Software Engineering
III B.sc computer science A section

Lecture Notes Compiled by

DR.M.D.ANANDA RAJ M.C.A, M.Phil, M.B.A,Ph.D,NET

ASST.PROFESSOR,

DEPT OF COMPUTER SCIENCE,LOYOLA COLLEGE

mdanandaraj@loyolacollege.edu

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
- (3) documentation that describes the operation and use of the programs.

definition

- SE:the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines.
- IEEE Definition:

The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software

Characteristics of software

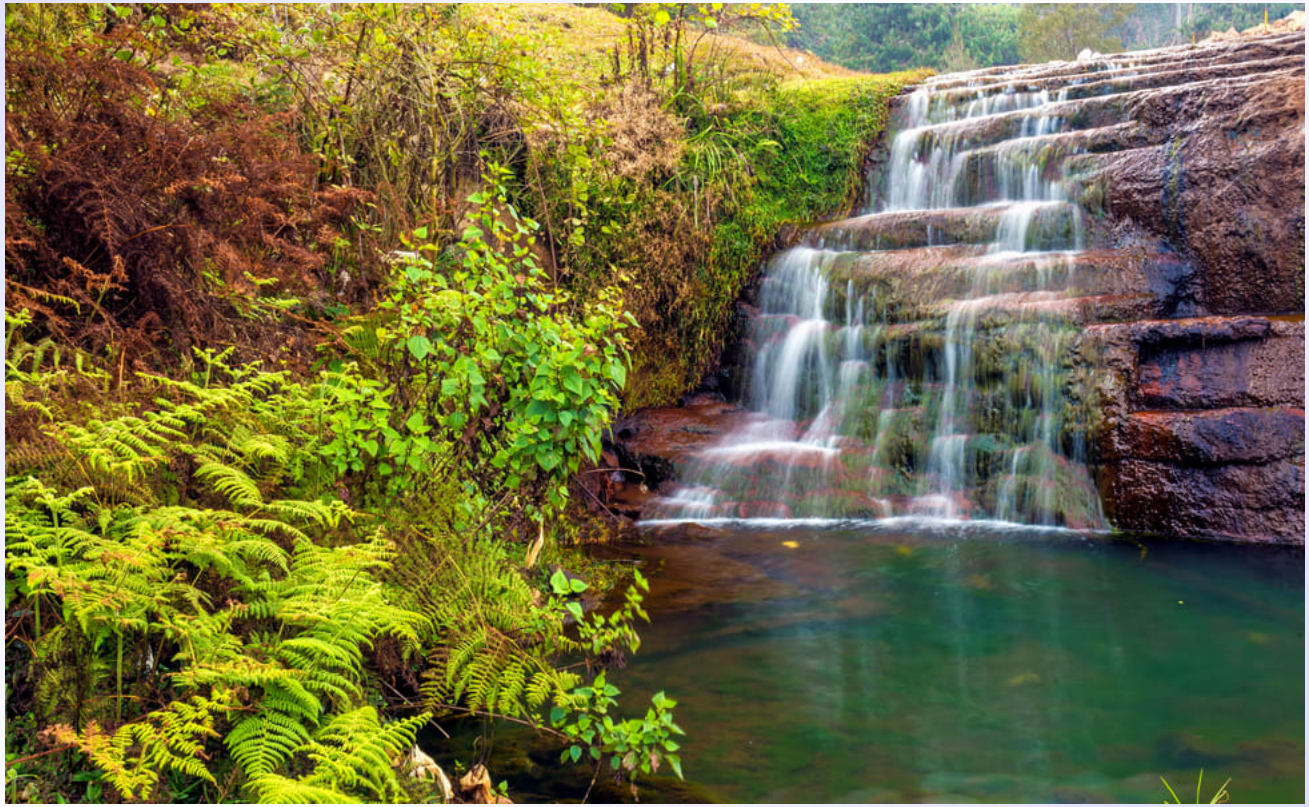
- Software is developed or engineered but not manufactured
- Software doesnot wear out
- Sotware can be customised based on the requirements of the customer.

