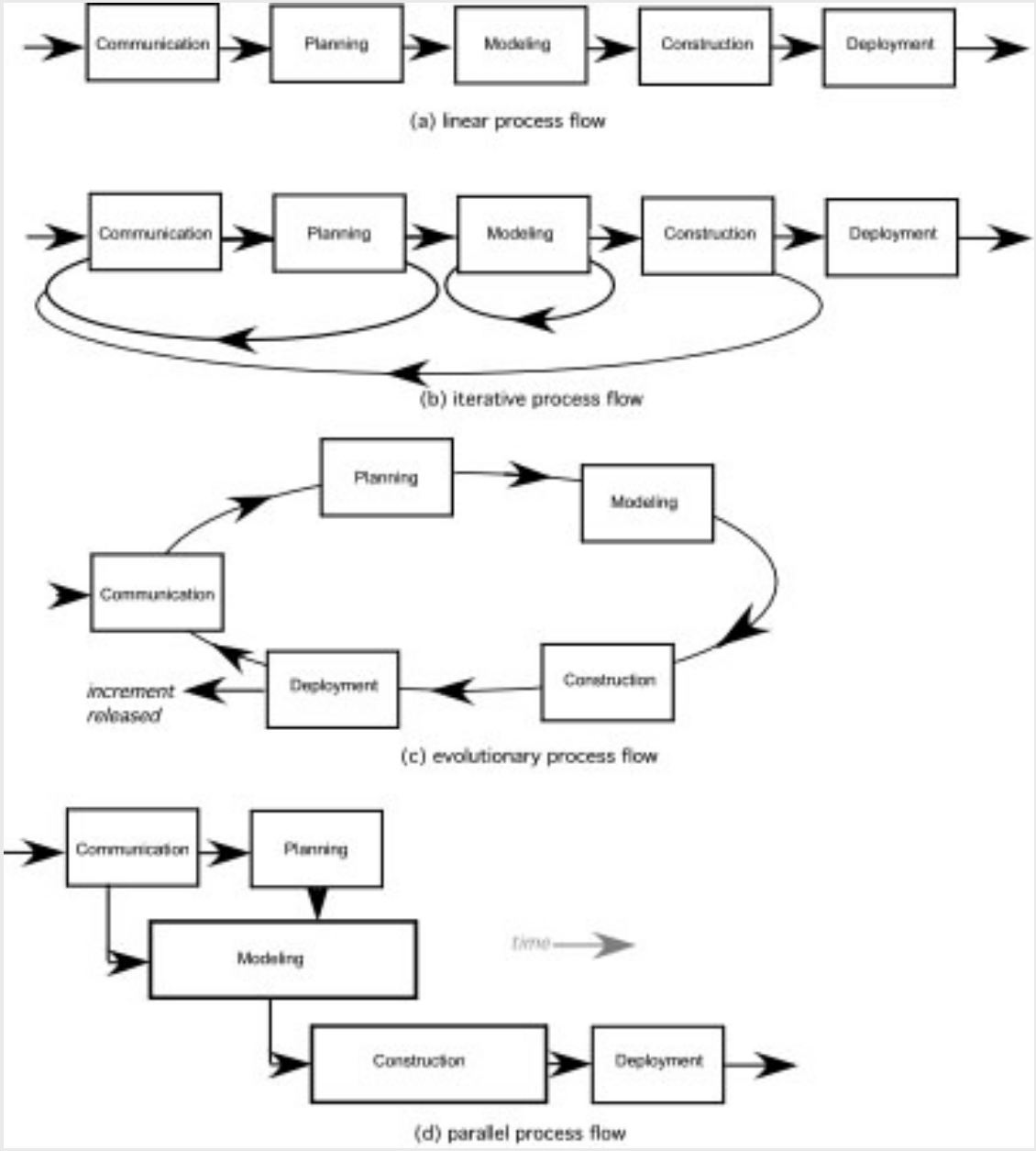
Umbrella Activities

- Software project tracking and control
  - To assess progress against the project plan and take any necessary action to maintain the schedule.
- Risk management
  - Assesses risks that may affect the outcome of the project or the quality of the product.
- Software quality assurance
- Technical reviews
  - To uncover and remove errors before they are propagated to the next activity.

Umbrella Activities

- Measurement
  - Defines and collects process, project, and product measures that assist the team in delivering software that meets stakeholders' needs.
- Software configuration management
  - Manages the effects of change throughout the software process.
- Reusability management
  - Establishes mechanisms to achieve reusable components.
- Work product preparation and production
  - Encompasses the activities required to create work products such as models, documents, logs, forms, and lists.

