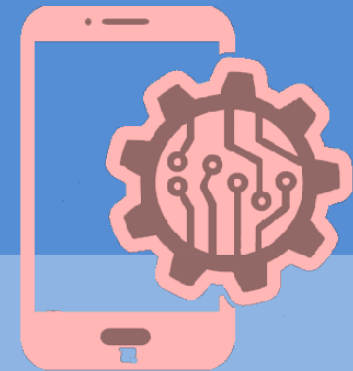


Lecture 1.2 Software process model

Introduction to Software Engineering



Software Process



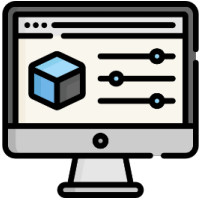


- ❑ A **process** is a collection of activities, actions and tasks that are performed when some work product is to be created
- ❑ A process is not a **rigid prescription** for how to build the software
- ❑ Rather it is **adaptable approach** that enables the people doing the work to **pick** and **choose** the **appropriate set of work actions** and tasks
- ❑ The **purpose** of software process is
 - to deliver software in timely manner and
 - within sufficient quality to satisfy those
 - Who has given proposal for software development and
 - Those who will use software

A **software process** (also known as **software methodology**) is a set of **related activities** that leads to the **production of the software**. These activities may involve the **development** of the **software** from the scratch (**Bespoke**), or, modifying an existing system.

