Software Engineering

INTRODUCTION

Introduction

- **Software** is more than just **a program code**. **A program is an executable code**, which serves some computational purpose. Software is considered to be **collection of executable programming code**, associated **libraries** and **documentations**. Software, when made for a specific requirement is called **software product.**
- **Engineering** on the other hand, is all about **developing products**, using **well-defined, scientific principles** and **methods.**

Introduction

- **Software engineering** is an engineering branch associated with **development of software product using well-defined scientific principles, methods** and procedures. The outcome of software engineering is an efficient and reliable software product.
- **Definitions:**
- The application of a **systematic, disciplined, quantifiable approach** to the development, operation and maintenance of software; that is, the application of engineering to software.
- Software engineering is the establishment and use of sound **engineering principles in order to obtain economically software** that is reliable and work efficiently on **real machines.**

Need of software Engineering

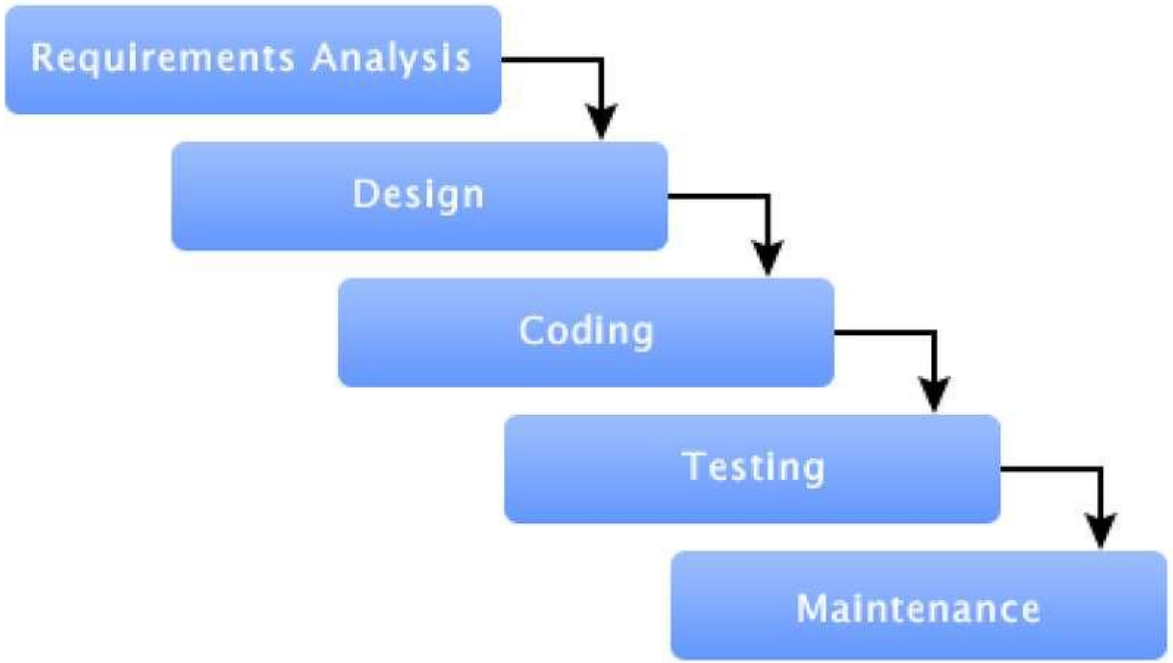
- **Large software:**
- It is **easier to build** a wall than to a **house or building,** likewise, as the **size of software become large engineering.**
- **Scalability:**
- easier to **re-create new software**
- **Cost:**
- **H**ardware, cost of software remains high
- **Dynamic nature :**
- **A**dapting nature of software
- **Quality Management :**
- software development provides **better and quality** software product.

software Engineering Process Paradigms

- **Process model (or)software engineering paradigm**
- Waterfall model
- Prototyping model
- Rapid application Development model

Waterfall model

- **Linear sequential model** or **classic life cycle** or **waterfall model.**
- Systematic ,sequential approach
- System levels :
- Requirements Analysis
- Design
- Coding
- Testing
- Maintenance



Waterfall model

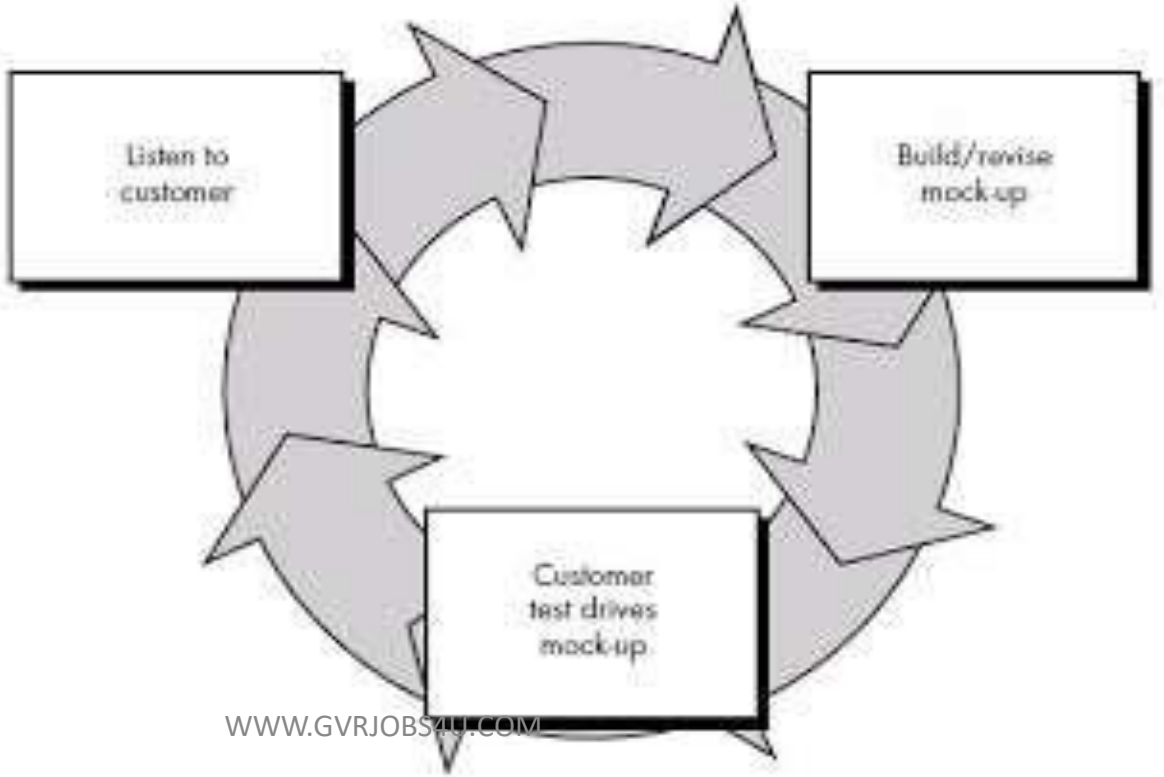
- **Requirements Analysis:**
- **->** information domain ,function ,behavioral.
- **Design** :
- ->Data structure ,software architecture ,interface representation ,algorithmic details.
- **Coding :**
- **->**program ,design is translated into machine readable form.
- **Testing:**
- **->** uncover errors ,fix the bugs ,execution of all paths.
- **Maintenance:**
- **->**longest life cycle phase.

