Chapter - 01

Overview of Software Engineering & The Software Development Process

Marks-20

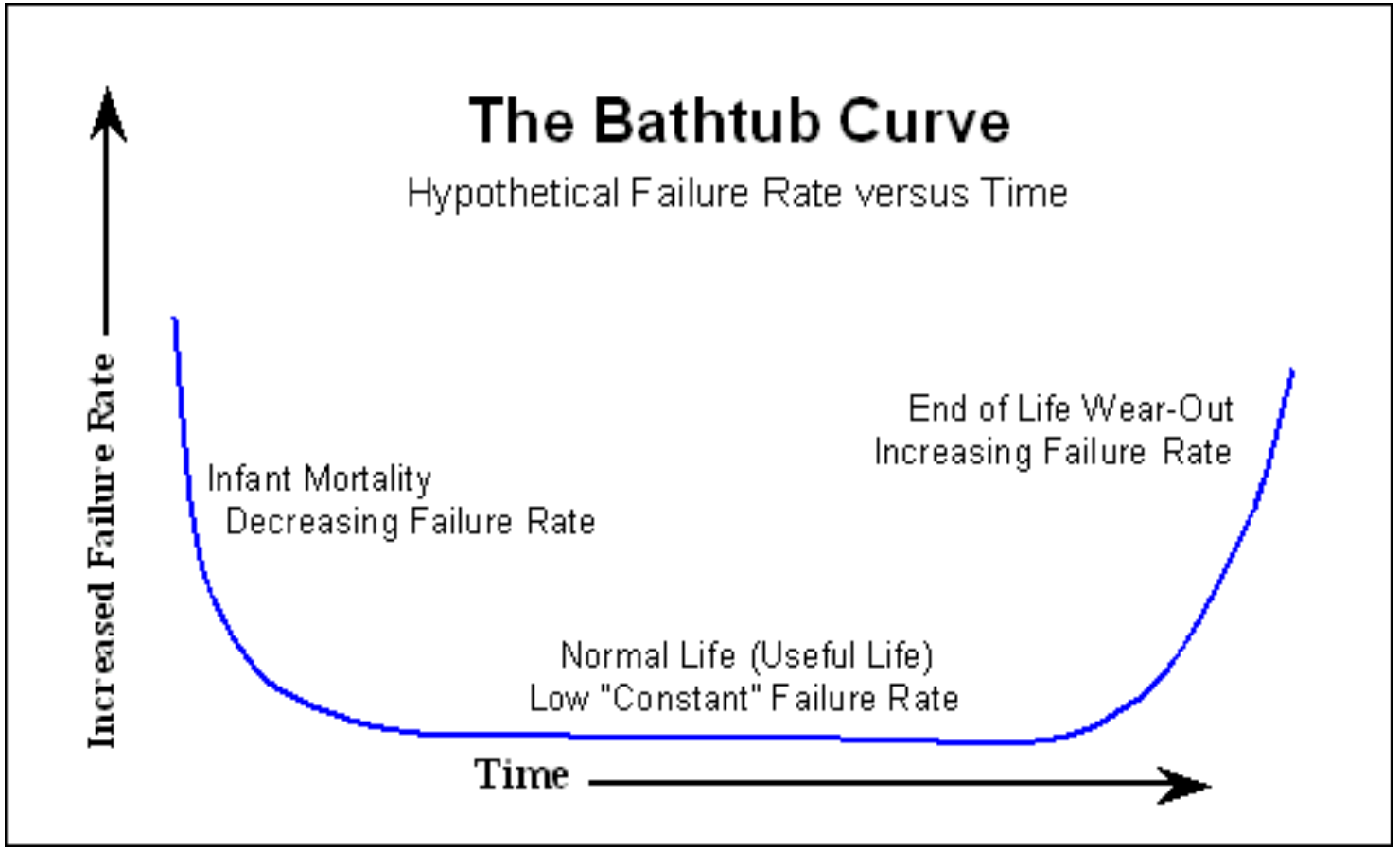
Definition of Software

- Software is a set of instructions that when executed, provide desired features, functions and performance.
- It is a datastructure that enables the programs to manipulate the information.
- Software is a document that describes the operation and use of programs.

Characteristics

1. Software is developed or engineered; it is not manufactured.
2. Software doesnt “wear out”.
3. Although the industry is moving towards component based constructions, most software continues to be custom built.

