UNIT-I.

① SDLC.
- Requirement Analysis.
- Design.
- Coding
- Testing
- Deployment & Maintainence

② SDLC
- Traditional
  - Waterfall
  - V Mode
  - Prototype
  - Spiral
  - RAD
- Conventional
  - Agile.
  - XP.
  - SCrum.

③ functional V/S Non-functional

④ Effort & cost in software project → Static single variable, Static Multivariable.

⑤ watson & Felix Model with SEL Model.

⑥ COCOMO Model produced by "Boehm".

⑦ Types of project in COCOMO.

i) Effort: Total Effort to dev. Software product
ii) Tdev: - Estimated time to dev. Software

- (i) Organic: equation: Effort = 2.4 (KDLOC) 1.05 PM; Tdev = 2.5 (Effort) 0.38 Month
- (ii) semidetached: eqn: Effort = 3.0 (KLLOC) 1.12 PM; Tdev = 2.5 (Effort) 0.35 Month
- (iii) Embeded: eqn: Effort = 3.6 (KDLOC) 1.20 PM; Tdev = 2.5 (Effort) 0.32 Month

⑧ Basic of cost estimation can be done by:-
1. Basic Model. → Effort: $a * (KLOC)^{a_2}$ PM, TDEV: $b_1 * (effort)^{b_2}$ Month
2. Intermediate Model.
3. Detailed Model.

→ Effort: $a_i (KLOC)^{b_i} EAF$
→ D: $c_i (Effort)^{d_i}$

Effort Adjustment Factor

⑨ Risk Management: Project Risk, Technical Risk, Business Risk, Known Risk, Predictable Risk, unpredictable risk.

Risk Management
- Risk Assesment → Risk identification, Analysis, Prioritization
- Risk control
  - Risk Management planning
  - Risk Monitoring
  - Risk Resolution.

⑩ Software Configuration Management.

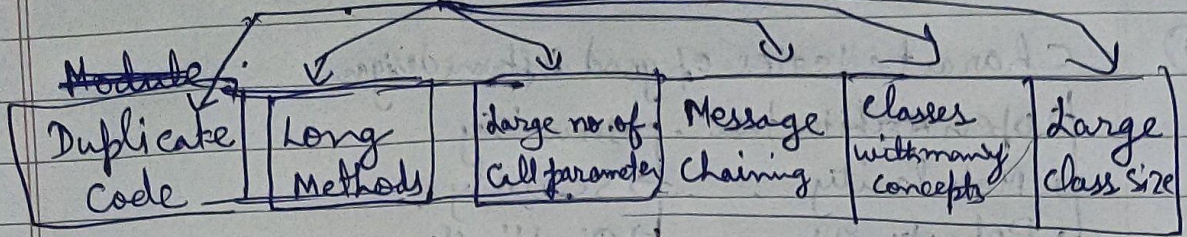
classmate
Date
Page

Unit ⇒ 2

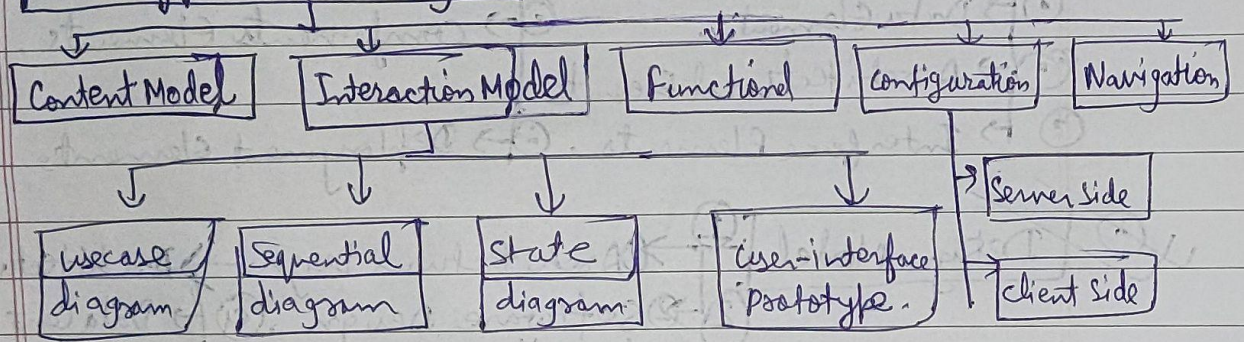
- Software Design.
- Characteristics of good Software design.
  - Open Architecture.
  - Modularity.
  - Robustness
  - Security
  - Scalability.
  - Simplicity.
- Design types / technique
  - Prototypes
  - Design Reuse
  - Structural Modeling
  - System Analysis
  - Object oriented modeling
  - Entity Relationship Models.
- Analysis Model → Design model
- Design Model element
  - Data Element.
  - Architectural Elements
  - Interface Elements.
  - Components Elements.
  - Deployment Elements.
- Design Model
  1. Architectural Design → Architectural Style.
     - (a) Data Centered Architecture
     - (b) Data Flow Architecture
     - (c) Call & Return Architecture
     - (d) Object oriented Architecture
     - (e) Layered Architecture
  2. Software Design.
     - (a) Top down.
     - (b) Bottom up.
- Architectural Pattern
  - concurrency.
  - Persistence
  - Distribution.
- Architectural context.
- Arche type.
- Component structure.
- Refined component structure
- Analyze Architectural Design.
  - Architectural complexity
  - Architectural description Language (ADL).
- Architectural Design Method.
  - Horizontal Partitioning
  - Vertical Partitioning - factoring