Waterfall model

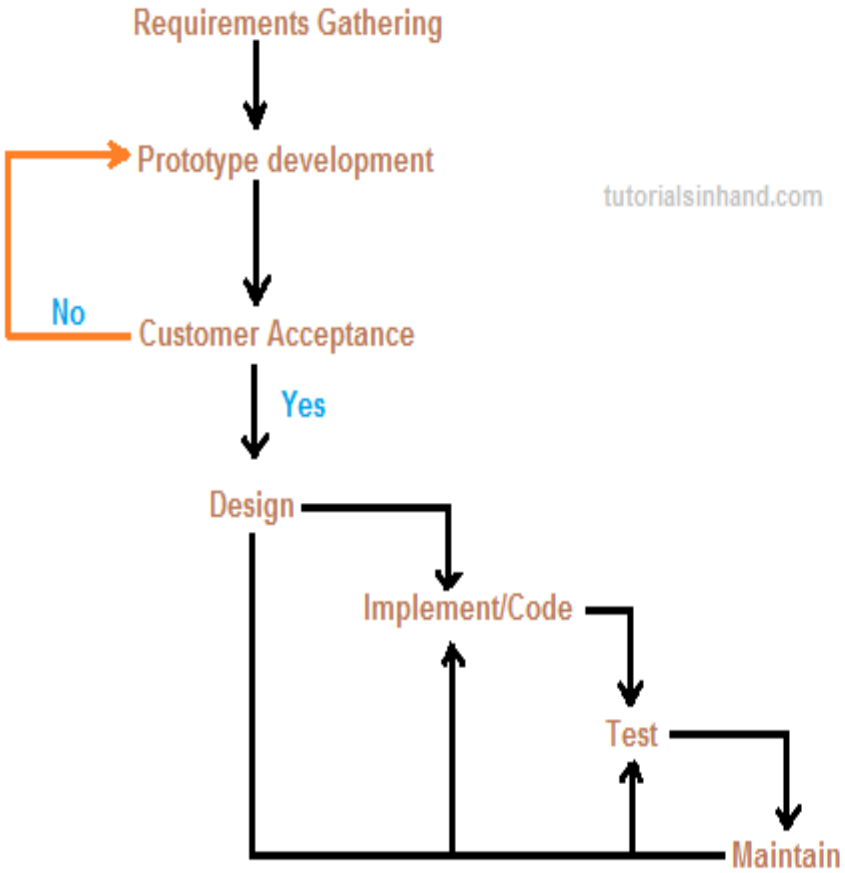
- **Advantages :**
- **Simple and easy to understand**
- Each phase has specific **deliverables and review process**
- Phases are processed and **complete one at a time** .**phase don't overlap.**
- **Smaller projects.**
- **Disadvantages :**
- Once an application is in the testing stage ,it is **very difficult to go back and change.**
- Produced **until late during the life cycle.**
- **High amount** of risk and uncertainty
- Not good for **complex and object oriented projects**
- **Poor model** for long projects
- Not suitable for moderate to **high risk of changing.**

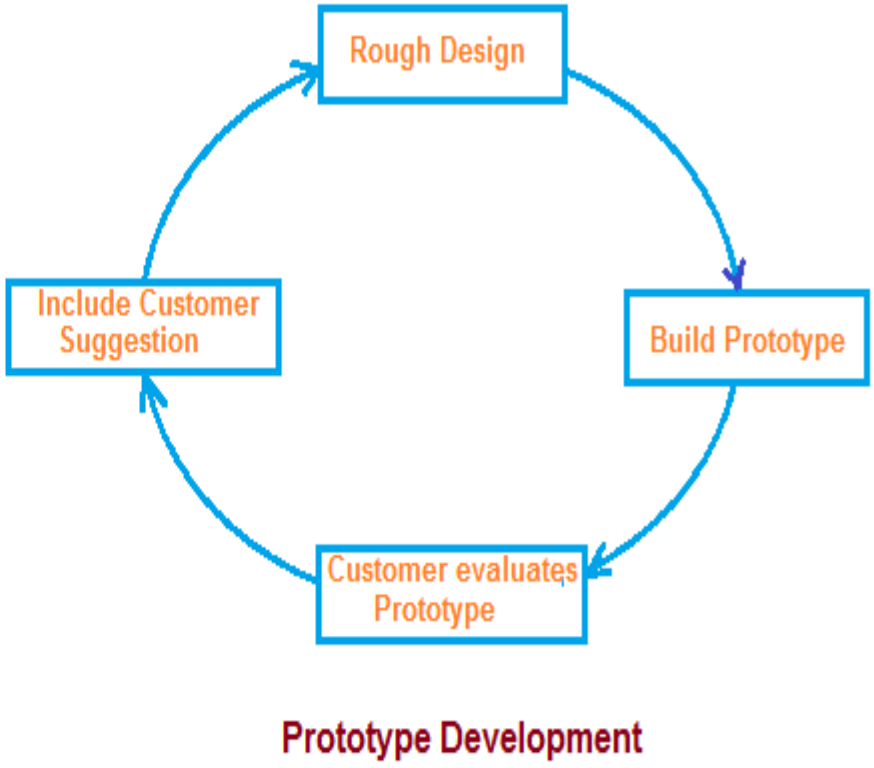
Prototyping Model

- Prototyping paradigm begins with **requirements gathering.**
- **Developer and customer meet** and define the overall objectives for the software , identity whatever requirement are known and **outline areas** .
- A **quick design** then occurs.



