

## \* Good v/s Poor SE :-

Good SE	Poor SE
<ul style="list-style-type: none"><li>1. Systematic &amp; well structured</li><li>2. High Security &amp; Better Data Protection</li><li>3. Easy maintenance &amp; Future Scalability</li><li>4. Low development cost &amp; optimized performance</li></ul>	<ul style="list-style-type: none"><li>1. Unstructured &amp; Random Coding</li><li>2. Security loopholes &amp; hacking risks</li><li>3. Hard to maintain &amp; no scalability</li><li>4. High development cost &amp; inefficient performance.</li></ul>

## "Software Engineering"

- ऐसी engineering जिसमें Computer Systems या कोई श्री electronic device के लिए Software develop किया जाता है।
- इसमें user की Requirements को Analyze कर उनके According software develop किया जाता है।
- Father ⇒ Barry Boehm

## \* Need of SE =>

1. बड़े Software को Manage.
2. उम्मादा Scalability के लिए।
3. Cost को manage करने में।
4. Software के dynamic nature को Manage करने में।
5. Quality Management में।

## \* Characteristics of SE →

### 1. Large Software →

→ Real life में किसी घर की बजाय एक दीवार बनाना easy होता है, तो इसी प्रकार जब कोई software बड़ा हो जाता है, तो SE उसे develop & manage करने में help करती है।

### 2. Cost →

→ SE की process को follow करके software की cost को कम किया जा सकता है।  
→ SE सही process को follow करने के बारे में बहाता है।

### 3. Dynamic Nature ⇒

→ Software का nature dynamic & flexible होता है, इसलिए Time according उसमें update करने की need होती है, जिसमें SE का बड़ा Role है।

### 4. Quality Management ⇒

→ Software को बनाने की methods & Quality दृग्दशा better होनी चाहिए, इसमें SE का बड़ा Role है।

### 5. Scalability ⇒

→ किसी भी Software को develop करने में Best Scientific & Engineering Concepts का use करने से future expansion में problem नहीं होती है।  
→ Demand & load को handle करने वाला।

## \* Software Testing =>

→ किसी software को execute करने की process, जिसका Main objective bugs को findout करता होता है।

## \* Types of Software Testing =>

### 1. Unit Testing =>

→ Component Testing / Module Testing / Functional Testing

→ एक प्रकार की functional Testing जिसमें software की प्रत्येक unit / component को Test किया जाता है।

→ Coding phase में

→ Programmer द्वारा

## 2. Integration Testing ⇒

- सभी units/components को एक साथ combine कर Test करना।
- Main Objective = सभी units एक साथ कैसे function करेगी।
- 4 तरीके ⇒

A. Top down

B. Bottom Up

C. Sandwich → Topdown + Bottomup (Hybrid)

D. Big Bang → सभी units को एक साथ एक group में combine कर Test करना

- इसमें किसी एक unit को नब तक integrate नहीं किया जाता है,
- जब तक सभी units Ready नहीं हो जायें।
- Risky
- छोटे systems के लिए Best.

### 3. Alpha Testing

- लघसे ज्यादा बार की जाने वाली Testing.
- किसी Software की users के लिए Release करने से पहले उसे error free व issue free करता।
- इस Testing के लिए Coding Knowledge दोगा जरूरी होता है, इसलिए मह Testing Software Engineers करते हैं।
- Acceptance Testing
- Quality Assurance Team Verification करती है।

Q \*which Environment is used  
to do alphatesting?

A. Developer Environment

~~B. User Environment~~

~~C. Quality Environment~~

#### 4. Beta Testing ⇒

- Customers के द्वारा Testing.
- इसमें Software का केवल एक Version Limited users के लिए Launch किया जाता है, जिसे Use कर Feedback देते हैं।
- इन feedbacks के अनुसार Software को Error free कर Stable Version Launch किया जाता है।

## \* Alpha V/S Beta Testing ⇒

Aspect	Alpha	Beta
1. Objective	Identify internal errors & validate core functionalities	Gather real world feedback & validate usability
2. Participants	Software Developers & Quality Assurance Team	External Users (Limited)
3. Focus	Functionality, Performance, Security	User Experience, Compatibility
4. Timing	Before Beta Testing	Before Final Release
5. Methods	White & Black Box Testing	Black Box & Exploratory Testing
6. Risk	Low Risk & Controlled	Higher Risk due to large audience

## 5. Stress Testing ⇒

- Software की stability & Reliability को Verify करता है।
- कोई Software कितना Load सहन कर सकता है या किस Load पर Crash हो जाएगा।
- Heavy load = बहुत छारे programs को एक साथ open करना।
- Non Functional Testing
- Endurance Testing / Torture Testing
  - ↓
  - सहन शक्ति

## 6. Recovery Testing ⇒

- इस Testing में मैं validate किया जाता है, कि जब कोई Software Crash हो जाता है, तो क्यों Recover करेगा या नहीं।
- Testing Team हुआ perform.