Different types of software

7 software
WAPE SEA

- system software
- application software
- engineering/scientific software
- embedded software
- product-line software
- WebApps (Web applications)
- AI software

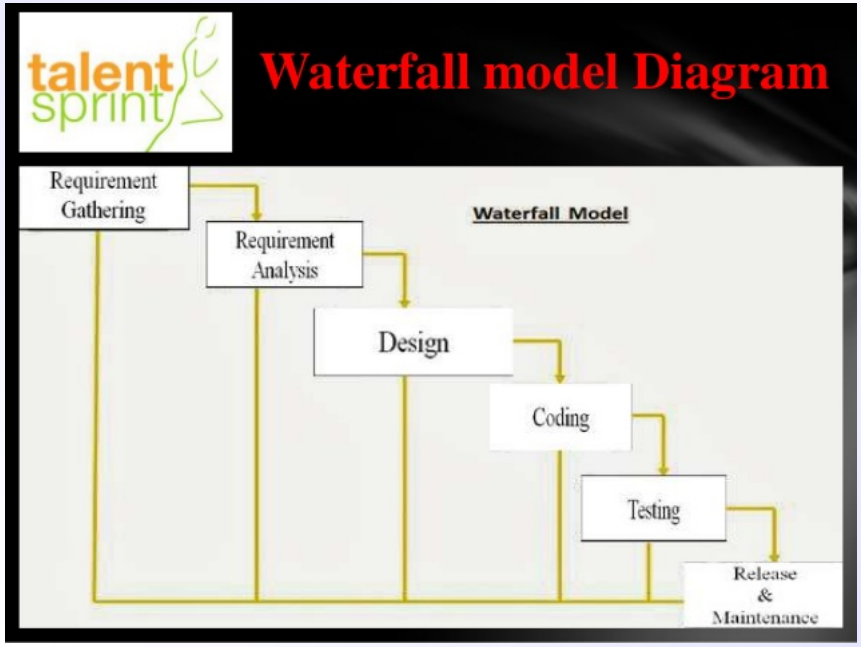
Water falls?



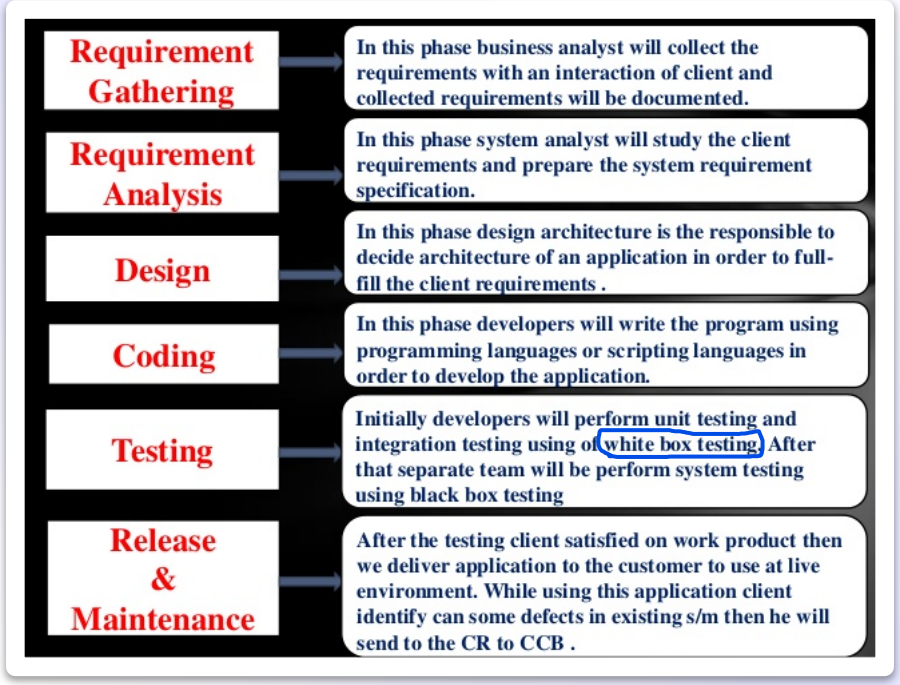
New categories software

- Open world computing—pervasive, distributed computing Ubiquitous computing—wireless networks
- Netsourcing—the Web as a computing engine
- Open source—”free” source code open to the computing community

Water fall model (or) Linear sequential model



Phases..



white box testing is a integration testing

black box testing is a seperate team will perform a system testing

Advantages

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

Disadvantages

- 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.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing.

When to use this model

- This model is used only when the requirements are very well known, clear and fixed.
- Product definition is stable.
- Technology is understood.
- There are no ambiguous requirements
- Ample resources with required expertise are available freely
- The project is short.