Prototyping Model





Prototype Development

Prototyping Model

Prototyping Model

- **Advantages:**
- **User are actively involved** in the development
- Errors can be detected **much earlier**
- Quicker user **feedback is available** leading better solutions
- Missing functionality can be **easily identified**
- Users get a **better understanding**
- **Disadvantages:**
- **Repairing** way of building system
- Practically **may increase the complexity** of the system
- **Incomplete applications** may cause applications.

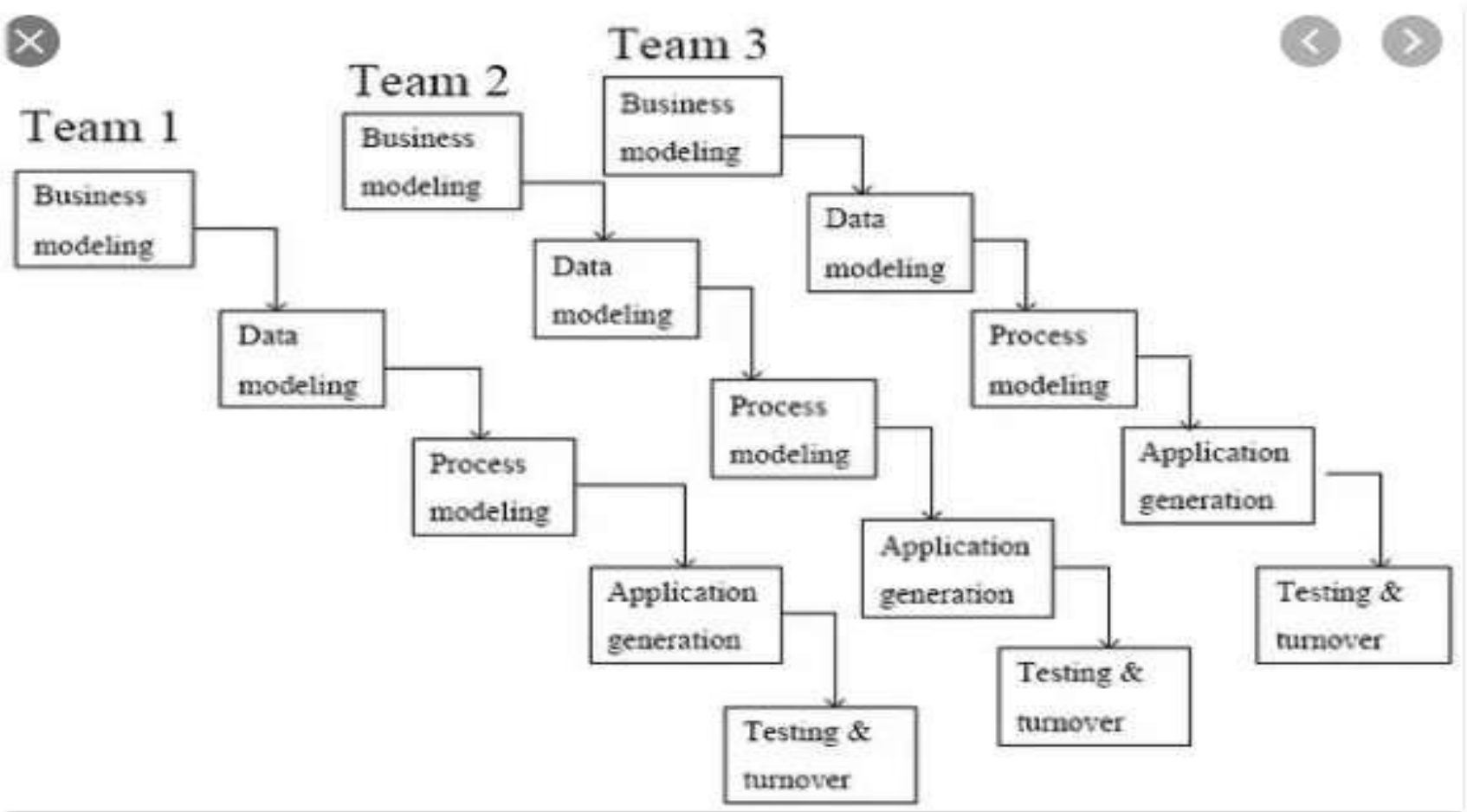
Rapid Application Development(RAD) Model

- It is an **incremental software development process model** that emphasizes an extremely **short development cycle**.
- It is a **high –speed adaptation** of the linear sequential model
- It enables a development **team** to create a **fully functional system** within **very short time periods** (e.g. 60 to 90 days).
- **The following Phases :**
- Business modeling
- Data modeling
- Process modeling
- Application generation
- Testing and turnover

Rapid Application Development(RAD) Model

- **Business modeling:** Information flow among business functions,
- **Data modeling :** data objects ,relationship between objects.
- **Process modeling**: information flow , adding ,modifying , deleting and retrieving a data objects.
- **Application generation: G**eneration programming languages.
- **Testing and turnover:** reduces overall testing time, reuses.