UNIT → 2

- Module Division (Refactoring)



- Module coupling
- Component Level Design
- User Interface Design
- Pattern oriented Design
- Web Application Design



- Design Reuse
- Concurrent Engineering
- Design-Life cycle Management

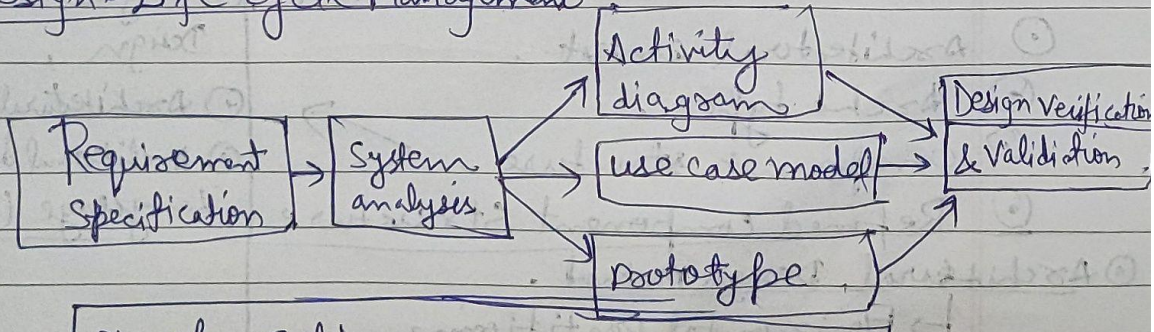


Fig of Software Design Life cycle

Unit ⇒ 3

Software Construction

- (a) → Coding Standard
- (b) → Coding Framework
- (c) → Reviews
- (d) → Coding Method
- (e) → Configuration Management
- (f) → Software construction Artifacts

(a) Coding Standard (RCMMSS)

- (R) Reliability
- (C) Clarity
- (M) Modularity
- Maintainability
- (S) Safety
- Simplicity

(b) Coding Framework

(c) Reviews

- Quality control
- Desk check
  - Peer review
- Walk through
- Code Reviews
- Inspections

(d) Coding Method

1. Structured Programming.
2. Object-oriented Programming.
3. Automatic code generation.
4. Software Code Reuse.
5. Pair Programming.
6. Test-Driven Development.

Software Code Reuse →
- (a) Libraries.
- (b) Open source.
- (c) Software as a service.
- (d) Inheritance.

Unit - 4.

Software Testing

(1) Introduction = Problem in Traditional.

(2) Testing = (A) Verification (Static testing).
(B) Validation (Dynamic testing).

(3) Test Strategy & planning = (A) Test Prioritization [by using test case]
(B) Risk Management → (i) Project-Size.
(C) Effort Estimation → (ii) Test-strategy.
(D) → Test Point Analysis → (iii) Productivity (Knowledge base)

(4) Effort Estimation [Methods Include].

(A) Test point Analysis | (B) wideband Delphi Technique | (C) Experience Based estimation (Quality + Customer strategy)

(C) Not in syllabus

no. of function point → (i) → Project size } after Execution.
(ii) → Test Strategy }
(iii) → Productivity } experience + Knowledge.

(5) Test Project Monitoring and control

(6) (A) Test Case Design. (pdf)

(B) Test Case Management.

(C) Test Bed Preparation.

(D) Test case Execution.

(E) ~~Defect Tracking~~.

(G) Test case Reporting.