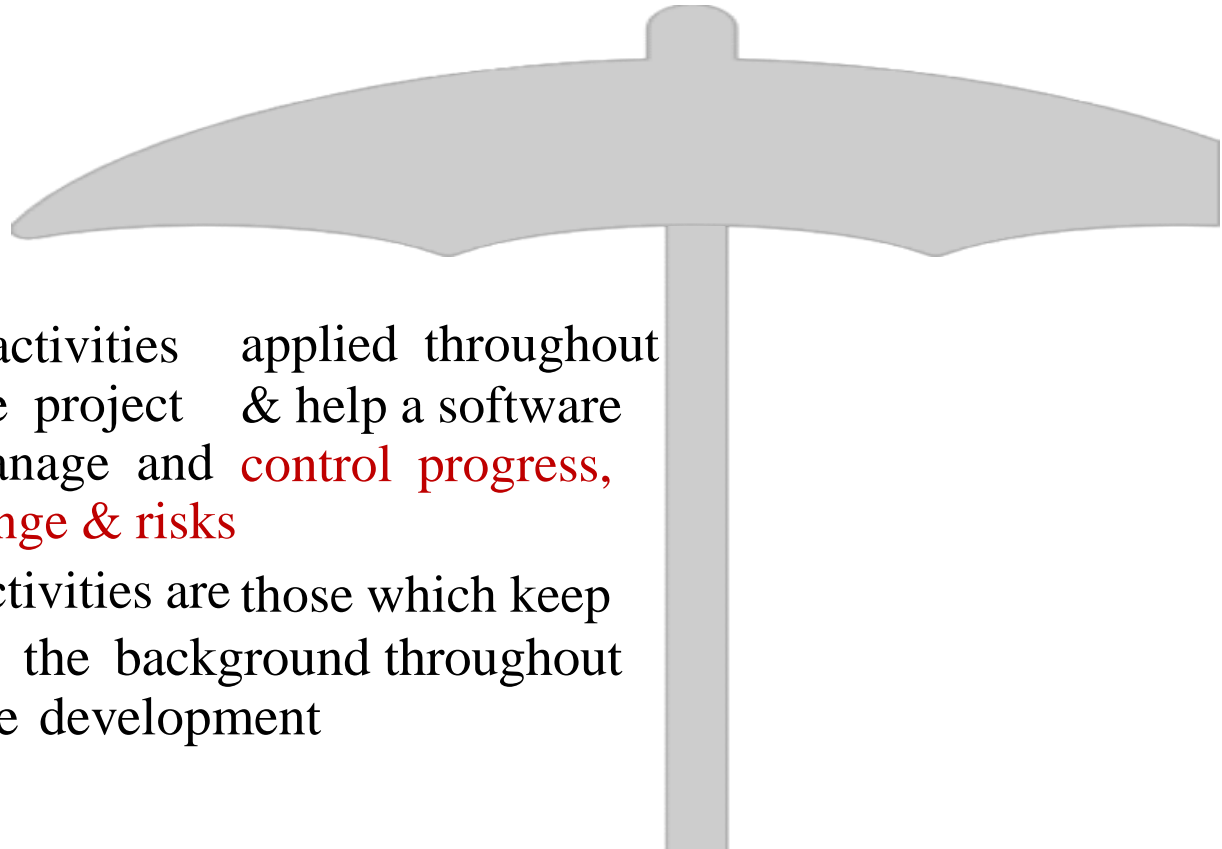
Framework

- A **framework** is a Standard way to build and deploy applications.
- **Software Process Framework** is a foundation of complete software engineering process. It also includes number of framework activities that are applicable to all software projects.

Process Framework Activities

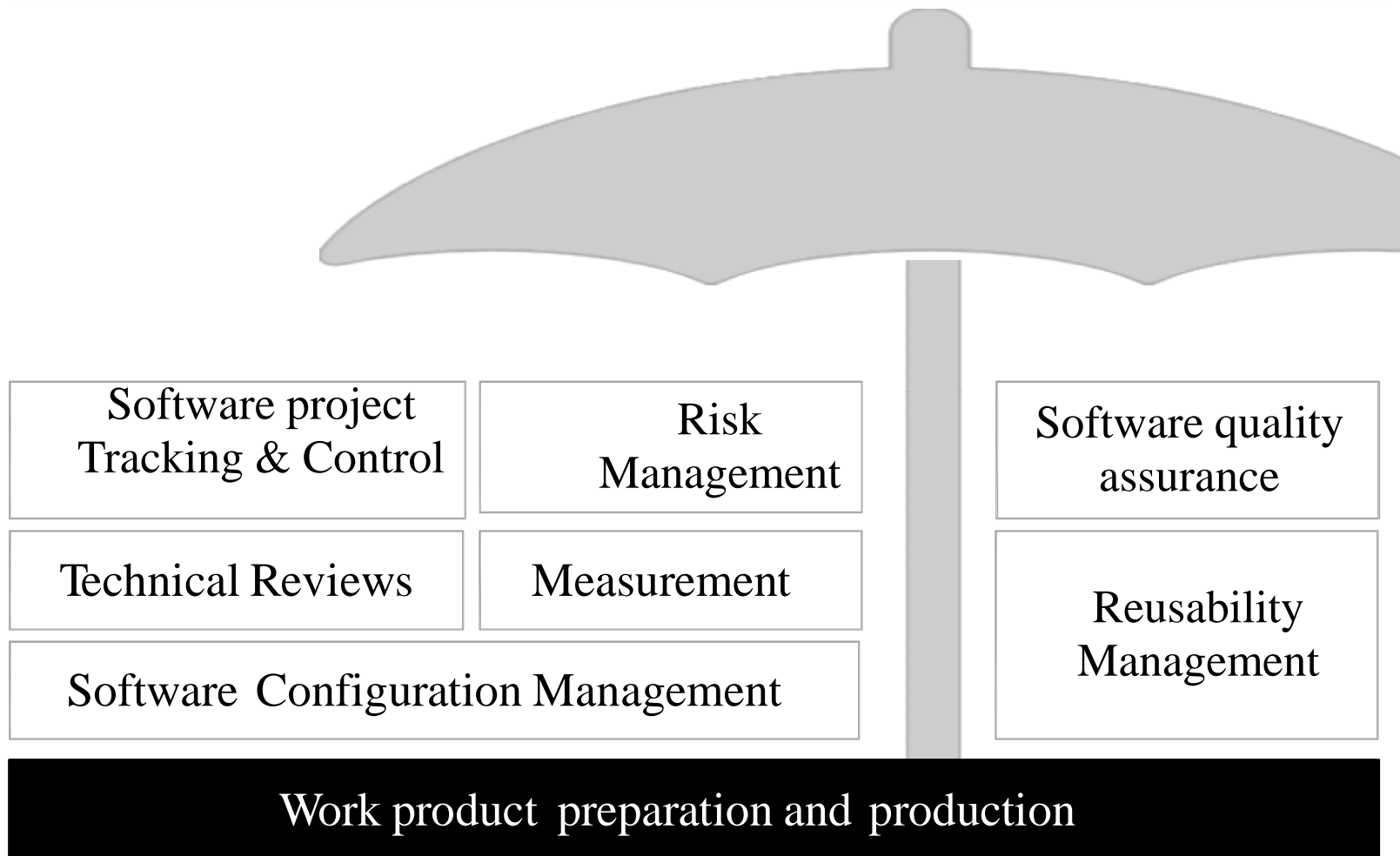
Communication	 <p>1</p> <p>Communication with Customers / stack holders to understand project requirements for defining software features</p>	Planning	 <p>2</p> <p>Software Project Plan which defines workflow that is to follow. It describes technical task, risks, resources, product to be produced & work schedule</p>
Modeling	 <p>3</p> <p>Creating models to understand requirements and shows design of software to achieve requirements</p>	Construction	 <p>4</p> <p>Code Generation (manual or automated) & Testing (to uncover errors in the code)</p>
Deployment	 <p>5</p>	<p>Deliver Software to Customer Collect feedback from customer based on evaluation Software Support</p>	