Rapid Application Development(RAD) Model



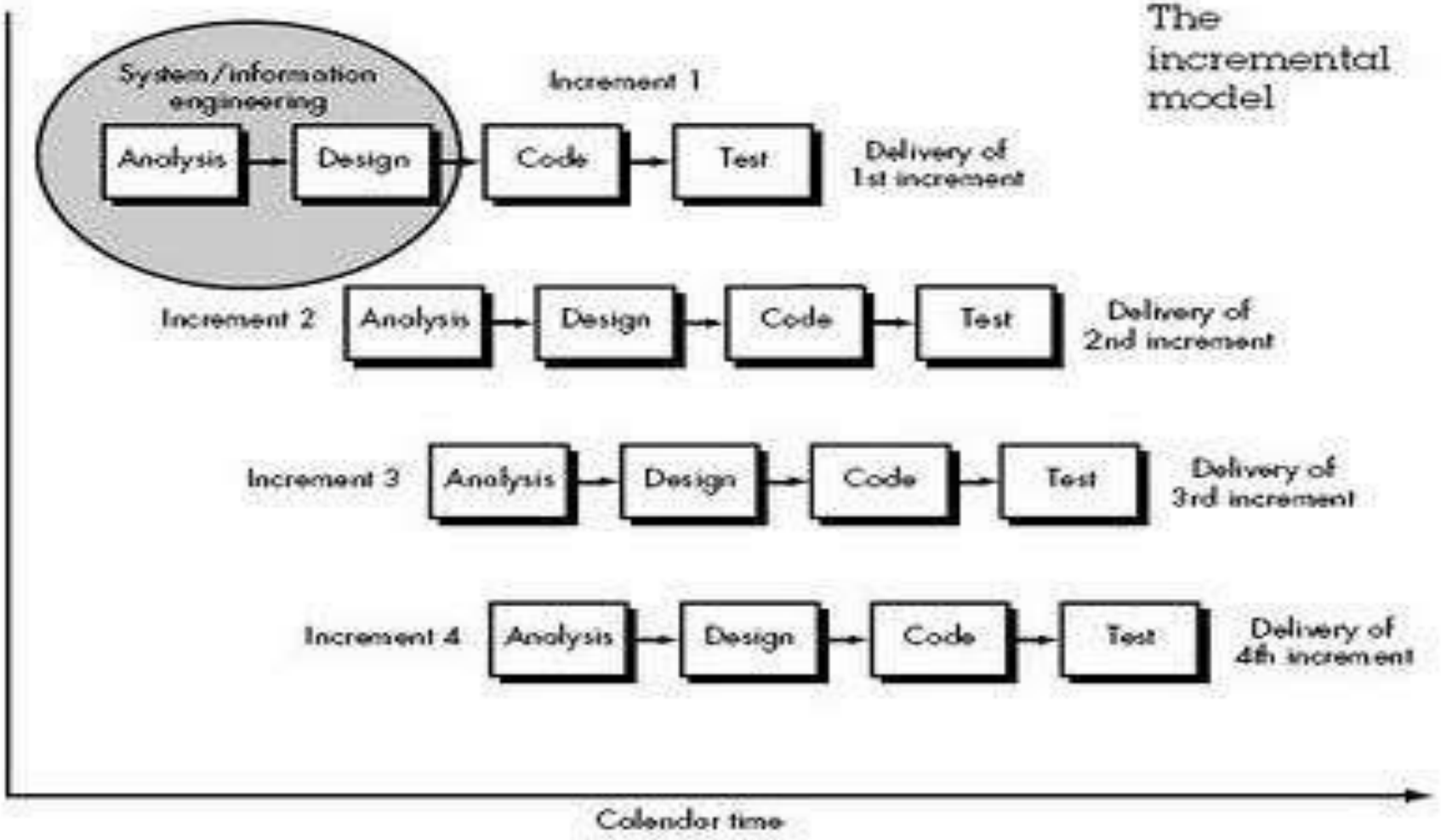
Evolutionary process Models

- Evolutionary models are **iterative models.**
- Software engineers to develop increasingly more complete **versions of the software.**
- **Incremental model**
- **Spiral model.**

Incremental model

- Linear sequential model.
- Iterative philosophy of prototyping.
- EX: word-processing ,what's app .etc
- Increment 1 :
- Analysis , Design ,code ,Test
- Increment 2 :
- Analysis , Design ,code ,Test
- ….
- Increment n :
- Analysis , Design ,code ,Test