SOFTWARE MYTHS

- Myth means wrong belief or mis information
- Software myth: beliefs about software and the process used to build it.
- ***misleading beliefs that have caused serious problems for managers and programmers***

Different types of myths

- Management myths

  one or group of people who manages the software company
- Customer myths

  - user or the end user who is going to use the software
- Developer or practitioner myths

  -one who is going to develop the software

Management myths

- Development problems can be solved by developing and documenting standards.
- Development problems can be solved by using state-of-the art tools.
- When schedules slip, just add more people to finish the job

Mangement myths

1. Books that indicate standards and procedures for building software are enough to produce good programs.

- **REALITY:** Software people **do not look at the standards and procedures**, and sometimes standards and procedures are **out of date** or **incomplete**.

2. The software group has got the **newest and fastest computers** so they should be able to produce good programs.

- **REALITY:** Computer Aided Software Eng.(CASE) **Tools** are more **important than Hardware** for achieving good quality and productivity. Therefore, latest model of PC is not enough.

3. If we get **behind the schedule** in a software project, we **can add more programmers** and finish on time.

- **REALITY: Adding people to a late software project makes it later!** (As new people are added, they **have to be trained** by the others, and this **reduces** their time.)

4. A general statement of objectives is enough to begin writing programs. We can fill in the details later.

- **REALITY:** This is **major cause of failure** in software projects. We **need a formal** and **detailed description** of the system *modules* , *interfaces*, and *design criteria first*.

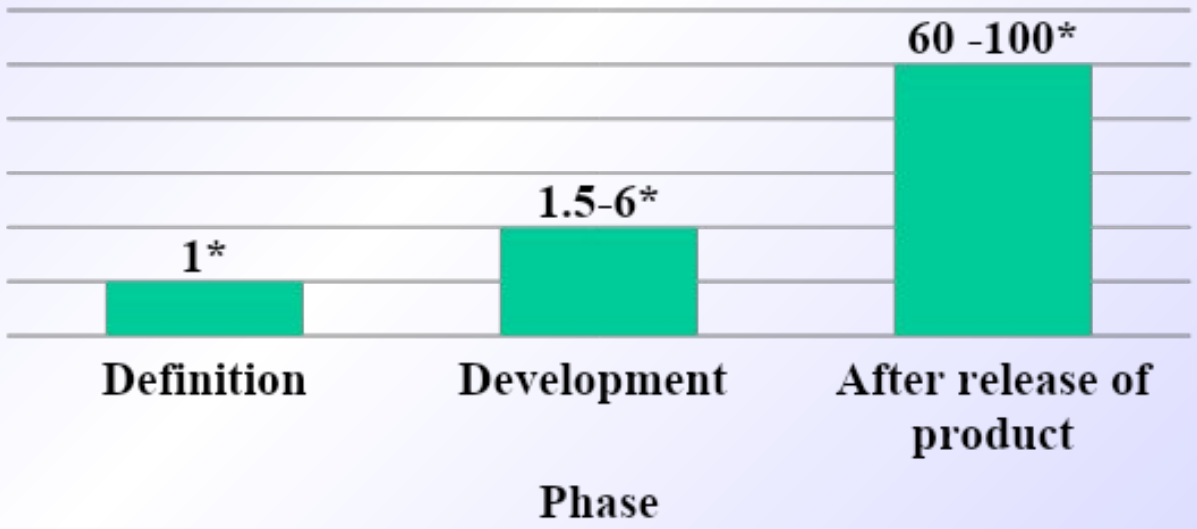
Customer myths

4. A general statement of objectives is enough to begin writing programs. We can fill in the details later.

- **REALITY:** This is **major cause of failure** in software projects. We **need a formal** and **detailed description** of the system *modules* , *interfaces*, and *design criteria first*.

5. Project requirements are changing continuously. However, we can easily handle the changes since software is flexible.

- **REALITY:**



Developer myths

Once we write the program and it works, our job is done.

- **REALITY:**
- «The sooner you begin writing code, the longer it will take you to complete»
- **50-70 % of the effort** is spent **after delivery** to customer.