Bathtub Curve for Hardware failure



Bathtub Curve for Software failure

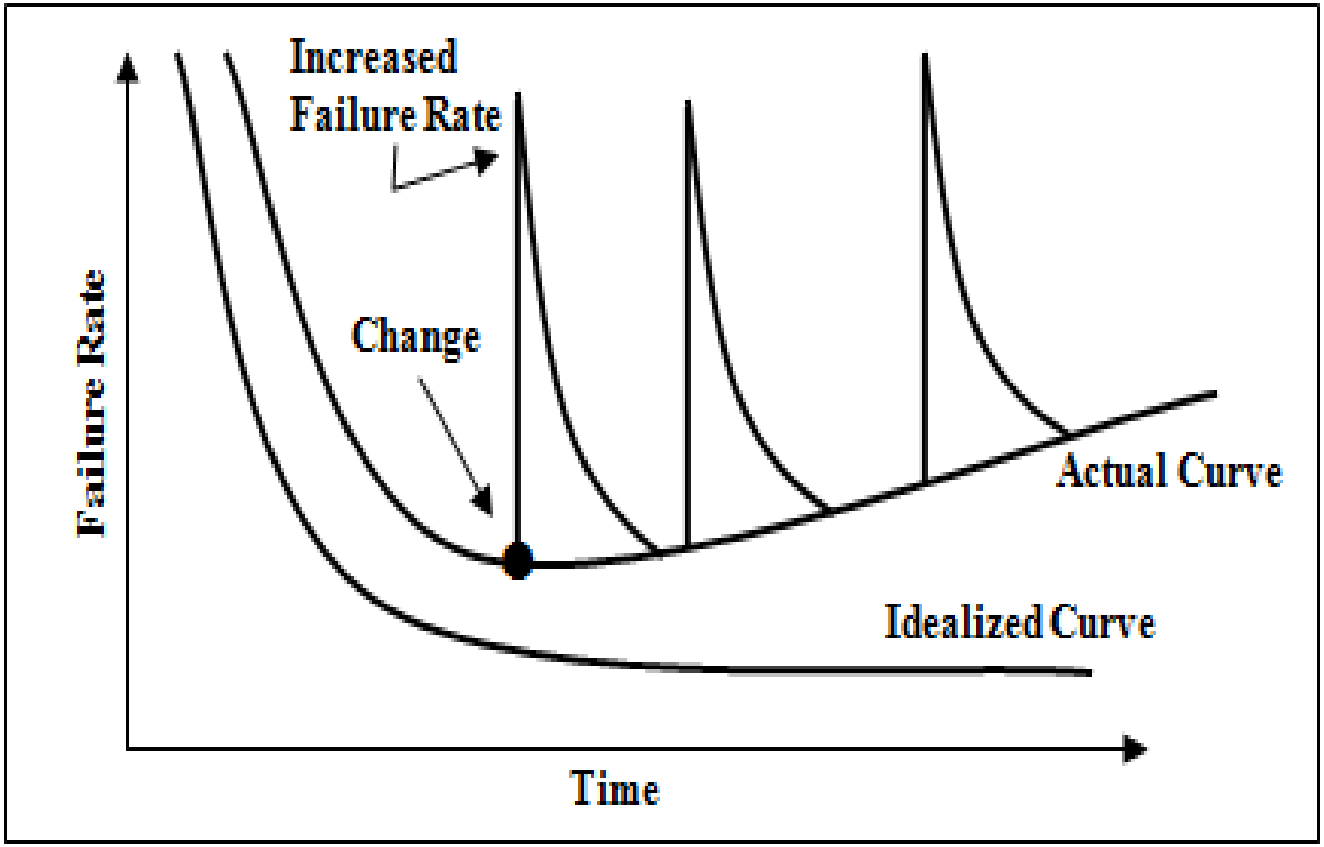


Figure 02: Failure Curves for software

Types/Categories of Software

1. System Software
2. Application Software
3. Engineering/Scientific Software
4. Embedded Software
5. Product-line Software
6. Web applications
7. Artificial Intelligence Software

System Software

- It is a collection of programs written to service other programs.
- System software area is characterized by heavy interaction with computer hardware.
- Ex: Operating system, Compilers, Editors, File management utilities, Drivers, Networking Software, Telecommunication Processors.

Application Software

- It consist of standalone programs that solve specific business needs.
- Application software is used to control business functions in real time.
- Ex: Microsoft office suite, Google docs, Browser

Engineering/Scientific Software

- This application range from astronomy to volconology, automotives stress analysis to space shuttle orbital dynamics, molecular biology to automated manufacturing etc.

Embedded Software

- It resides within a product or system and is used to implement and controll features and functions for the end users and for the system itself.
- Ex: Keypad control for microwave oven, Digital functions, Dashboard displays etc

Product-line Software

- Desigened to provide a specific capability for use by many different customes.
- It focus on limited marketplace or address mass consumer markets.
- Ex: Word processing, Spread sheets, Computer graphics, Entertainment, Multimedia, Database management, Business financial application.

Web applications

- Span a wide area of applications.
- In their simplest form, WebApps can be a little more than a set of linked hypertex files that present information using text and limited graphics.

Artificial Intelligence Software

- AI software makes use of nonnumerical algorithms to solve complex problems that are not amenable to computation or straight forward analysis.
- Ex: Robotics, Pattern recognition, Artificial nueral networks, etc

Definition of Software Testing

“The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works on real machines.”

Need of Software Engineering

1. Scientific and engineering approch to develop.
2. Project has to be divided into processes. Frame work activities, activities, task, etc.
3. Scheduling and controlling are the main activities guided by software project.
4. Different models are required for designing and analysis.
5. Huge management of resources.
6. Continuously deal with time and new technology challenges.

Relationship between System Engineering & Software Engineering

- System engineering takes place before Software engineering. It mainly focuses on system.
- System engineering understands role of people, procedures, database, hardware, software and other components.
- It analysis Modeling, validating and management etc of operational requirements.

- Software engineering is derived from System engineering.
- It mainly focuses on software product engineering and development process.
- It is a part of System engineering.
- System engineering is overall study of a system where software is going to be placed.