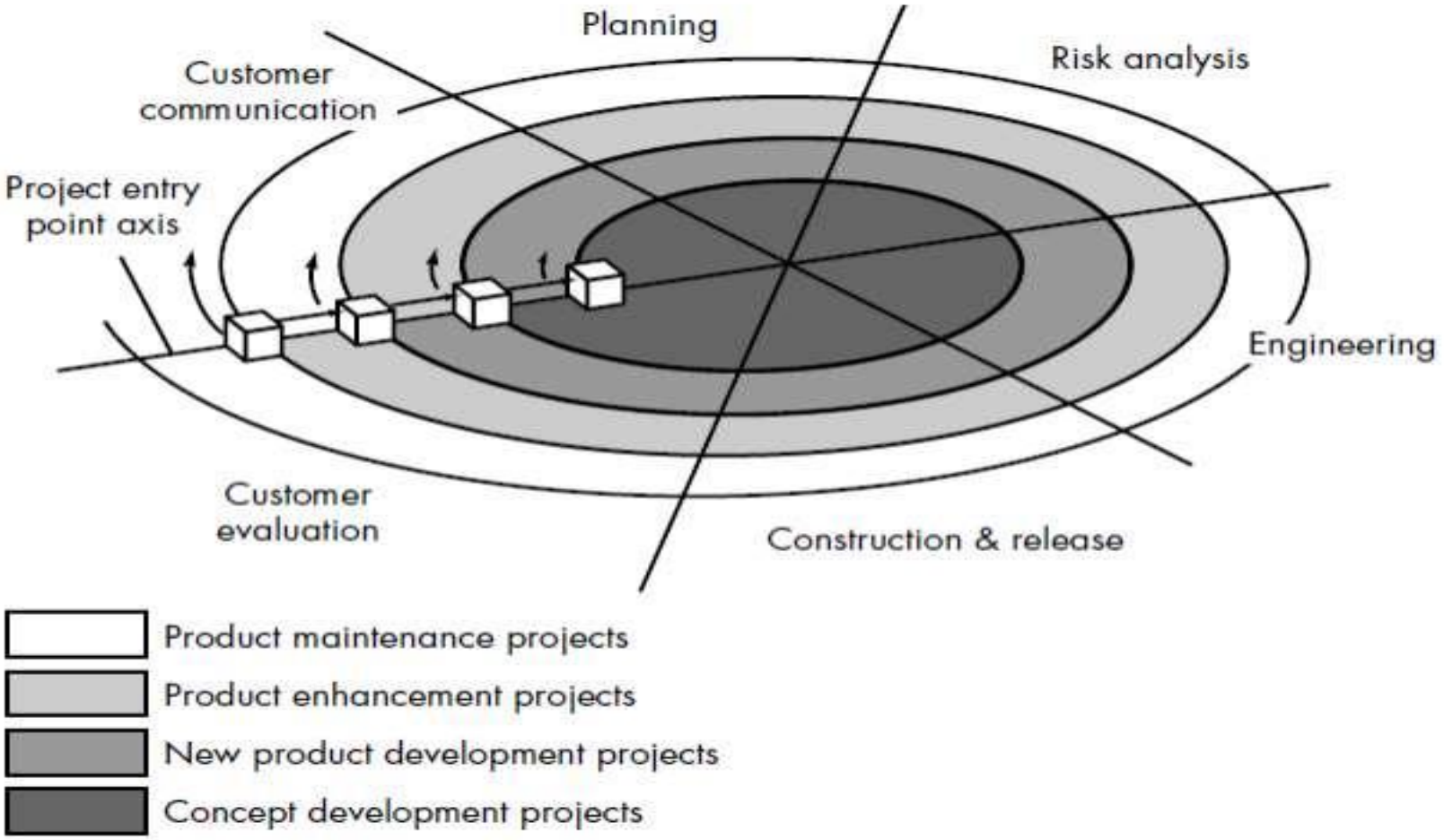
Incremental model



Spiral model

- Iterative nature.
- Incremental version of the software.
- **Phases :**
- **Customer communication:** effective communication between **developer and customer.**
- **Planning**: resources, timelines, and other project related information.
- **Risk analysis**: **technical and management risks**
- **Engineering:** build one or more representations of the **application**
- **Construction and release:** construct, test, install, and provide user support
- **Customer evaluation:** obtain customer feedback based on evaluation of the software

Spiral model



Project Management

- A Project can be **characterized as:**
- • Every project may have a **unique and distinct goal.**
- • Project is **not a routine** activity or **day-to-day operation.**
- • Project comes with a **start and end time.**
- • Project ends when its **goal is achieved**. Hence, it is a temporary phase in the lifetime of an organization.
- • Project needs adequate resources in terms of **time, manpower, finance, material, and knowledge-bank.**

Project Management

- Collection of **several operations** done in order to **achieve a goal.**
- A. Software Project
- B. Need of software project management
- C. Software Management Activities

Project Management

- A. Software Project :
- A Software Project is the complete procedure of software development from **requirement gathering to testing and maintenance**, carried out according to the **execution methodologies**, in a specified **period of time** to achieve intended software product.
- B. Need of software project management

Project Management

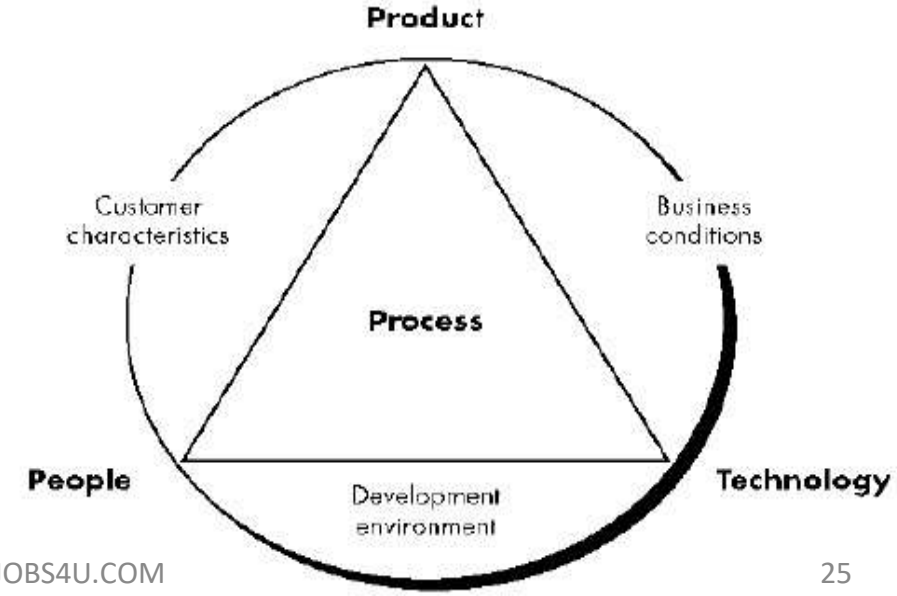
- C. Software Management Activities:
- Project management **activities** may include:
- i. Project Planning
- ii. Scope Management
- iii .Project Estimation
- **i. Project Planning :** Software project **planning is task.**
- **ii. Scope Management:**
- Define the **scope** , **Decide its verification and control** , **Divide the project into various smaller ,**
- **iii .Project Estimation :**
- •Software size estimation:
- •Effort estimation :
- •Time estimation:
- •Cost Estimation:

Process and Project Metrics

- collected across **all projects.**
- **long term software**
- Project metrics enable a **software project manager**
- (1) status of project,
- (2) risks,
- (3) uncover problem
- (4) adjust work flow
- (5) control and quality of software.

Process and Project Metrics

- Metrics are used in both the **process and project domains.**
- **A. Process Metrics:**
- **Measure specific attributes.**
- **Development environment** (e.g., integrated software tools),
- **Business conditions** (e.g., deadlines, business rules), and
- **Customer characteristics** (e.g., ease of communication and collaboration).



Process and Project Metrics

- **B. Project Metrics:**
- Project **workflow and technical activities**
- **a. Types of Project Metrics:**
- **i. Size-Oriented Metrics:**
- size of the software
- **For Example:**
- Errors per KLOC (thousand lines of code)
- Defects per KLOC
- $ per KLOC
- Pages of documentation per KLOC

Process and Project Metrics

- **ii. Function-Oriented(FP) Metrics:**
- normalization value and information domain
- **iii. Reconciling LOC and FP Metrics:**
- Relation b/w line of code and functional points
- quality of the design.
- **iv. Object-Oriented Metrics:**
- Estimate object-oriented software projects.
- **v. Use-Case–Oriented Metrics:**
- customer-level or business domain requirements.
- **vi. WebApp Project Metrics:**
- combination of content and functionality.
- **vii. Quality Metrics:**
- high-quality system
- satisfies a market need.