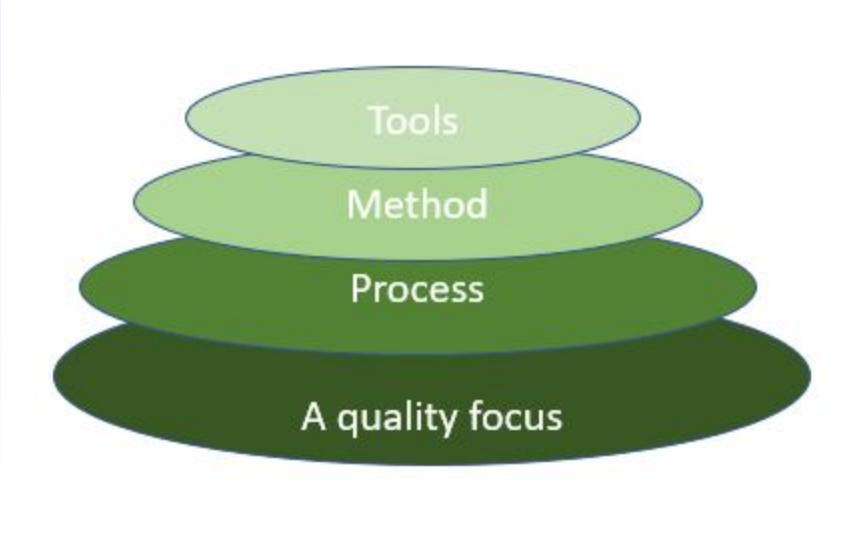
- Also,the **final product** should include **documents** and **data**.
- **Because, Proper documentation** is very important for success, and it guides the **maintenance**.

*Project success depends solely on the quality of the delivered **program***. we *can't assess software quality until the program is running.*

Software engineering –Layered technology

- Software engineering: it is a fully layered technology, to develop software we need to go from one layer to another. All the layers are connected and each layer demands the fulfillment of the previous layer.

Layer- diagram



Layered technology

1.A Quality Focus: It defines the continuous process improvement principles of software. It provides integrity that means providing security to the software so that data can be accessed by only an authorized person, no outsider can access the data. It also focuses on maintainability and usability.

2.Process: It is the foundation or base layer of software engineering. It is key that binds all the layers together which enables the development of software before the deadline or on time. Process defines a framework that must be established for the effective delivery of software engineering technology. The software process covers all the activities, actions, and tasks required to be carried out for software development.

Process Activities..

- **Communication:** It is the first and foremost thing for the development of software. Communication is necessary to know the actual demand of the client.
- **Planning:** It basically means drawing a map for reduced the complication of development.
- **Modeling:** In this process, a model is created according to the client for better understanding.
- **Construction:** It includes the coding and testing of the problem.
- **Deployment:-** It includes the delivery of software to the client for evaluation and feedback

Layered technology

3. Method: During the process of software development the answers to all "how-to-do" questions are given by method. It has the information of all the tasks which includes communication, requirement analysis, design modeling, program construction, testing, and support.

4. Tools: Software engineering tools provide a self-operating system for processes and methods. Tools are integrated which means information created by one tool can be used by another