Software Engineering – A Layered Technology Approach

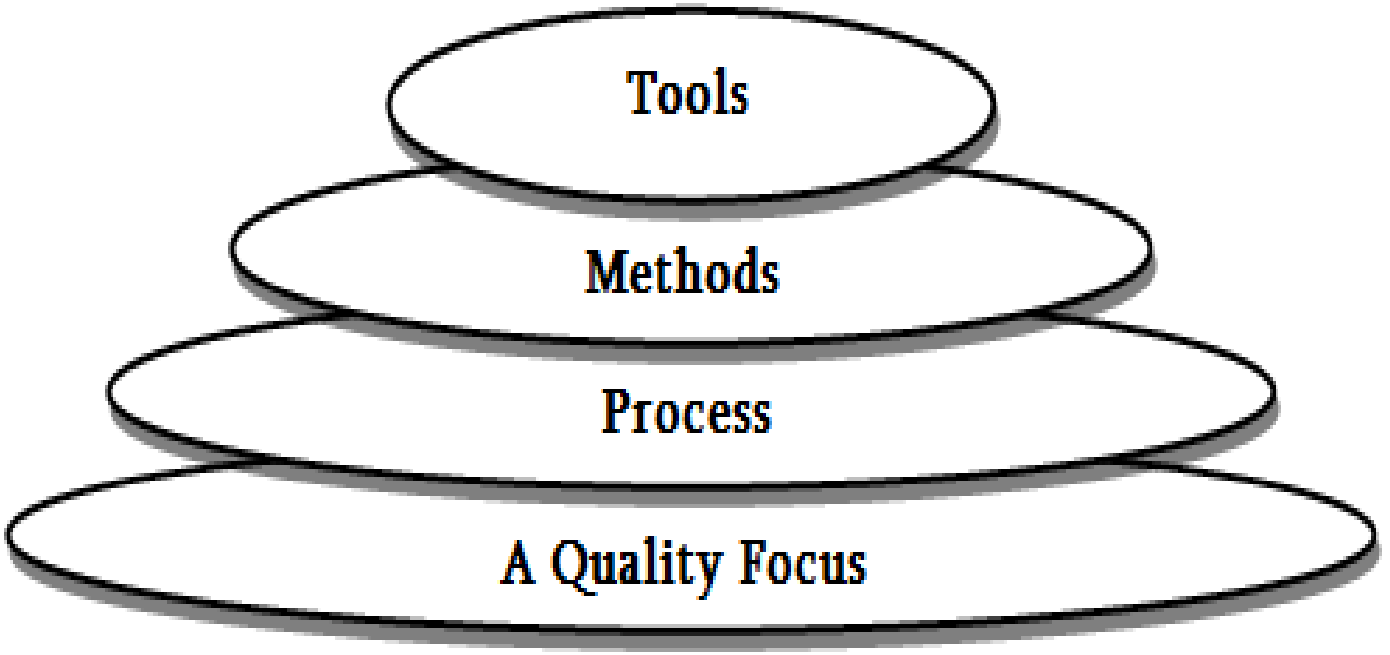


Figure: Flowchart of the Layers of Software Development

SOFTWARE ENGINEERING-A LAYERED TECHNOLOGY

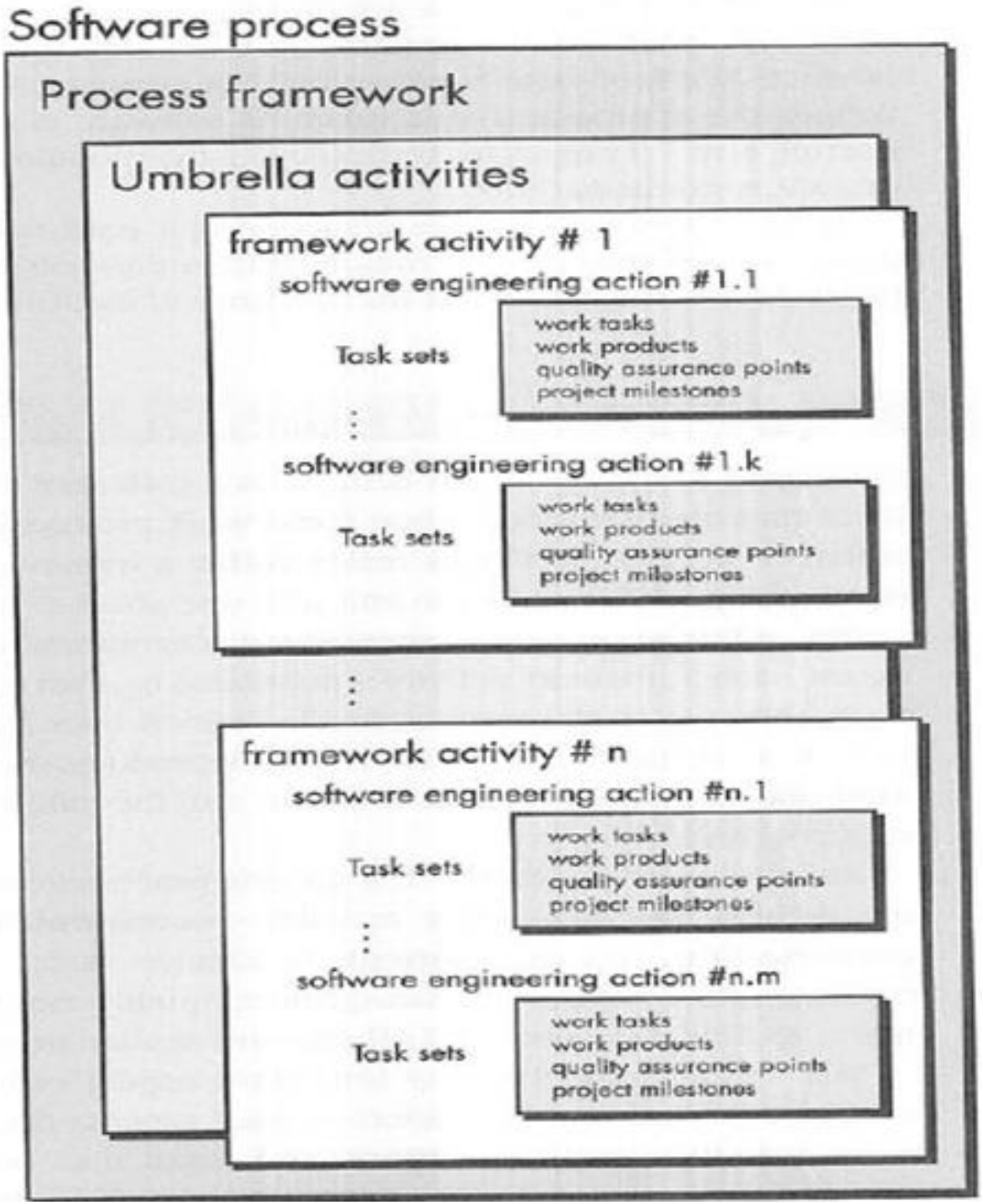
- Quality focus
  - Bedrock that supports software Engineering.
- Process
  - Foundation for software Engineering
- Methods
  - Provide technical How-to's for building software
- Tools
  - Provide semi-automatic and automatic support to methods

- Any engineering approach must rest on an organizational commitment to quality.
- Total Quality Management, Six Sigma and similar philosophies foster a continuous process improvement culture.
- This culture in turn develop increasingly more effective approaches to software engineering.
- The bedrock that supports SE is a quality focus.
- The foundation for SE is process layer.
- SE process is glue that holds technology layer together.

- Process defines a framework that must be established for effective delivery of SE technology.
- SE methods provide the technical “how to's” for building software.
- Methods include communication, requirements analysis, design modeling, program construction, testing and support.
- SE tools provide automated or semiautomated support for the process and methods.
- When tools are integrated so that information created by one tool is used by other tool.
- It leads to computer aided software engineering.

Software Process

- A software process as a framework for the tasks that are required to build high quality software.
- A software process defines the approach that is taken as software is engineered. But SE also encompasses technologies.


Software process
Process framework
Umbrella activities
framework activity # 1
software engineering action #1.1
Task sets
work tasks
work products
quality assurance points
project milestones
software engineering action #1.k
Task sets
work tasks
work products
quality assurance points
project milestones
framework activity # n
software engineering action #n.1
Task sets
work tasks
work products
quality assurance points
project milestones
software engineering action #n.m
Task sets
work tasks
work products
quality assurance points
project milestones

Process framework

- A process framework establishes the foundation for a complete software process by identifying a small number of framework activities that are applicable to all software projects.
- SE actions: a collection of related tasks that produces a major SE work product.
- Each action is populated with individual work tasks that accomplish some part of work implied by action.

Generic process framework activities

1. Communication
2. Planning
3. Modeling
4. Construction
5. Deployment

The framework described in the generic view of SE is complimented by number of Umbrella activities.

Umbrella Activities

1. Software project tracking and control.
2. Risk management.
3. Software quality assurance .
4. Formal Technical reviews.
5. Measurement.
6. Software configuration management.
7. Re usability management.
8. Work product preparation and production.

1. Software project tracking and control: assess progress against the plan and take actions to maintain the schedule.
2. Risk management: assesses risks that may affect the outcome of project or quality of product.
3. Software quality assurance: defines and conduct activities required to ensure s/w quality.
4. Formal Technical reviews: assesses SE work products to uncover and remove errors before going to the next activity.

5. Measurement: define and collects process, project, and product measures in delivering s/w that meets customers need.
6. Software configuration management: manage the effects of change throughout the software process.
7. Re usability management: defines criteria for work product reuse and establishes mechanism to achieve reusable components.
8. Work product preparation and production: create work products such as models, documents, logs, forms and lists.

Personal and Team process Models

Personal Software process:-

The PSP model defines five framework activities :-

1. Planning
2. High level Design
3. High level Design Review
4. Development
5. Postmortem

Team Software Process

- Build self directed teams that plan and track their work,establish goals,own their processes and plans.
- Show managers how to coach and motivate their teams and how to help them.
- Accelerate software process improvement.
- Provide improvement guidance to high maturity organisations.
- Facilitate university teaching of industrial grade team skills.