Software Estimation

- project estimation such as **size , effort, time and cost.**
- Project manager can estimate the **listed two techniques** –
- **1) Decomposition Technique:**
- This technique **assumption the software.**
- There are two main models -
- •**Line of Code:** Here the estimation is **number of line of codes**
- •**Function Points:** Here the estimation is **number of function points.**

Empirical Estimation Technique

- estimation model.
- empirically derived formulas to predict effort
- Such as function of **LOC or FP Values.**
- **A. The Structure of Estimation Models:**
- estimation model is using **regression analysis** on data collected from **past software projects.**
- The **overall structure** of such models takes the form
- $E = A + B \times (ev)^C$
- Where A, B, and C are empirically derived **constants**,
- E = effort in **person-months.**
- ev = estimation variable (either LOC or FP).

Empirical Estimation Technique

- **LOC-oriented estimation models proposed in the literature are**
- $E = 5.2 \times (KLOC)^{0.91}$ Walston-Felix model
- $E = 5.5 + 0.73 \times (KLOC)^{1.16}$ Bailey-Basili model
- $E = 3.2 \times (KLOC)^{1.05}$ Boehm simple model
- $E = 5.288 \times (KLOC)^{1.047}$ Doty model for KLOC > 9
- **FP-oriented models have also been proposed.**
- These include
- E = -13.39 + 0.0545 FP Albrecht and Gaffney model
- E = 60.62 x 7.728 x 10-8 FP3 Kemerer model
- E = 585.7 + 15.12 FP Matson, Barnett, and Mellichamp model
- A quick examination of different result for the **same values of LOC or FP.**

Empirical Estimation Technique

- **B. The COCOMO Model:**
- introduced by Barry Boehm
- **hierarchy** of software estimation models
- **CO**nstructive **CO**st **Mo**del
- **-> cost**
- More comprehensive estimation model, called **COCOMO II** .
- **hierarchy of estimation models** that address the following **areas:**
- **i. Application composition model:**
- consideration **of software and system interaction**, assessment of performance, and **evaluation of technology**.
- **ii. Early design stage model:**
- **once requirements** stabilized and basic software.

Empirical Estimation Technique

- **iii. Post-architecture-stage model:**
- construction of the software.
- COCOMO II models require **sizing information.**
- sizing options are available as part of the model **hierarchy:**
- **->object points,**
- **->function points,**
- **->lines of source code.**
- **The COCOMO II application**
- ->composition model uses **object points.**
- ->More sophisticated estimation models (**using FP and KLOC**)
- **Like function points, the object point** is an **indirect software measure counts** of the number of
- (1) screens (at the user interface),
- (2) reports,
- (3)components to build the application.

Empirical Estimation Technique

- **C. The Software Equation:**
- The software equation is a **dynamic multivariable model.**
- derived from productivity **data collected for over 4000 contemporary software projects.**
- Based on these data, an estimation model of the form
- $$E = [LOC \times B^{0.333}/P^3] \times (1/t^4)$$
- Where E = effort in **person-months or person-years**
- t = project duration in **months or years**
- B = "special skills factor"**16**
- P = "**productivity parameter**" that reflects:
- • Overall process **maturity and management** practices

- The software equation has **two independent parameters:**
- · an **estimate of size (in LOC)**
- · an indication of project **duration in calendar months or years.**
- To simplify the estimation process and use a **more common form for their estimation model,**

Risk Analysis

- Process of defining and analyzing the **dangers.**
- risk analysis **report** can be used to align **technology-related objectives.**
- risk analysis report can be either **quantitative or qualitative.**
- **Quantitative risk analysis:** determine the **probabilities of various adverse events**
- **Qualitative risk analysis: does not** involve numerical **probabilities or predictions of loss.** qualitative method involves defining
- the various threats, determining the extent of vulnerabilities and devising
- countermeasures should an attack occur.
- **A) Types of Risks:**
- **a. Scope Risk:**
- **> Scope creep:** project grows in **complexity as clients add** to the **requirements and developers**
- **> Integration issues**
- **>Hardware & Software defects**
- **> Change in dependencies**

Risk Analysis

- **b. Scheduling Risk:** include **unexpected delays** at an **external vendor**, **natural factors**, **errors in estimation and delays in acquisition of parts.**
- reduce scheduling risks use tools such as a
- **Work Breakdown Structure** (WBS) and
- **RACI matrix** (Responsibilities, Accountabilities, Consulting and
- Information)
- **c. Resource Risk:** risk mainly arises from **outsourcing and personnel related issues.**
- **Ex:** project requires a lot of **website front end work** and your team **doesn't have a designer skilled** in HTML/CSS, you could face unexpected delays there.
- **d. Technology Risk:** risk includes delays arising **out of software & hardware defects** or the **failure of an underlying service** or a **platform.**

Risk Analysis

- **B. Project Risk Management: all activities**
- >Experienced staff leaving the project and new staff coming in.
- >Change in organizational management.
- > Requirement change or misinterpreting requirement.
- > Under-estimation of required time and resources.
- > Technological changes, environmental changes, business competition.
- **C. Risk Management Process:**
- **> Identification:** Make note of **all possible risks**, which may occur in the project.
- **>Categorize:** Categorize known risks into **high, medium and low risk**
- **> Manage :** Analyze the probability of occurrence of **risks at various phases.**
- **> Monitor :** Closely monitor the **potential risks** and their **early symptom**

Software Project scheduling

- Refers to roadmap of all activities .
- Specific order and with in time slot
- Various tasks
- Scheduling
  - Break down tasks tin smaller
  - Find out various task
  - Estimate time
  - Divides time into work units
  - Assign adequate number of work units for each task
  - Calculate total time.