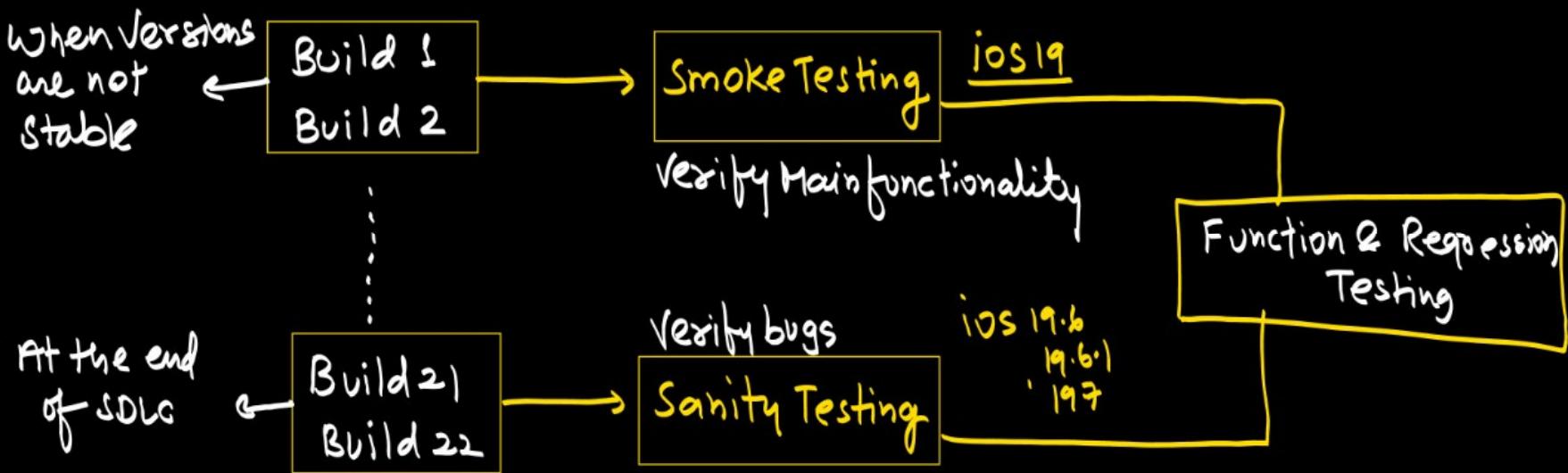
## 7. Security Testing ⇒

- इस Testing में मैं check किया जाता है, कि Software Data को Secure रख सकता है या नहीं।
- इसमें भृगु भी check किया जाता है, कि Software hack हो जाने पर मैं data की security को कैसे maintain करेगा।
- Non functional Testing.

## 8. Smoke Testing →

→ Build Verification Testing

→ उस development team एक कोड build करती है, तो software Testing team उस build को validate करती है वह check करती है, कि वह error free हो।



## 9. Regression Testing ⇒

- Software में किये गये changes की वजह से उलझी working में कोई issue नहीं आ रहा है,
- Testing Regression Testing होती है।
- Repeatedly Testing

## 10. System Testing ⇒

- बहुत सारा जग्या software Maximum Operating Systems पर work करता है या नहीं।
- Black box Testing

## 11. Performance Testing ⇒

- Performance engineer हुआ।
- इसका Use Runtime performance को Test करने में किया जाता है।
- इसमें Software की speed & effectiveness को Test किया जाता है।

## 12. Ad-hoc Testing ⇒

- बिना किसी planning & documentation के perform की जाती है।

## \* Black v/s white v/s Gray Testing →

Aspects	Black	White	Gray
1. Knowledge of Code	No Knowledge	Full Knowledge	Partial Knowledge
2. Focus	Functionality & Behaviors	Internal Code & logic	Combination of Both
3. performed By	Testers, Quality Assurance Engineers	Developer	Testers with some knowledge
4. Testing Approach	Input-Output Validation	Code path Analysis	Both functional & structured Testing