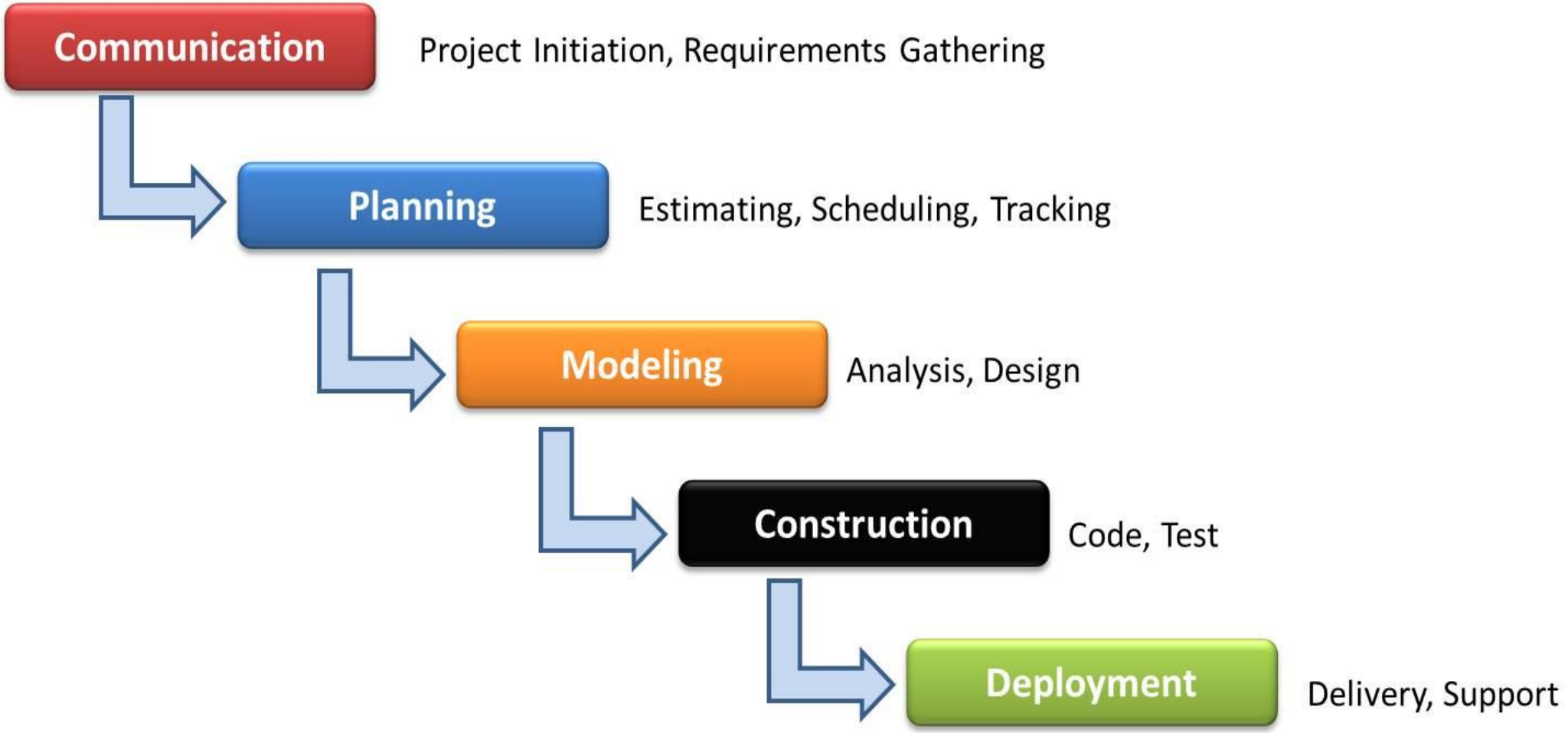
- A self directed team has a consistent understanding of its overall goals and objectives.
- It defines role and responsibilities for each member, track project data.
- Continually assesses risk and reacts to it and manage project status.

Prescriptive process model

There are called “prescriptive” because they prescribe

- a set of process framework **activities**
- software engineering **actions**,
- **tasks,**
- **work products,**
- **quality assurance** and
- **change control** mechanism for each project.

The Waterfall model



The Waterfall Model: A Traditional Approach of SDLC

- The Waterfall Model was first Process Model to be introduced.
- It is also referred to as a **linear-sequential life cycle model**.
- It is **very simple to understand and use**.
- In a waterfall model, each phase must be completed fully before the next phase can begin.
- This type of model is basically **used for the for the project which is small and there are no uncertain requirements**.
- At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project.
- In this model the testing starts only after the development is complete. **In waterfall model phases do not overlap**.

Communication

In communication the major task is **requirements gathering** which helps to find out the exact need of customer.

Planning

It includes some major activities such as planning for schedule, tasks, tracks on the process and the estimation related to the project

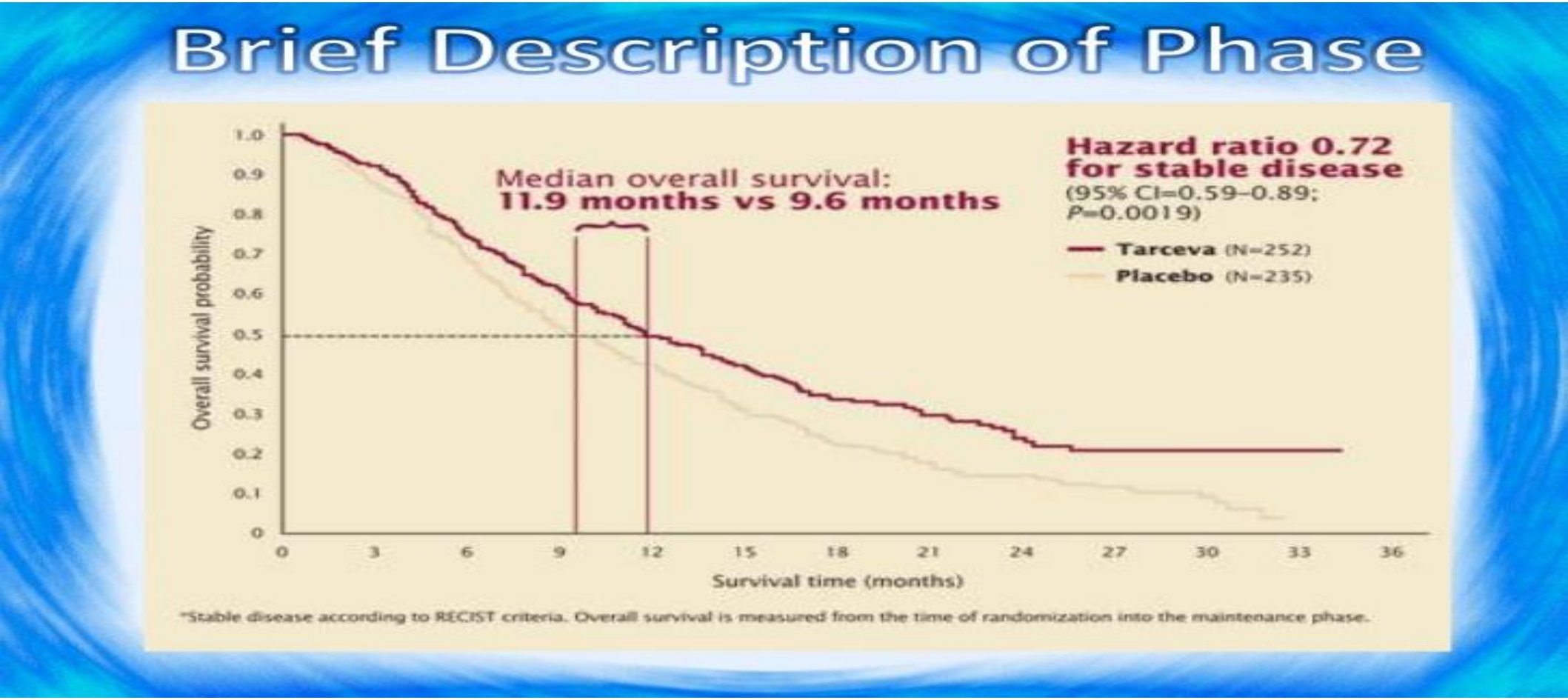
Modelling

Modelling is used to analyse the data and as per the analysis the data and process will be designed.