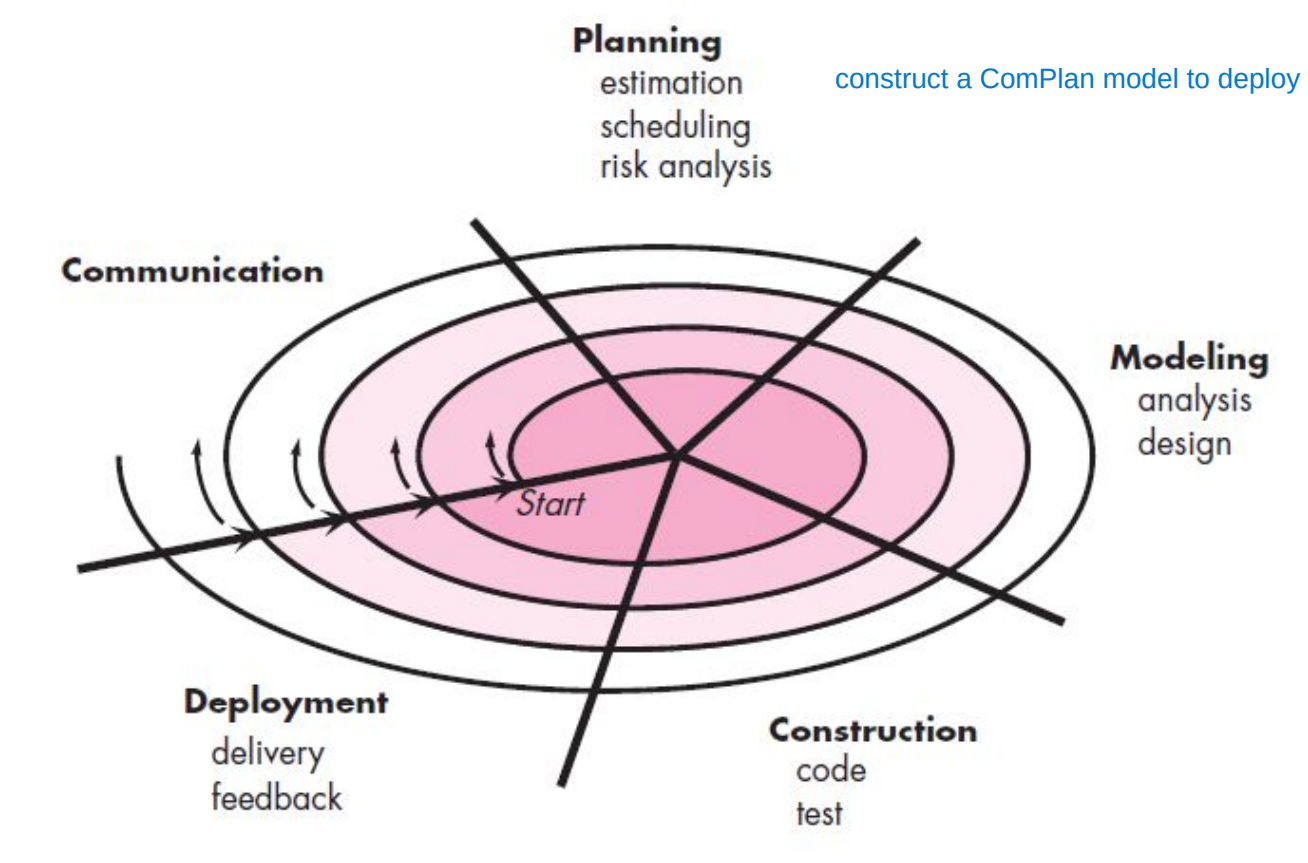
Spiral Model

- Spiral Model is a risk-driven software development process model. It is a combination of waterfall model and iterative model. Spiral Model helps to adopt software development elements of multiple process models for the software project based on unique risk patterns ensuring efficient development process.

- Each phase of spiral model in software engineering begins with a design goal and ends with the client reviewing the progress.

- Each phase of spiral model in software engineering begins with a design goal and ends with the client reviewing the progress.

Spiral model -diagram



Spiral model

- Spiral Model has 4 important phases :
- **1.planning**
- **2.Risk analysis**
- **3.Engineering**
- **4.Evaluation**
- ·**Planning** : It includes estimating the cost, schedule and resources for the iteration. It also involves understanding the system requirements for continuous communication between the system analyst and the customer
- ·**Risk Analysis** : Identification of potential risk is done while risk mitigation strategy is planned and finalized
- ·**Engineering**: It includes testing, coding and deploying software at the customer site
- ·**Evaluation** : Evaluation of software by the customer. Also, includes identifying and monitoring risks such as schedule slippage and cost overrun

When to use Spiral Model

- A Spiral model in software engineering is used when project is large
- When releases are required to be frequent, spiral methodology is used
- When creation of a prototype is applicable
- When risk and costs evaluation is important
- Spiral methodology is useful for medium to high-risk projects
- When requirements are unclear and complex, Spiral model in SDLC is useful
- When changes may require at any time
- When long term project commitment is not feasible due to changes in economic priorities

Spiral Model Advantages:

Advantages:

Additional functionality or changes can be done at a later stage

- Cost estimation becomes easy as the prototype building is done in small fragments
- Continuous or repeated development helps in risk management
- Development is fast and features are added in a systematic way in Spiral development
- There is always a space for customer feedback

Disadvantages :

- Risk of not meeting the schedule or budget
- Spiral development works best for large projects only also demands risk assessment expertise
- For its smooth operation spiral model protocol needs to be followed strictly
- Documentation is more as it has intermediate phases
- Spiral software development is not advisable for smaller project, it might cost them a lot

RAD -Model

- **Rapid Application Development (RAD)**
- RAD is an iterative and incremental software development process that focuses on creating prototypes and then refining them through continuous feedback and collaboration. The primary goal of RAD is to deliver high-quality software applications faster than traditional development methodologies.