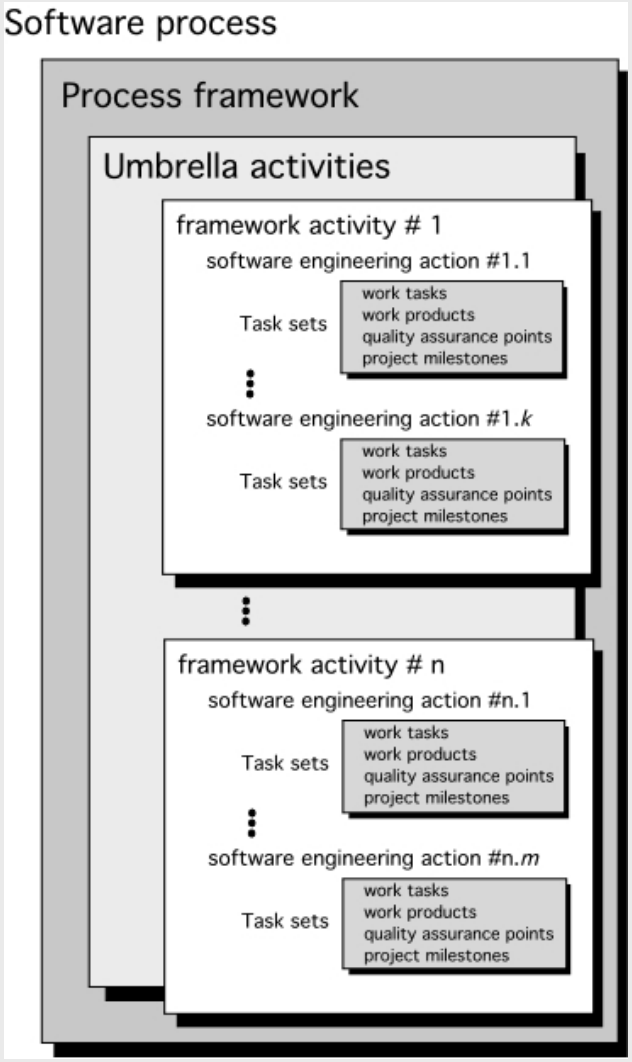
Process Flow



Adapting a Process Model

- the overall flow of activities, actions, and tasks and the interdependencies among them
- the degree to which actions, tasks, products, process are defined
- the manner which other activities are applied
- the level of autonomy given to the software team
- the degree to which team organization and roles are prescribed

A Generic Process Model



Identifying a Task Set

- A task set defines the actual work to be done to accomplish the objectives of a software engineering action.
  - A list of the tasks to be accomplished
  - A list of the work products to be produced
  - A list of the quality assurance filters to be applied

Example (for a small project)

- Activity: communication
  - Action: phone conversation
    - Task set:
      - Make contact with stakeholders
      - Discuss requirements and develop notes
      - Organize notes into a brief written statement of requirements
      - Email to stakeholders for review and approval

Example (for a more complex project)

- Activity: communication
  - Action: elicitation
    - Task set:
      - Make a list of stakeholders
      - Interview each stakeholder to determine wants and needs
      - Build a preliminary list of functions and features based on the stakeholder input
      - Schedule a series of facilitated application specification meetings
      - Produce informal user scenarios and refine them based on stakeholder feedback
      - Prioritize requirements
      - Package requirements for incremental delivery
      - Note constraints and restrictions that will be placed on the system
      - Discuss methods for validating the system

Process assessment and improvement

- The process itself must be assessed to ensure it meets the criteria
- The criteria for a successful process: Chapter 17

The Essence of Practice

- Polya (in his classic book *How to solve It*):

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

But there are also Exceptions

- Srinivasa Ramanujan

Hooker's General Principles

- 1: *The Reason It All Exists*
  - *A software system exists for one reason: to provide value to its users. Always ask "Does this add real value to the system?"*
- 2: *KISS (Keep It Simple, Stupid!)*
  - *The more elegant designs are usually the simpler ones.*
- 3: *Maintain the Vision*
  - *Without conceptual integrity, a system threatens to become a patchwork of incompatible designs, held together by the wrong kind of screws …*
  - *Have an empowered architect who can hold the vision and enforce compliance*

Hooker's General Principles

- 4: *What You Produce, Others Will Consume*
  - Specify with an eye to the users. Design, keeping the implementers in mind. Code with concern for those that must maintain and extend the system.
- 5: *Be Open to the Future*
  - *Systems must be ready to adapt to these and other changes.* Always ask "what if" and prepare for all possible answers.
- 6: *Plan Ahead for Reuse*
  - *Achieving a high level of reuse is one of the hardest goal to accomplish in developing a software system.*
- 7: *Think!*
  - *Placing clear, complete thought before action almost always produces better results.*

How a project starts

- The case of SafeHome!