Construction

Construction is based on the design of the project. According the design of the project coding and testing is done.

Deployment



The product is actually delivered that is installed at customer's site. As well as feedback is taken from the customer to ensure the quality of product.

When to Use

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined.
- There is a need to get a product to the market early.
- A new technology is being used.
- Resources with needed skill set are not available
- There are some high risk features and goals.

Advantages of waterfall model:

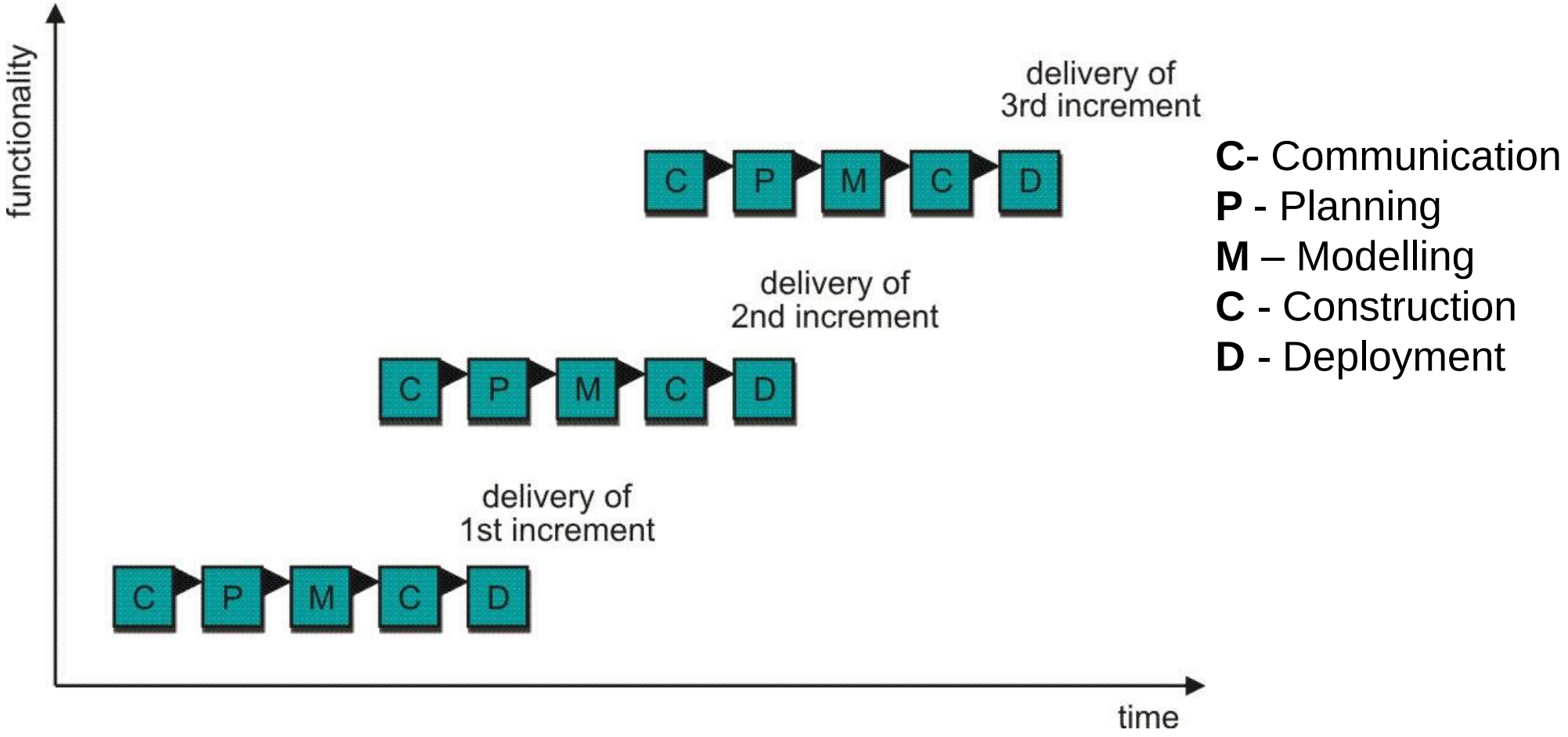
1. This model is simple and easy to understand and use.
2. It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
3. In this model phases are processed and completed one at a time. Phases do not overlap.
4. Waterfall model works well for smaller projects where requirements are very well understood.

Disadvantages of waterfall model:

1. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
2. No working software is produced until late during the life cycle.
3. High amounts of risk and uncertainty.
4. Major design problems may not be detected early.
5. The model implies that once the product is finished, everything else is maintenance

Incremental process Model

Incremental Process Model



Delivers software in small but usable pieces, each piece builds on pieces already delivered

Incremental process Model

- This combines elements of waterfall model applied in parallel process flows.
- Each linear sequence produce **deliverable "increments"** of the s/w product.
- It produce a s/w product as a **series of increment release**.
- When an incremental model is used the **first increment is often a core product** i.e. basic requirement
- The core product is used by customer. As a result of use ,a plan is developed for the next increment.
- This process is repeated following the delivery of each increment, until the complete product is produced.

For **example** : **Word processing s/w** is developed using incremental paradigm then,

1) In 1st:**Basic file management editing & document production** functions are delivered

2) In 2nd : More **sophisticated editing & document** production capabilities are delivered

3) In 3rd : **Spelling & grammar checking** functions are delivered

4) In last : **Advanced web page layout** capabilities functions are delivered.

Advantages

- Customers **get usable functionality earlier** than with waterfall.
- **Early feedback** improves likelihood of producing a product that satisfies customers.
- The **quality** of the final product **is better**
  - The core functionality is **developed early and tested multiple times** (during each release)
  - Only a relatively small amount of functionality added in each release: easier to get it right and test it thoroughly
  - **Detect design problems early** and get a chance to redesign