Advantages :

1. **1. Accelerated Development Process**
2. **2. Enhanced Collaboration and Stakeholder Involvement**
3. **Flexible and Adaptable**
4. **Early Detection of Issues**
5. **Cost-Effective Development**.
6. **Customer-Centric Approach**

RAD-Model

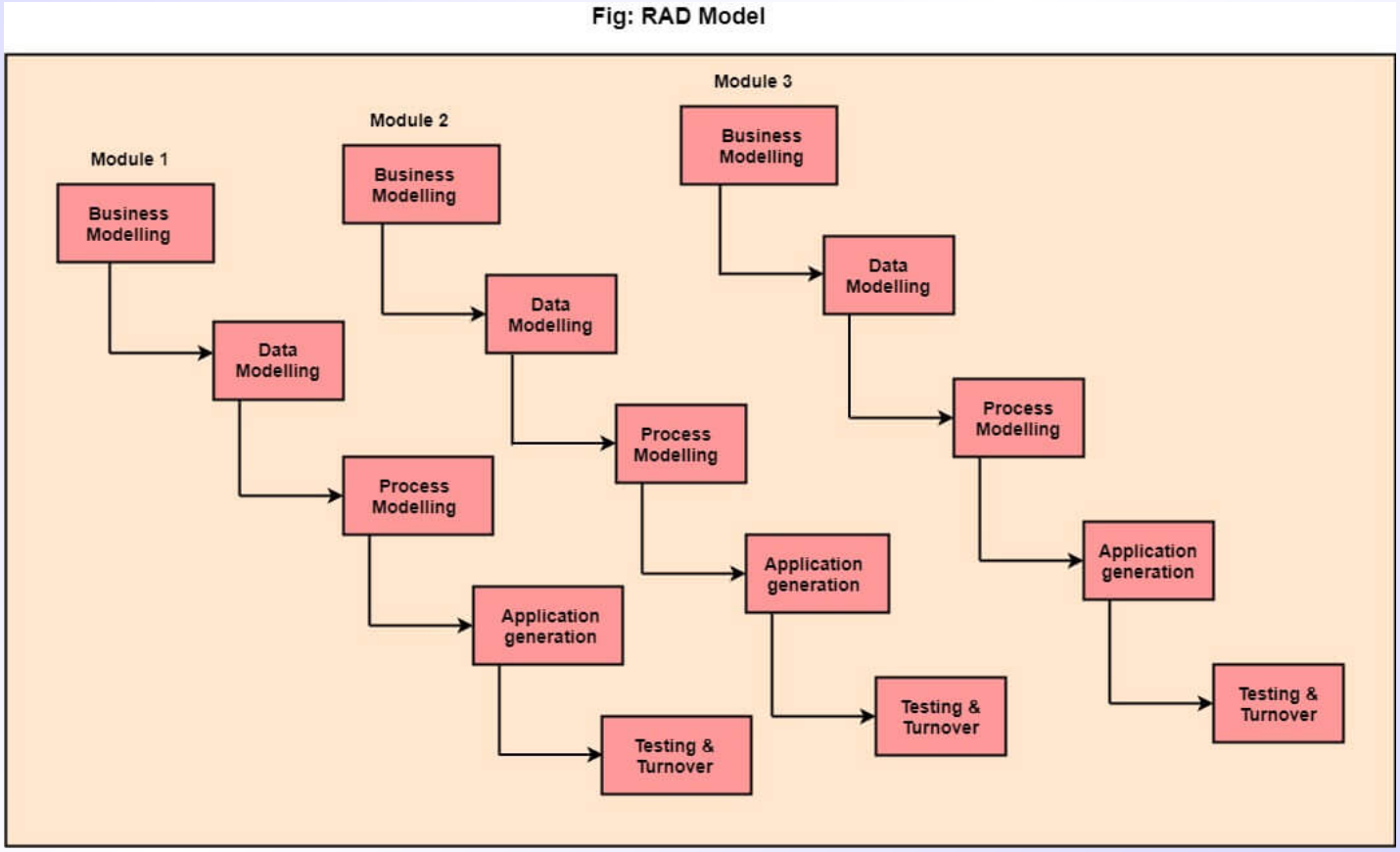
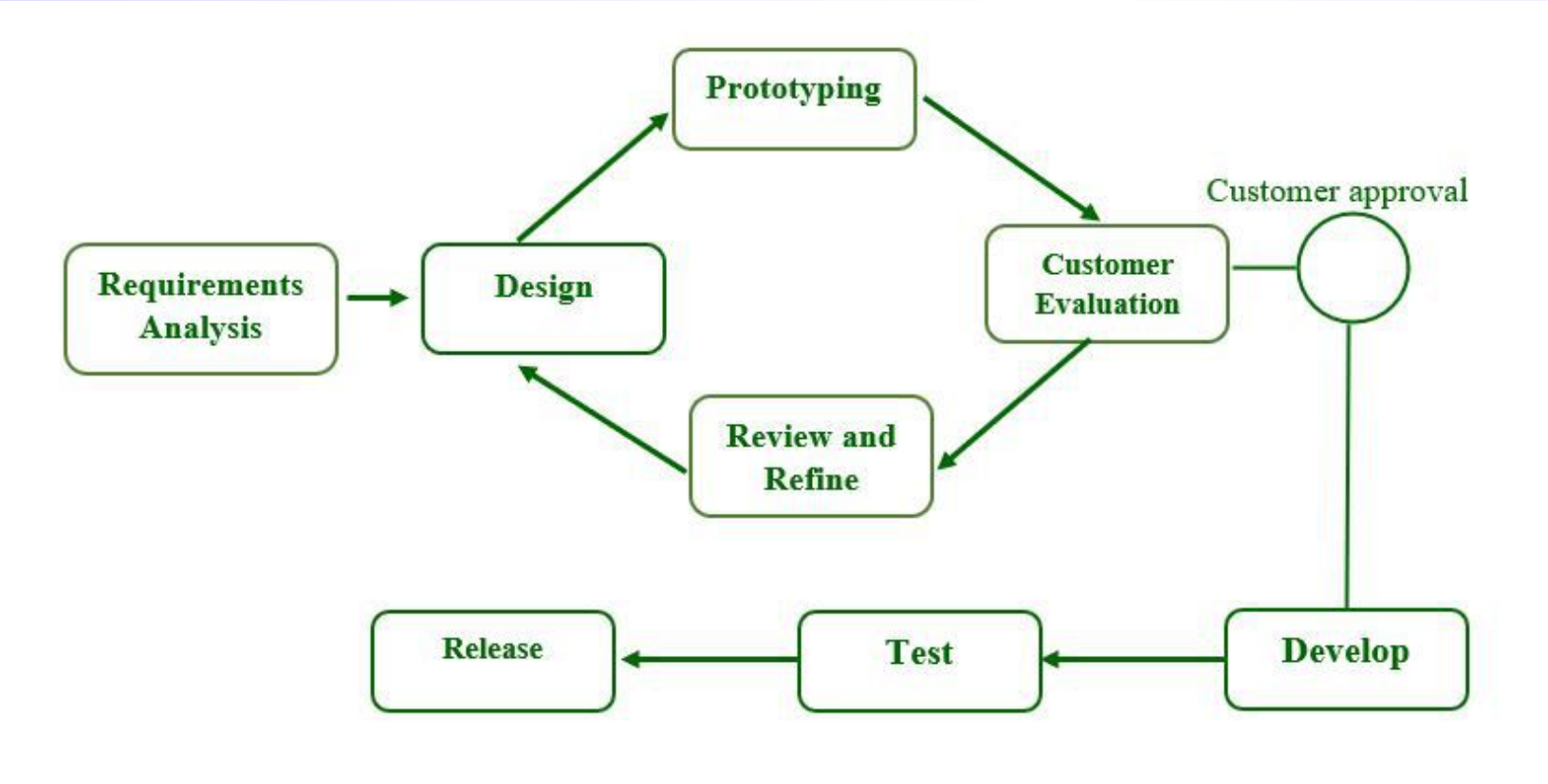