Umbrella Activities



- Umbrella activities applied throughout the software project & help a software team to manage and **control progress, quality, change & risks**
- Umbrella activities are those which keep running in the background throughout the software development

Umbrella Activities



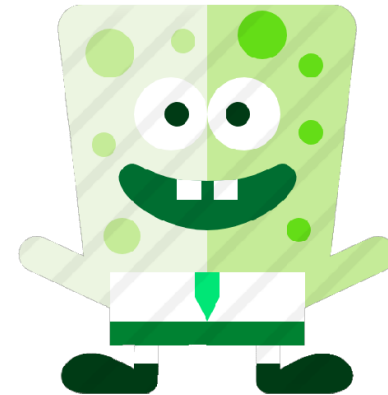
Umbrella Activities

- **Software Project Tracking and Control**
 - Assessing progress against the project plan.
 - Take adequate action to maintain schedule.
- **Formal Technical Reviews**
 - Assessing software work products in an effort to uncover and remove errors before goes into next action or activity.
- **Software Quality Assurance**
 - Define and conducts the activities required to ensure software quality.

Umbrella Activities Cont.

- **Software configuration management:** is the task of tracking and controlling changes in the software
- **Reusability Management:** It defines criteria for work products reuse and establishes mechanism to achieve reusable component.
- **Risk management:** Assesses risks that may effect the outcome of project or quality of product (i.e. software)
- **Measurement:**
 - Define and collects process, project, and product measures
 - Assist the team in delivering software that meets customer's needs.

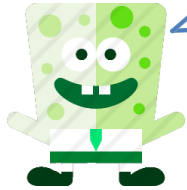
Software Myths



- ☐ Beliefs about software and the process used to build it.
- ☐ “**Misleading Attitudes that cause serious problem**” are myths.
 - Management Myths
 - Customer Myths
 - Practitioner's (Developer) Myths

Myth: A widely held but false belief or idea.

Management Myth - 1

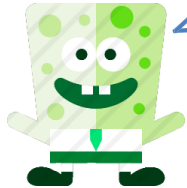


We **have standards and procedures** to build a system, which is enough.

□ Reality:

- Are software practitioners aware of standard's existence?
- Does it reflect modern software engineering practice?
- Is it complete?
- Is it streamlined to improve time to delivery while still maintaining a focus on quality?
- In many cases, the answer to all of these questions is "**NO**."

Management Myth - 2



We have **the latest computers and development tools**

□ Reality:

- It takes much more than the latest model computers to do high-quality software development.
- Computer-aided software engineering (CASE) tools are more important than hardware.

Management Myth - 3



We **can add more programmers** and can catch up the schedule.

□ Reality:

- Software development is not a mechanicle process like manufacturing.
- In the words of Fred Brooks : "adding people to a late software project makes it later."
- People who were working must spend time educating the newcomers.
- People can be added but only in a planned and well-coordinated manner.

Management Myth - 4



I outsourced the development activity, now I can relax and can wait for the final working product.

□ Reality:

- If an organization does not understand how to manage and control software projects internally, it will invariably struggle when it outsources software projects.