Disadvantages

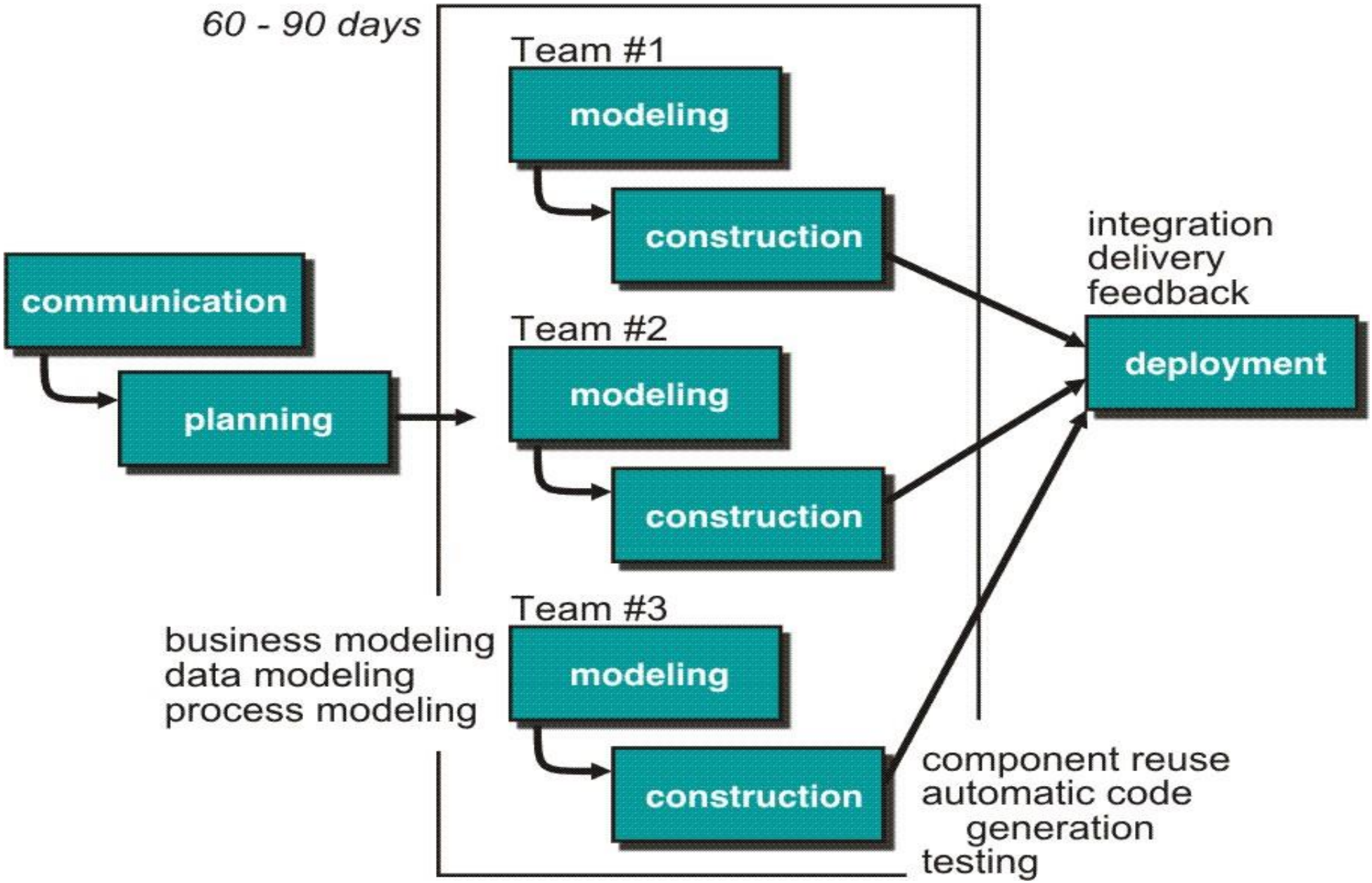
- Needs **good planning and design**.
- Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.
- Total **cost is higher** than waterfall.

Sr. No	Waterfall Model	Incremental Model
1	When there is need to make well - defined adaptations or enhancements to an existing system and product definition is stable, waterfall model is used.	When there is need to provide limited set of functionality to users quickly and then refine and expand on that functionality in later Software releases, incremental approach is used.
2	It requires well understanding of requirement and familiar technology	Requirements of the complete system are clearly defined and understood
3	Sequential in nature	Incremental in nature
4	Difficult to accommodate changes after the process has started	Changes can be accommodated when planning for next increment
5	Risk can be identified at the end which may cause failure to the product	Risk can be identified in each Increment plan
6	The customer can see the working model of the project at the end. After review of the working model, if the customer get dissatisfied then it cause serious problems	The core product is used by the customer (or undergoes detailed review). As a result of use and/or evaluation, a plan is developed for next increment

Rapid Application Development Model(RAD)

- RAD is an incremental software process model that emphasizes a **short development cycle**.
- Using **Component based construction** approach.

Rapid Application Development Model(RAD)



The RAD approach activities :

1. **Communication:** Works to understand the business problems
2. **Planning** : Is essential because multiple s/w teams work in parallel on different system function.
3. **Modelling** : **3 Major phases -Business** modelling, **Data** modelling & **process** modelling.
4. **Construction** :Emphasizes on the use of **pre-existing s/w components & application**.
5. **Deployment** : **Changes & innovations are done** if required for customer satisfaction.

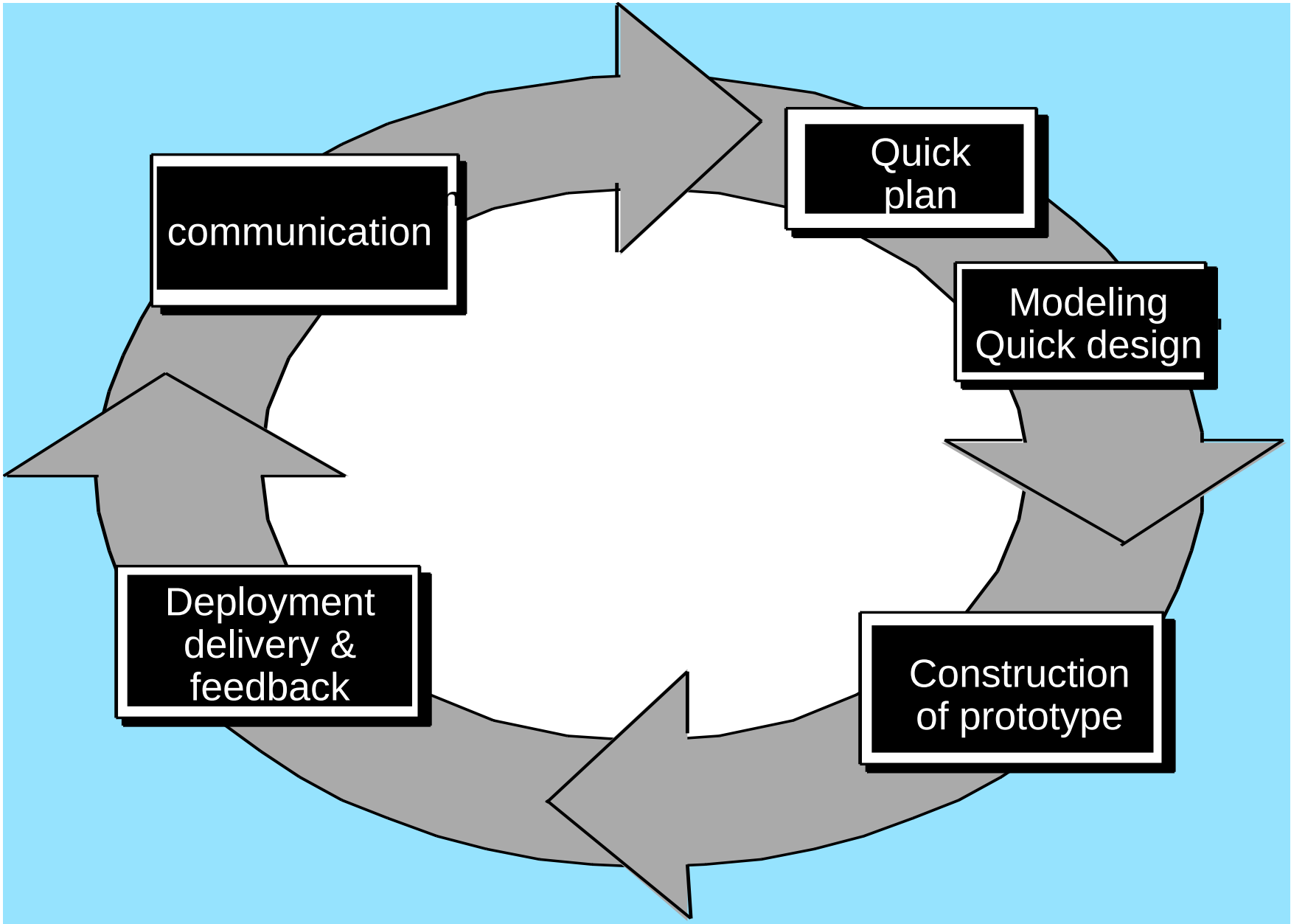
Advantages

1. Useful when the time limit for development is **too short**.
2. Since **reusability** is used ,many of the program components are already tested. This reduce overall testing time.
3. All functions are modularized so it is easy to work with