Unit-2

Requirement analysis

Requirement analysis and types

- Process of **establishing the services**
- >**customer requirement**
- -> rang from **high level abstraction** statement of services
- Two kind of requirement
- ->**user requirement:** → high level abstraction statement of services
- **>System requirement:**
- -> software system function ,services and operational constructs.
- ->system requirement document
- ->functional specification
- **Three types**
- Functional requirement
- Non functional requirement
- Domain requirement

Requirement analysis and types

- **1)Functional requirement:**
- ->system services
- ->types of software
- -> high level system
- ->services
- -> ambiguous requirement
- **Principles:**
- Complete : all facilities
- Consistent : no conflicts
- **2) Non functional requirement :**
- -> system properties ,constraints ,reliability, response time and storage Functional requirement
- ->constraints is I/O devices
- Affects all over architecture
- Security requirement
- **Three classes**
- Product requirement
- Organizational requirement
- External requirement

Requirement analysis and types

- **Product requirement:**
- >Particular way ,speed, reliability
- **Organizational requirement:**
- >policies and procedures ->process and implementation
- **External requirement:**
- >system development process
- **<u>3) Domain requirement:</u>**
- New functional requirement
- **> understandability**: language of application domain
- **> implicitness**: understand the area.
-

Requirement Engineering and phases

- Process of collecting the software requirement from the client then understanding ,evaluate and document .it is called as **requirement engineering.**
- **Seven phases(tasks)**
- **1)Inception :** task
- >asking questions
- >evaluate proper solution
- > relationship customer and developer
- **2)Elicitation :**
- >find requirement from anybody.
- >face following problems
- >problem of scope : unnecessary technical
- >problem of understanding: it is b/w customer and developers
- >problem of volatility : change from time to time different.

Requirement Engineering and phases

- **3)Elaboration :**
- >expanded and refined in elaboration
- > using functions, feature and constraints of software.
- **4)Negotiation :**
- >project will be achieved with limited business resources .
- >create rough guesses of development ,cost and delivery time.
- **5)Specification:**
- >requirement engineer constructs a final work product.
- >requirement specification
- >formalize requirement functional and behavioral.
- > both graphical and textual formats
- **6)Validation :**
- >output of the requirement
- >validation steps
- >customer stakeholder help

Requirement Engineering and phases

- **7)Requirement Management** : set of activities
- > identify, control and track requirement
- > start and unique identifier
- >finalizing the requirement
- > traceability table
- > features, source ,dependencies ,subsystems and interface.

Requirement Engineering process

- Four steps process
- >feasibility study
- >requirement gathering
- >software requirement specification
- >software requirement validation
- **Feasibility study :**
- >client approaches organization
- >desired developed.
- >rough idea
- >analysts does detailed study
- The feasibility study is focused towards goal of the organization
- >materialized
- >implementation
- >contribution
- >cost constraints
- >value of objects

Requirement Engineering process

- **Requirement gathering:**
- >gathering requirement from user
- >analysts and engineer communicate with the client and end user.
- **Software requirement specification:**
- >document created by system analysts after requirement are collected from various stakeholder.
- > Interact with
  - Hardware
  - External interface
  - Speed of operation
  - Response time of system
  - Platform
  - Maintainability
  - Speed recovery after crash
  - Security ,quality ,limitation
  - Natural language
-

Requirement Engineering process

- Features :
  - Natural language
  - Technical requirement, structured language
  - Design description written in pseudo code
  - Format of forms and GUI screens
  - Condition and mathematical notation for DFD ..etc.
- **software requirement validation:**
- >Documents are validated
- User might ask for illegal ,impractical solution or experts may interpret the requirement incorrectly.
- **Condition:**
- >Practically implemented
- >Valid and as per functionality and domain of software
- >Ambiguities
- >Complete
- >demonstrated

Requirement Engineering process

- **<u>Requirement Elicitation process:</u>**
- >**requirement gathering:** developer and client discuss
- >**organizing requirement :**prioritize and arrange.
- >**negotiation and discuss:** stakeholders
- >**documentation :** all formal and informal ,functional and non functional, requirement are documented.

Feasibility study

- Select the best system and performance requirement
- Activity is financially and technically
- Activity involves
  - Analysis of problem
  - Collection of relevant information
  - Processing required data
  - Output data require

- 3 different types :
- Technical feasibility
- Economic feasibility
- Operational feasibility

Feasibility study

- Technical feasibility :
- >specifying equipment
- >satisfy the user requirement
- >needs
  - Produce output in given time
  - Response time(condition)
  - Ability to process transaction at speed
  - Communicate data

  System requirement:

  >how many workstations are required

  >how these units are interconnected

  >operate and communicate smoothly

  >speed input and output

Feasibility study

- Economic feasibility :
  - Evaluating the effectiveness of proposed system
  - Cost/benefit analysis
  - Each phase is accuracy
- Operational feasibility:
  - Related to human organization and political aspects
  - Points
  - >what change will be brought with the system
  - >what organizational structure are disturbed
  - >what new skills will be required

  >small group of people -> system analysis and design process

  >developed and installed.

Problem of Requirement

- Requirement analysis issues:
- Stakeholder issues:
  - User do not understand ,clear idea
  - User will not commit to set of written requirements
  - User new requirement after cost and schedule fixed
  - Communication slow
  - Do not participate in reviews
  - Technically unsophisticated
  - Do not understand development process
  - Do not know present technology
- Engineer/ developer issues:
  - Possible problem
  - >different vocabularies b/w technical personnel and end user
  - >engineers and developers
  - >domain knowledge to understand

Attempted solutions:

One attempted solution

>techniques introduced in the 1990, prototyping ,unified modeling language(UML)

>new class of application simulation(recreation) or application definition tools in market.

Problem of Requirement

These tools offers:

- electronic whiteboards
- ability to capture business logic and data needs
-ability to generate high fidelity prototypes that closely imitate the final application.