Customer Myth - 1



A **general statement of objectives** (requirements) is **sufficient** to start a development.

□ Reality:

- Comprehensive (detailed) statements of requirements is not always possible, an ambiguous (unclear) “statement of objectives” can lead to disaster.
- Unambiguous (clear) requirements can be gathered only through effective and continuous communication between customer and developer.

Customer Myth - 2



Requirement Changes can be easily accommodated because software is very flexible.

□ Reality:

- It is true that software requirements change, but the impact of change varies with the time at which it is introduced.
- When requirements changes are requested early the cost impact is relatively small.

Practitioner's (Developer) Myth - 1



Once we write the program, our job is done.

□ Reality:

- Experts say "the sooner you begin 'writing code', the longer it will take you to get done."
- Industry data indicates that 60 to 80 % effort expended on software will be after it is delivered to the customer for the first time.

Practitioner's (Developer) Myth - 2



I can't access quality until it is running.

□ Reality:

- One of the most effective software quality assurance mechanisms can be applied from the beginning of a project - the technical review.
- Software reviews are more effective “quality filter” than testing for finding software defects.

Practitioner's (Developer) Myth - 3



Working **program** is the **only deliverable** work **product**.

□ Reality:

- A working program is only one part of a software configuration.
- A variety of work products (e.g., models, documents, plans) provide a foundation for successful engineering and, more important, guidance for software support.

Practitioner's (Developer) Myth - 4



Software engineering is about unnecessary documentation.

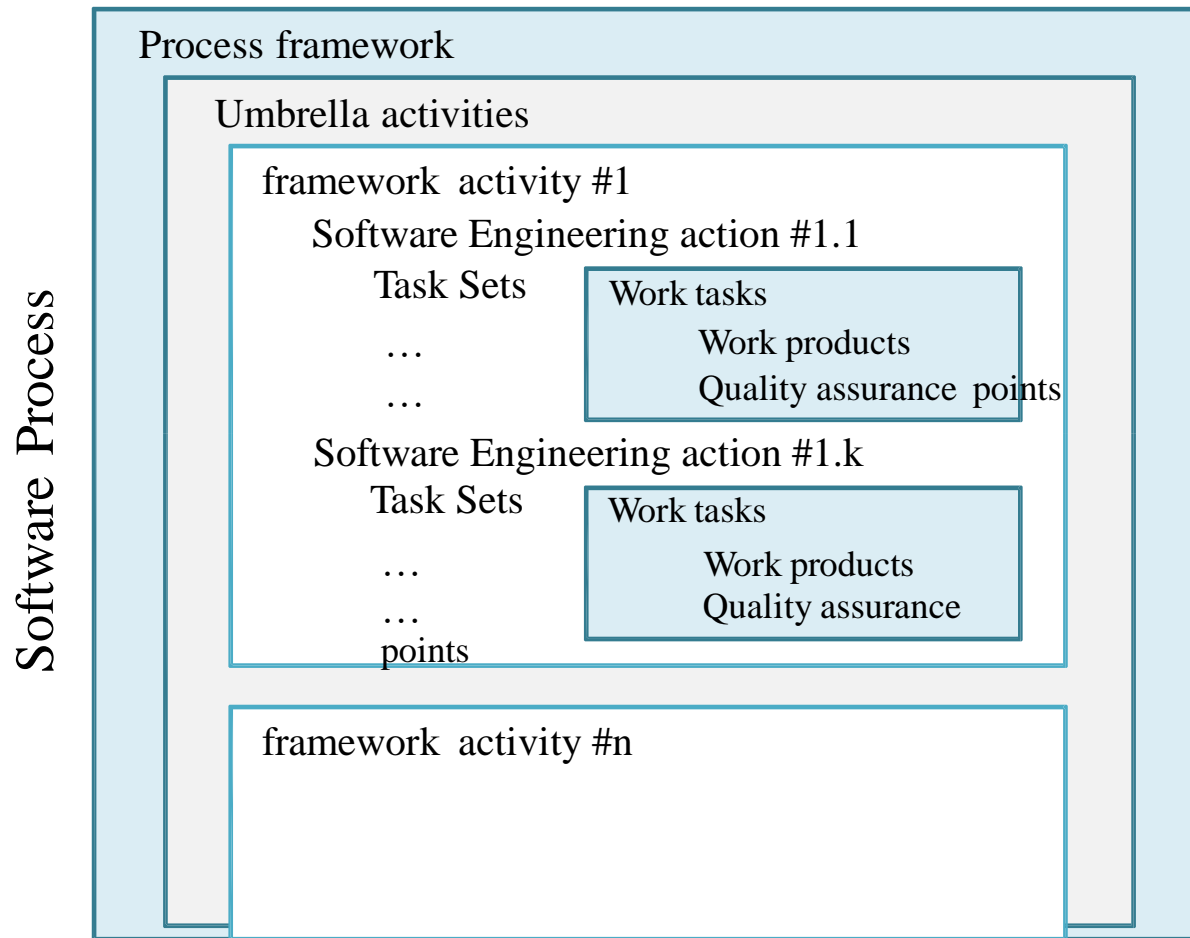
□ Reality:

- Software engineering is not about creating documents. It is about creating a quality product.
- Better quality leads to reduced rework. And reduced rework results in faster delivery times.

Software Process

- A process is a collection of **activities, actions and tasks** that are performed when some work product is to be created.
- **Each of these** activities, actions & tasks **reside within** a **framework** or model
- Each framework **activity is populated** by **set of** software engineering **actions**
- Each software engineering **action is defined by** a **task set** that identifies work to be completed, product to be produced, quality assurance points & milestones to indicate progress

Software Process Framework



Software Process Framework

- **Software Process Framework** is a foundation of complete software engineering process. Software process framework includes all set of umbrella activities. It also includes number of framework activities that are applicable to all software projects.

Software Process Models

- The process model is the abstract representation of process.
- Also known as **Software development life cycle (SDLC)** or Application development life cycle Models
- Process models **prescribe** a distinct set of **activities, actions, tasks and milestones (deliverables)** required to engineer high quality software.
- Process **models are not perfect**, but **provide roadmap** for software engineering work.
- Software models provide stability, control and organization to a process that if not managed can easily get out of control.
- Software process models are adapted (adjusted) to meet the needs of software engineers and managers for a specific project.

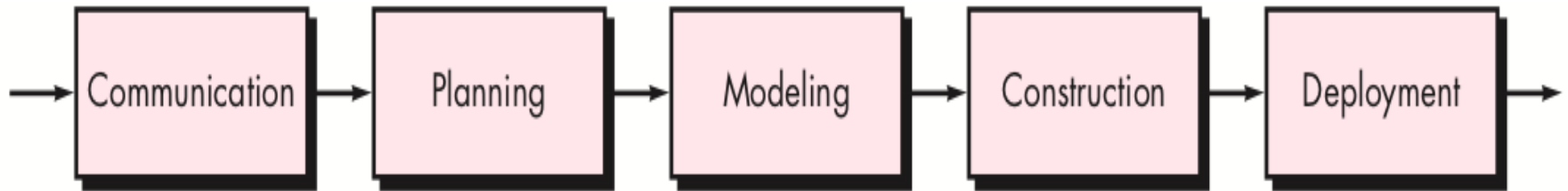
Software process model

- **Process models** prescribe a distinct set of activities, actions, tasks, milestones, and work products required to engineer high quality software.

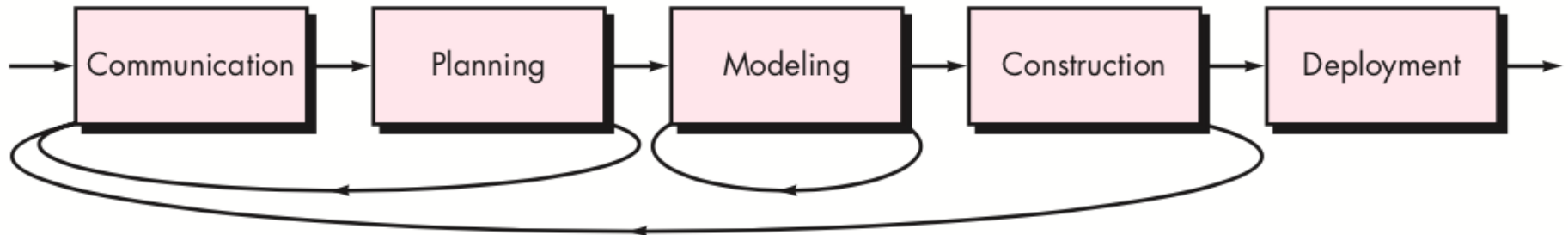
To solve actual problems in an industry setting, a software engineer or a team of engineers must incorporate a development strategy that encompasses the process, methods, and tools layers and the generic phases.