Disadvantages

- For large projects RAD **require highly skilled engineers** in the team.
- Both end customer and developer should be committed to complete the system in time, if **commitment** is lacking RAD will fail.
- RAD is based on **Object Oriented approach** and if it is difficult to modularize the project the RAD may not work well.

Evolutionary Models: Prototyping



Evolutionary Models: Prototyping

- Business and product requirement often change as development proceed.
- Software engineer need a process model that has been explicitly designed to **accommodate a product that evolves over time**.
- Evolutionary models are **iterative**.
- Enables software engineers to **develop increasingly more complete version of the software**.

There are two types of evolutionary development:

– **Exploratory** development

• Start with requirements that are well defined

• Add new features when customers propose new requirements

– **Throw-away** prototyping

• Objective is to understand customer's requirements (i.e. they often don't know what they want), hence poor requirements to start

• Use means such as prototyping to focus on poorly understood requirements, redefine requirements as you progress

Steps in Prototyping

- Begins with requirement Gathering
- Identify whatever requirements are known.
- Outline areas where further definition is mandatory.
- A quick design occurs.
- Quick design leads to the construction of prototype.
- Prototype is evaluated by the customer.
- Requirements are refined
- Prototype is turned to satisfy the needs of customer

Advantages

1. The risk factor is very low
2. With less investment of finance & time, the requirements are confirmed

Disadvantages

1. Leads to implementing and then repairing way of building systems.
2. Practically, this methodology may increase the complexity of the system.
3. Incomplete application may cause application not to be used as the full system.

When to use Prototype model:

- This model should be used when the desired system needs to have a lot of interaction with the end users.
- Typically, **online systems, web interfaces** have a very high amount of interaction with end users, are best suited for Prototype model.
- Prototyping ensures that the end users constantly work with the system and provide a feedback which is incorporated in the prototype to result in a usable system.
- They are excellent for designing good human computer interface systems.

Spiral Model

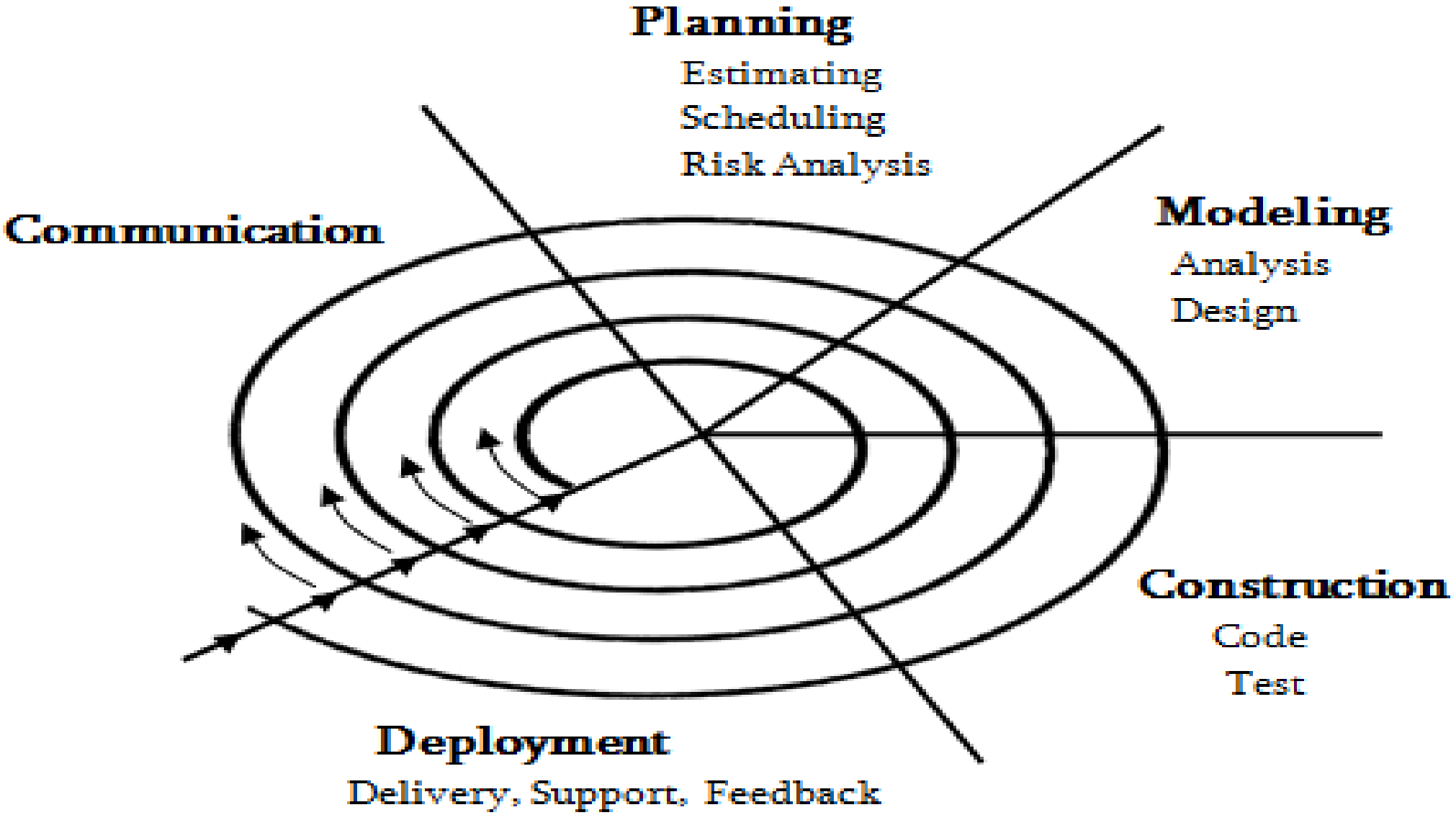


Figure : Spiral Model

- Spiral model is a **combination of iterative development process model** and **sequential** linear development model i.e. waterfall model
- It allows for incremental releases of the product, or incremental refinement through each iteration around the spiral.

Advantages of Spiral model:

High amount of risk analysis hence, avoidance of Risk is enhanced.

1) Good for large and mission-critical projects.
2) Strong approval and documentation control.
3) Additional Functionality can be added at a later date.
4) Software is produced early in the software life cycle.