Software Requirement Analysis

- Requirement analysis is a process of **discovery, refinement ,modeling and specification.**
- Both the developer and customer take an action
- Focuses on "**what"** instead of "**how".**
- Input of the requirement analysis process
  - Software plan
  - System specification (if one exists)
- Will get output requirement.
  - Translated in to data, architectural ,interface and procedure design.
  - Customer and developer check the quality and process feedback.
- System analysis:
  - Divided into five areas
  - **1) Problem recognition**
  - **2) Evaluation and synthesis**
  - **3) Modeling**
  - **4) Specification**
  - **5) Review**

Software Requirement Analysis

- **Problem recognition (or system understanding)**
  - >**discover and understand** the system requirement.
  - >**refine** the requirements
- **Evaluation and synthesis:**
  - >what are the **alternative solutions**
  - >focus on **what solution selected** or **how to implement a solution**.
- **Modeling:**
  - >it is **various aspects** of system
  - >**required data**
  - >**information and control flow** of requirements.
  - >**operational behavior** of the requirements.
- **Specification:**
  - >**software functions and performance**
  - >**interfaces b/w system elements**
  - >**system constraints**
- **Review**

Analysis Process

- **Requirement analysis process:**
- >**observation of existing system** discussion with **users and customers**, task analysis and so on
- **Domain understanding:**
  - Understanding of application domain
- **Requirement collections:**
  - Interacting with customers, users.
- **Requirement classification:**
  - Group and classify the gathered requirement.
- **Conflict resolution:**
  - Resolve the conflict requirement
- **Prioritization :**
  - Identify and list of requirement their importance.
- **Requirements validation:**
  - Check and validate.

Analysis Process

Analysis Model

- Analysis model operates b/w **system description** and **design model.**
- Analysis model, information, functions and the behavior of the system is defined
- These are translated into the architecture, interface and component level design in the design modeling .
- **Elements of the analysis model :**
- **1) scenario based element :**
  - Type of elements
  - Use case diagram ,user stories.
- **2) class based elements:**
  - Object type of element **manipulated** by the system
  - Defines **the object , attributes and relationship**
  - Collaboration is occurring **b/w the classes**
  - Elements are the **class diagram ,collaboration diagram**
- **3) Behavioral elements:**
  - Behavioral elements represent of **state of system** and **how it is changed** by the external system.
  - Elements are **sequenced diagram ,state diagram.**

Analysis Model

- **4) Flow oriented elements:**
  - **Information flows and gets transformed.**
  - How the data objects are transformed **b/w various system function.**
- **Analysis rules of thumb:**
- Creating analysis model
- **The rules are as follows:**
  - Focuses on **business domain** ,**levels of abstraction must be high.**
  - Every element in the **model helps in understanding** and **focus on** information ,functions and **behavior of the system.**
  - Infrastructure and nonfunctional model **delay in the design.**
  - **System minimum coupling is required** . Interconnections **b/w modules** is known as coupling.
  - Analysis model **gives value to all the people** related to model
  - Model should be **simple as possible**. Because simple model **always help in easy understanding of the requirement.**

Analysis Concepts and Principles

- Overall role of software in **large system engineering** is **identified** during **system engineering**.
- It must be **achieved to build high quality software.**
- Set of underlying **concepts and principles.(different jobs).**
- **Analysis concepts**
- **Analysis principles**

Analysis concepts

- **Before requirement** ->
  - analyzed,
  - modeled, or specified
  - they must be gathered through an elicitation process.
- The **3 main concepts of analysis.**
- 1) Initialization process by using **common technique**.
- 2) **Facilitated Application Specification Technique** (FAST).
- 3) **Quality Function Deployment(QFD**).

Analysis concepts

- **1) Initialization process:**
  - **Awkwardness (discomfort)**
  - **b/w two adolescents (youthful)**
- Communication must be **asking context free questions**.
- Set of questions
  - Basic understanding of the problem

- **Solution:**
  - How would you characterize “good” output
  - What problem will this solutions
  - Environment in which the solution will be useds
  - Contrains affect the way the solution is approached.

Analysis concepts

- 2) **Facilitated Application Specification Technique** (FAST).
- Customer and software engineers have an unconscious(unaware) “us and them” mind set.
- No of **independent investigators** have developed team oriented approach to **requirement of analysis and specification** called **Facilitated Application Specification Technique** (FAST).
- meeting conducted neutral site and b/w software engineer and customers
- Rules for preparation and participation are established
- An agenda is suggested
- “Facilitator” (methods) control meeting
- Definition mechanism.
- Goal is to identify the problem.

Analysis concepts

- 3) **Quality Function Deployment(QFD**):
- Translates needs of the **customer into technical requirement for software**
- QFD identifies **three types requirement**
- **Normal requirement:**
  - Meet with customers
- **Expected requirement:**
  - Implicit(understood) to the product or system.
- **Exciting requirement:**
  - customer expectations
- Functional deployment:
  - Value of each requirement
- Information deployment:
  - Identify data objects and events
- Task deployment:
  - Behaviour of the system

Three are value of analysis

Analysis concepts

- 4) **Use Cases**:
  - Part of informal meeting
  - Software engineer can **create set of scenarios** that identify usage.
  - **Actor** represent **class of external entities** that play just one role.
  - One actors identify use case developed -> actor interact the system.

Analysis principles

- Unique point of view
- Set of operational principles
  - Understand the information domain
  - Define function
  - Behaviors
  - Depict(show) information ,function and behaviors →uncover details
- Set of guidelines
  - Understand the problem before beginner to create
  - Help user to understand
  - Record ,reason every requirement
  - Prioritize requirement
  - Eliminate ambiguity
- We are using 3 analysis principles as follows
- 1)information domain
- 2) modeling
- 3)partitioning

Analysis principles

- **1)information domain:**
- Built to process data to transform data from one form to another
- Information domain contains 3 different views
- 1) Information content and relationship:
  - Individual data and control objects → b/w data and objects
- 2) Information flow:
  - data and control changes ->b/w two transformation(functions)
- 3) Information structure:
  - Various data and control items
  - Data tree structure and data table (n- dimension)
- **2) modeling:**
- **Create models of the system to be built**
  - Focus on what the system do ,not how it does it
  - Graphic notation
  - Build models of function and behaviors
  - Three generic function s i.e. input, processing ,output.
  - Behavior models (state and events)

Analysis principles

- Important roles of models:
  - Model aids(help) analyst understanding the information and behavior system
  - Focal (main)points for review in the aspects
  - Foundation for design ,providing the designer.
- **3) Partitioning :**
- Establish a hierarchical representation of information or function
- Exposing increasing details -> vertically in the hierarchy
- Decomposing the problem -> horizontally in the hierarchy
- **Divided into 3 types**
- Configure system
- Interact with user
- Monitor sensors
  - Sensor event
  - Active alarm function s
    - Activate audible alarm and dial phone numbers