- This strategy is often referred to as a **process model** or a **software engineering paradigm**.
- *A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required.*

Process Flow

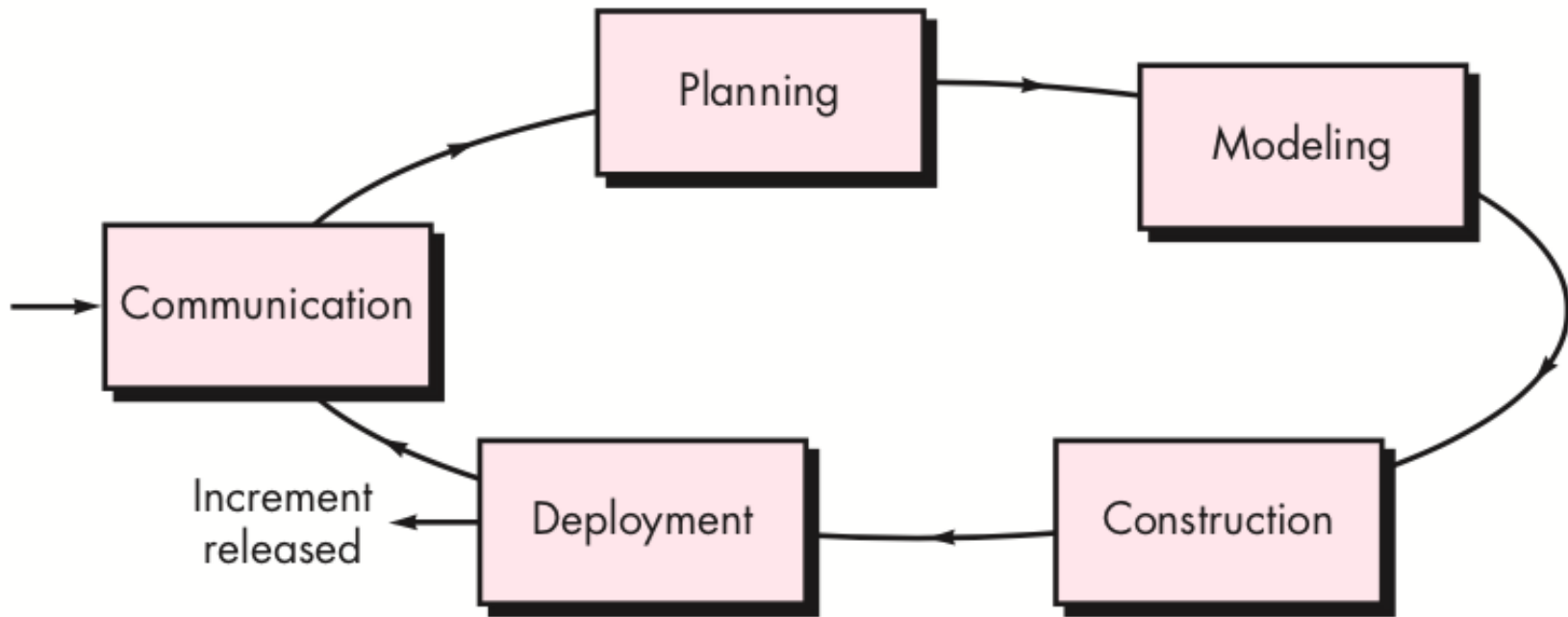


(a) Linear process flow



(b) Iterative process flow

Process Flow



(c) Evolutionary process flow

Types of Process Model

- Software development lifecycle model (Waterfall Model)
- Prototyping model
- V model
- incremental Model
- Iterative Model
- Spiral Model
- Agile Model