Disadvantages of Spiral model:

1. Can be a costly model to use.
2. Project's success is highly dependent on the risk analysis phase.
3. Doesn't work well for smaller projects.

When to use Spiral model:

- When costs and risk evaluation is important
- For medium to high-risk projects
- Users are unsure of their needs
- Requirements are complex
- Significant changes are expected (research and exploration)

Agile Software Development

- It focuses on the **rapid development of the s/w product** by considering the current market requirements and time limits.
- Todays **market is rapidly changing and unpredictable**.
- Agile solves the problem of long time and heavy documentation s/w development process.
- Agile focuses on **face to face or interactive processes** than documentation.

- It doesn't believe in more and more documentation because it makes difficult to find the required information.
- It **supports team to work together with management** for supporting technical decision making.
- This method **focuses mainly on coding** because it is directly deliverable to the users.
- Agile **saves man power, cost, documentation** and **time**.

Features of the Agile Software Development Approach

Modularity: Modularity allows a process to be broken into components called activities. A set of activities capable of transforming the vision of the software system into reality.

Iterative: Agile software processes acknowledge that we get things wrong before we get them right. Therefore, they focus on short cycles. Within each cycle, a certain set of activities is completed.

Time-Bound: Iterations become the perfect unit for planning the software development project. We can set time limits on each iteration and schedule them accordingly.

Parsimony: Agile software processes focus on parsimony. That is, they require a minimal number of activities necessary to mitigate risks and achieve their goals.

Adaptive: During an iteration, new risks may be exposed which require some activities that were not planned. The agile process adapts the process to attack these new found risks.

Incremental: An agile process does not try to build the entire system at once. Instead, it partitions the nontrivial system into increments which may be developed in parallel, at different times, and at different rates.

Convergent: Convergence states that we are actively attacking all of the risks worth attacking. As a result, the system becomes closer to the reality that we seek with each iteration.

People-Oriented: Agile processes favor people over process and technology. Developers that are empowered raise their productivity, quality, and performance.

Collaborative: Communication is a vital part of any software development project. When a project is developed in pieces, understanding how the pieces fit together is vital to creating the finished product.

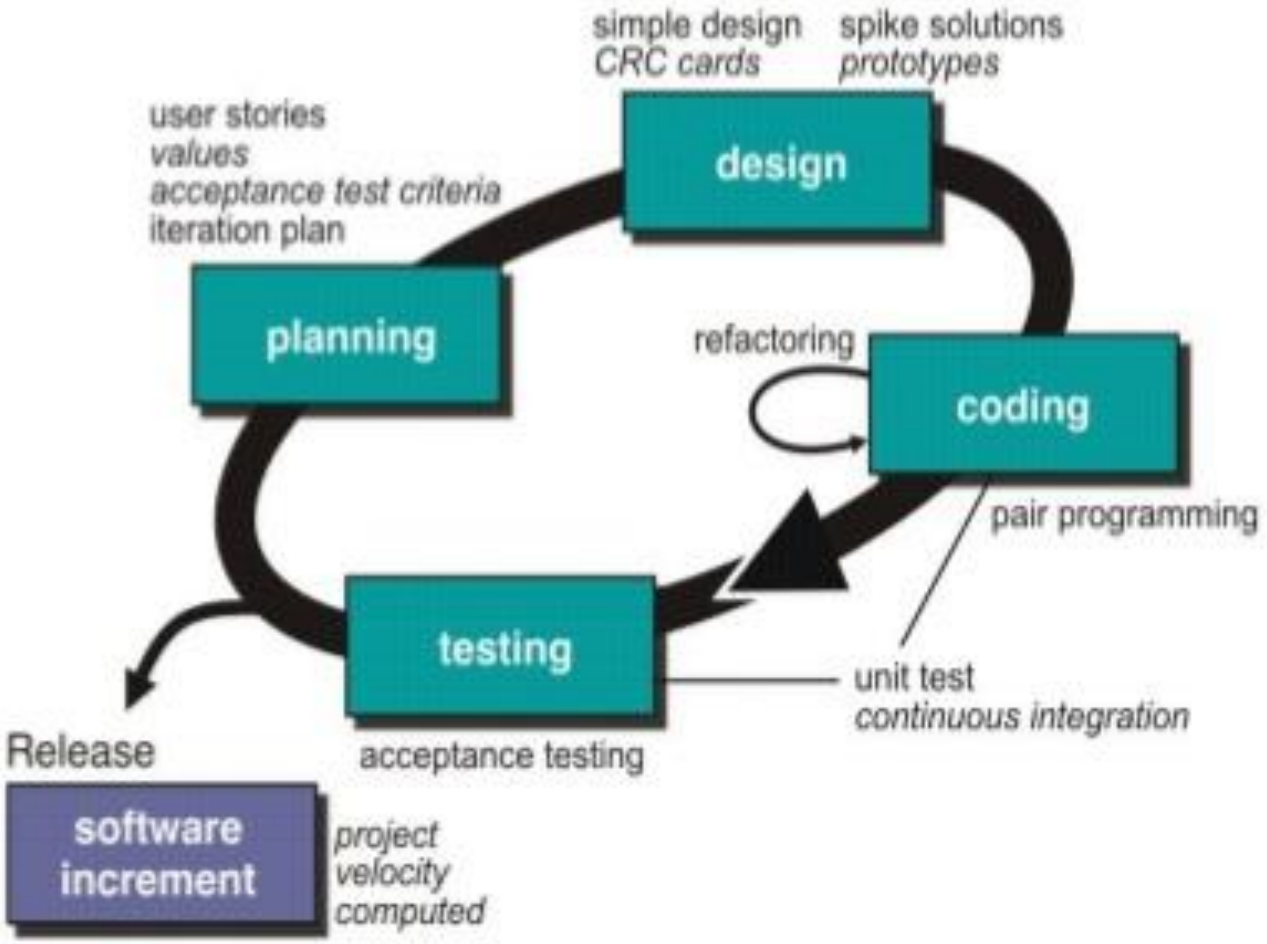
Difference between Prescriptive Process Model and

Sr. No	Prescriptive Process Model	Agile Process Model
1.	Product Oriented process. Process and technology are crucial	People oriented process. Favors people over technology
2.	A traditional approach for software product development	It is an recent approach for Project Management
3.	Traditional and modern approaches using generic process framework activities with medium to large cycle-time	Cycle-time reduction is most important
4.	Focus is on tasks, tools such as estimating, scheduling, tracking and control	Model focuses on modularity, iterative, time bound, parsimony, adaptive, incremental convergent, collaborative approach
5.	Models include Waterfall, Incremental, Prototype, RAD and spiral	Agile process model uses the concept of Extreme Programming

Extreme programming

- The best-known and most widely used agile method.
- Extreme Programming (XP) takes an ‘extreme’ approach to iterative development.
- New versions may be built **several times per day**;
- Increments are delivered to customers **every 2 weeks**;
- All tests must be run for every build and the build is only accepted if tests run successfully.

Extreme Programming (XP)



- XP is a disciplined approach to software development based on value of **simplicity, communication and feedback**.
- It empowers developers to confidently response to the changing needs of customers even late in life cycle.