\* SDLC ⇒

(I)

→ Software Development Life Cycle / System Development Life Cycle

→ मट किसी भी Software की Life cycle को Explain करता है।

→ 7 Steps ⇒

1. Requirement & Analysis

2. Feasibility Study

3. Design

4. Coding

5. Testing

6. Deployment / Installation / स्थापना / परिनियोजन

7. Maintenance

## 1. Requirement & Analysis ⇒

- इस Phase में software को उसा achieve करवारा चाहते हैं, उसकी details gather की जाती है।
- इस Phase में software project के बारे में details, software की Requirements, Software के कार्य, Software का use, Software के Aspects etc. की एक clear picture तैयार हो जाती है, जिसके Base पर Feasibility Study की जाती है।

## 2. Feasibility Study ⇒

- इस Phase में Requirements की detailed study की जाती है तथा software project के success होने की possibility को check किया जाता है।
- यदि Requirements को Apply करना possible नहीं हो, तो customer के Requirements के Modification करवाया जाता है।
- Cost Analysis of project.

\* SDLC ⇒

(I)

→ Software Development Life Cycle / System Development Life Cycle

→ यह किसी भी software की life cycle को explains करता है।

→ 7 steps ⇒

1. Requirement & Analysis

2. Feasibility Study

3. Design

4. Coding

5. Testing

6. Deployment / Installation / स्थापना / परिनियोजन

7. Maintenance

## 1. Requirement & Analysis ⇒

- इस Phase में software को उस achieve करवारा चाहते हैं, उसकी details gather की जाती है।
- इस Phase में software project के बारे में details, software की Requirements, Software के काम, Software का use, Software के Aspects etc. की एक clear picture तैयार हो जाती है, जिसके Base पर Feasibility Study की जाती है।

## 2. Feasibility Study ⇒

- इस Phase में Requirements की detailed study की जाती है तथा Software project के success होने की possibility को check किया जाता है।
- यदि Requirements को Apply करना possible नहीं हो, तो customer से Requirements में Modification करवाया जाता है।
- Cost Analysis of project.

→ Feasibility study का objective problem को solve करना नहीं होता है जबकि पहले decide करना होता है कि problem solve को सकारी है या नहीं।

→ 3 प्रकार-

- A. Technical Feasibility
- B. Operational Feasibility
- C. Economical Feasibility

### 3. Design ⇒

→ System / Software का blueprint.

→ इस Phase में system के Complete Technical Specifications का use करके system को design किया जाता है।

→ design में system efficiency & interactivity का पूरा ध्यान रखा जाता है।

#### 4. Coding =>

- Software design phase के बाद Software को develop / code करना start किया जाता है।
- इस Phase में developers / coders Software की Coding का काम करते हैं।
- इस Phase में Real Software तैयार हो जाता है।

#### 5. Testing =>

- इस phase में Real software & उसकी database design को Test & debug किया जाता है, ताकि system की performance & functionality & quality best हो।
- इसी phase में Code की Verification & Validation भी होता है।
- Long lasting phase.

## \* Verification v/s Validation $\Rightarrow$

Verification	Validation
<ol style="list-style-type: none"><li>1. Performed by Quality Assurance Team.</li><li>2. Application &amp; Software का Architecture, customer specifications के According हैं या नहीं।</li><li>3. Right product</li><li>4. यह product के documentation, design &amp; coding को check करता है।</li><li>5. White Box Testing</li></ol>	<ol style="list-style-type: none"><li>1. Executed by testing team.</li><li>2. Focuses on validating the actual Product.</li><li>3. Product is right.</li><li>4. यह product की functionality को test करता है।</li><li>5. Black box testing</li></ol>

## 6. Installation / Deployment / Implementation ⇒

→ Completed Software / System को user end / customer end पर deploy करना।

## 7. Maintenance ⇒

→ डॉउट System को user end पर install कर दिया जाता है और user उसे use में लेना start कर देता है तो Time to time जोन वाली errors को timely correct / Remove करना पड़ता है।

→ long lasting process at user end.