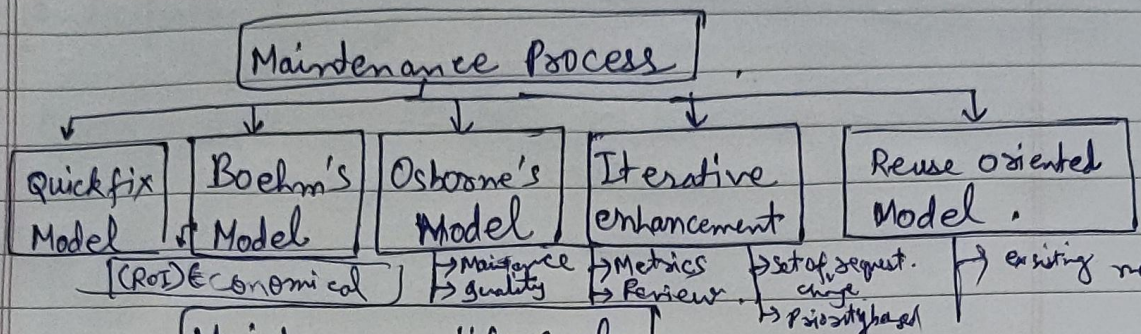
(F) Defect Tracking ⇒ Defect Login → Assign defect → Fix defect → Defect verification → Defect closure

Unit - 5

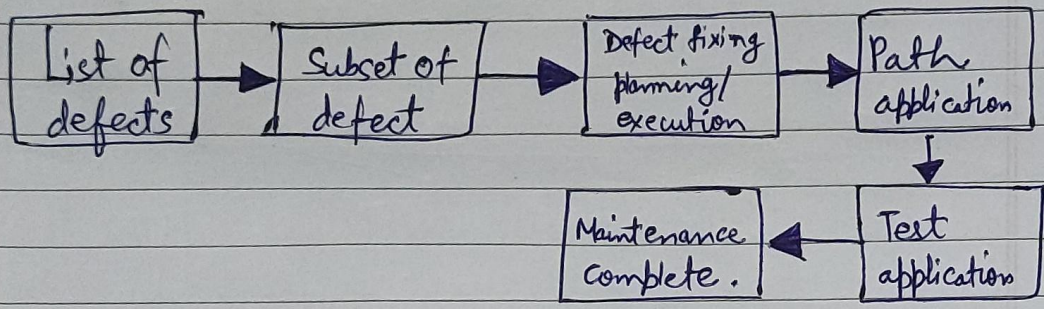
1. Product Release
   - Estimate cost of Providing support
   - Version selection
   - Decision for release
   - Walk around
   - Trained to staff
   - customer satisfaction.
2. Product Release Management
   - (a) → Alpha Release
   - (b) → Beta Release
   - (c) → Internal release.
   - (d) → Normal release.
3. Product Implementation.
4. Check Software Interface | Check Hardware Interface | Create master Data | Create test Data | Create user account | Create infrastructure for installation.
5. User Training.
6. Maintenance Introduction, (why?)
   - (a) Technology obsolescence
   - (b) Software defects
   - (c) Change in user requirement
   - (d) ⇒ Reason for Maintainance
7. ⇒ Reason for Maintenance in Software
   - (A) Software Defect
   - (B) New user Requirement
   - (C) Changed user requirement
   - (d) Technology obsolescence
   - Better technology
8. Maintenance Types.
   - (A) Corrective → patch
   - (B) Preventative → Hardware/software error
   - (C) Perfective → change in Business
   - (d) Adaptive. → API outdated
9. Maintenance Cost → (A) Revenue Loss (B) Opportunity Loss (C) Productivity Loss
10.

Unit-5

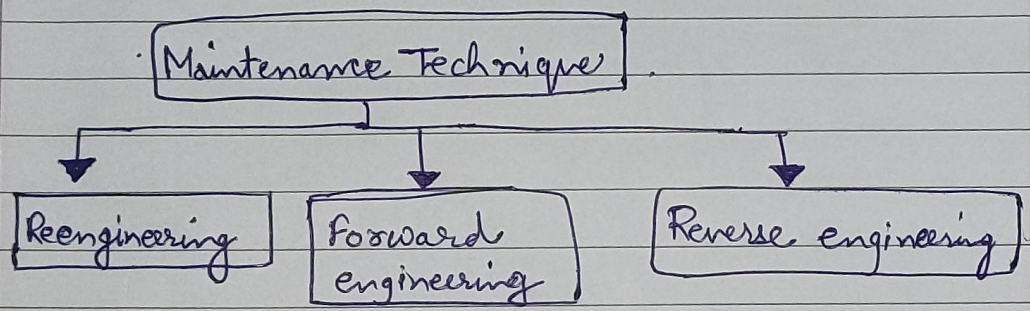
(8) Maintenance Process.



(9) Maintenance life cycle.



(10) Maintenance Technique.



(11) Software Release - Case Study.

(12) Software Maintenance - Case Study.