Fig: RAD Model
Module 1
Business Modelling
Data Modelling
Process Modelling
Application generation
Testing & Turnover
Module 2
Business Modelling
Data Modelling
Process Modelling
Application generation
Testing & Turnover
Module 3
Business Modelling
Data Modelling
Process Modelling
Application generation
Testing & Turnover

Disadvantages :

1. Complexity Management
2. Resource Intensive
3. Technical Debt
4. Limited Scalability
5. Dependency on User Involvement

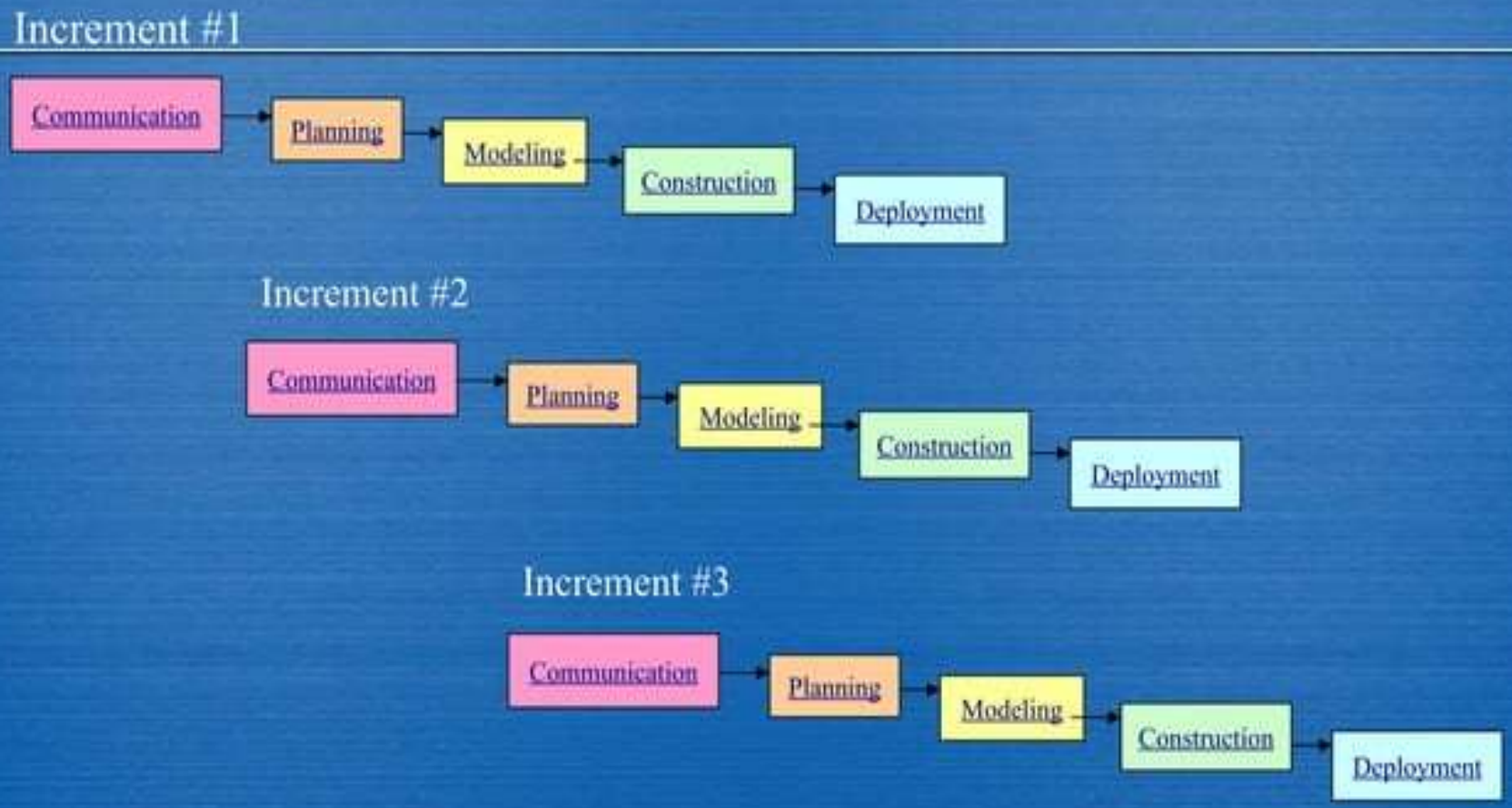
Prototyping model



Advantages and Disadvantages of Prototyping model

- **Advantages:**
- This model is flexible in design.
- It is easy to detect errors.
- We can find missing functionality easily.
- It can be reused by the developer for more complicated projects in the future.
- It ensures a greater level of customer satisfaction and comfort.
- **Disadvantages:**
- This model is costly.
- It has poor documentation because of continuously changing customer requirements.
- Customers sometimes demand the actual product to be delivered soon after seeing an early prototype..
- Customers may not be satisfied in the product after seeing the initial prototype.
- There may be incomplete or inadequate problem analysis.

Incremental Model (Diagram)



Incremental Model (Description)

- Used when requirements are well understood
- Multiple independent deliveries are identified
- Work flow is in a linear (i.e., sequential) fashion within an increment and is staggered between increments
- Iterative in nature; focuses on an operational product with each increment
- Provides a needed set of functionality sooner while delivering optional components later
- Useful also when staffing is too short for a full-scale development

12