### Types ⇒

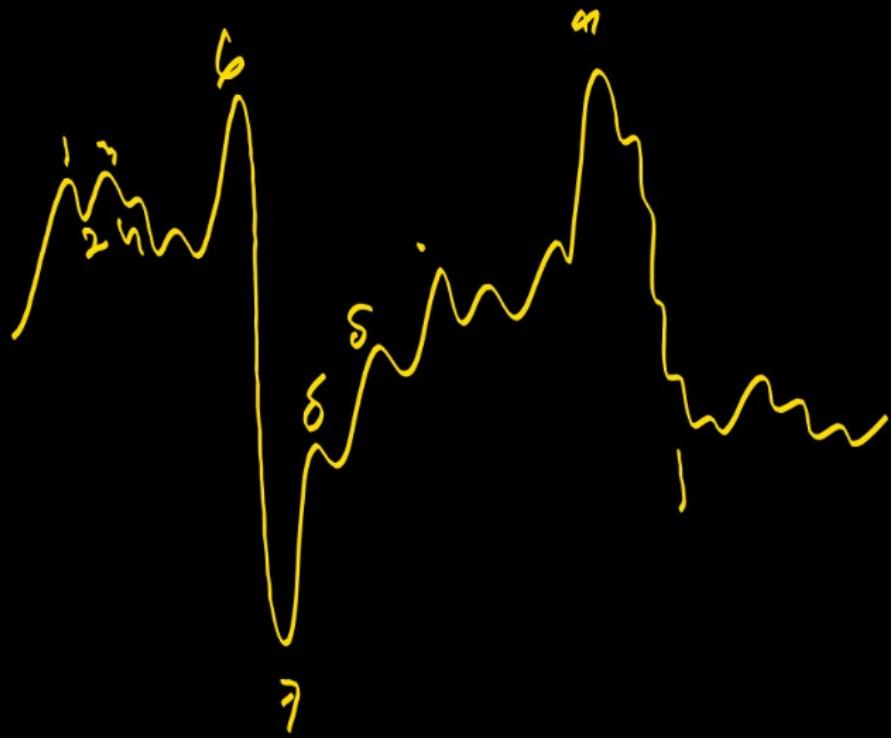
- A. Corrective
- B. Adaptive
- C. Perfective
- D. Preventive

\* SRS →

- Software Requirements Specification
- किसी software को develop करने के लिए जहरी Requirement का document or description.
- ऐसा document जिसमें पृष्ठ describe किया जाता है कि software perform कैसे करेगा, इसका behaviour क्या होगा, इसके features क्या - क्या होंगे।
- contract between client & development team.
- इसका प्रयोग करने से SDLC में मूल Time बढ़ाता है।
- Client product बनने के बाद SRS से Analysis कर सकता है, कि product as per the requirements है या नहीं।

## \* Characteristics of SRS =>

1. Complete
2. Correct
3. Clear
4. Accuracy
5. Consistency - Start से End तक
6. Verifiable (सत्यापन योग्य)
7. Modifiable (परिवर्तन योग्य)
8. Traceable - Source से Connected
9. Testable (परिक्षण योग्य)
10. Unambiguous - clear Meaning

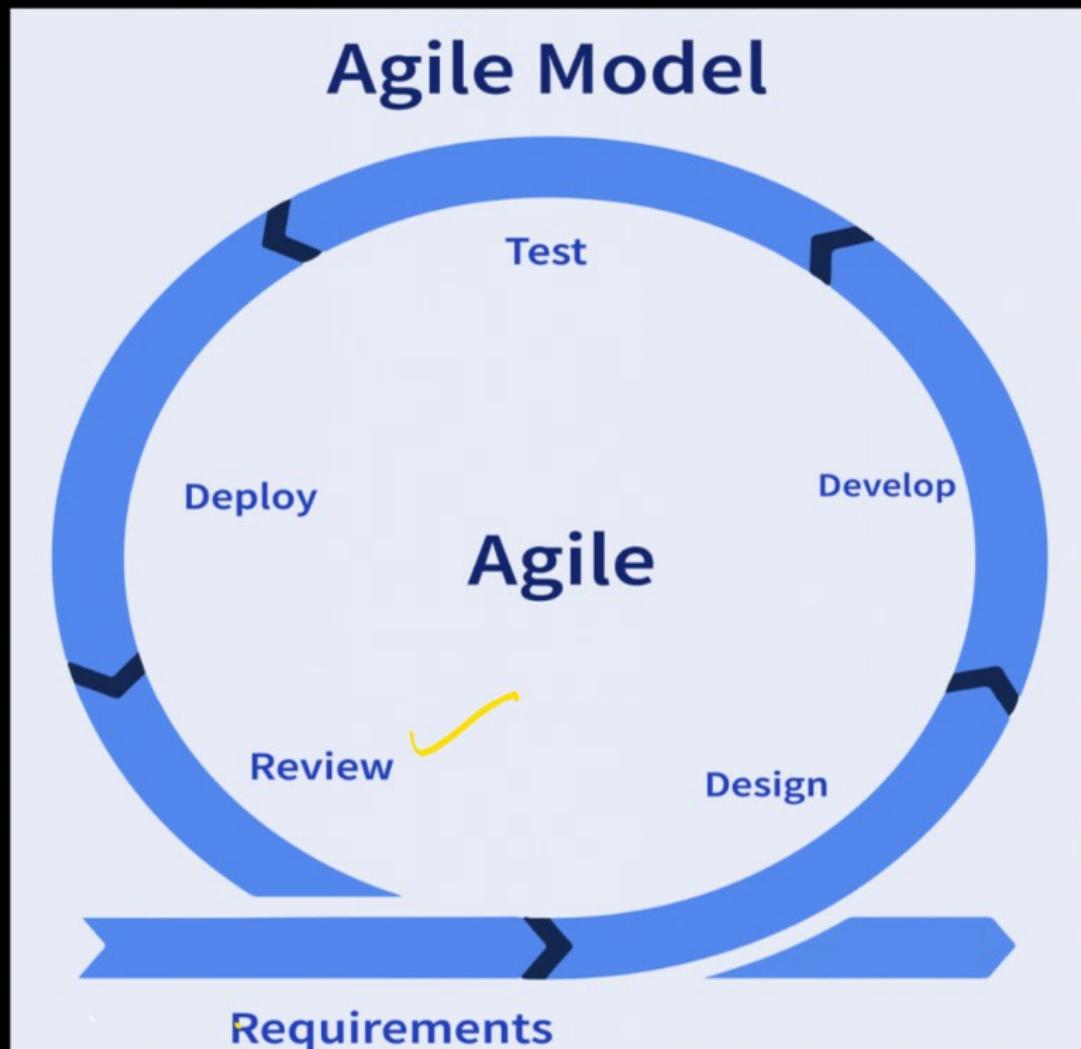


## \* Software Development Models ⇒

- 1. Agile Model
- 2. Waterfall Model
- 3. V Model
- 4. Incremental Model
- 5. RAD Model
- 6. Iterative Model
- 7. Spiral Model
- 8. Prototype Model
- 9. COCOMO Model
- 10. Big Bang Model
- 11. Kanban Model

## 1. Agile Model

- Generative + Incremental का Combo.
- Process Adaptibility & Customer Satisfaction पर Main focus.
- Agile model की इसलिए बनाया गया था, जोकि Software development के बीच स्वेच्छा changes कर सके और Project जल्दी पूरा हो।



- इसमें Software project को  $4 \times 2^0$  incremental parts में divide कर दिया जाता है, और छोटे से छोटे increment part को develop किया जाता है।
- इसलिए इसे Spiral Model की कहते हैं।
- इसमें iterations/modules को छोटा रखा जाता है, जिसके लिए easily manage की सकते।

#### \* Principle of Agile Model →

- Software development के दौरान Customer के साथ Contact बनाकर एवं उनके लिए Customer Requirements को समझने के लिए एक Customer Representative होता है, जो प्रत्येक iteration के बाद उसे Review करता है और Requirements को evaluate करता है।
- इसमें केवल documentation पर depend नहीं रहा जाता है। Customer को working software का demo भी दिया जाता है।

- इस Model में development team का size छोटा (5-7 लोग) रखा जाता है, ताकि सभी face to face communicate कर सकते हैं
- इसमें शहरी focus किया जाता है, कि जब भी कोई change करना हो, तो उसे जल्दी पुरा करना चाहिए।
- इस Model के development में दो Programmers काम करते हैं, नो उनमें से एक coding करता है तभी दुसरा code को Review करता है। इस प्रकार दोनों भवनी जटिल को change नी करते रहते हैं।

## \* Advantages of Agile =

1. इसमें 2 Programmers एक साथ कार्य करते हैं, जिससे Errors बहुत कम होती है।
2. इसमें Software Project में बहुत कम राम्र में Complete कर लिया जाता है।
3. Works on Real approach.
4. Teamwork
5. इसमें Customer Representative को सभी iterations का idea होता है, इसलिए वह Requirements को change कर सकता है।
6. No Need of planning.
7. Easy Management
8. यह developers को flexibility देता है।
9. Rules बहुत कम होते हैं, इसलिए documentation की less need.

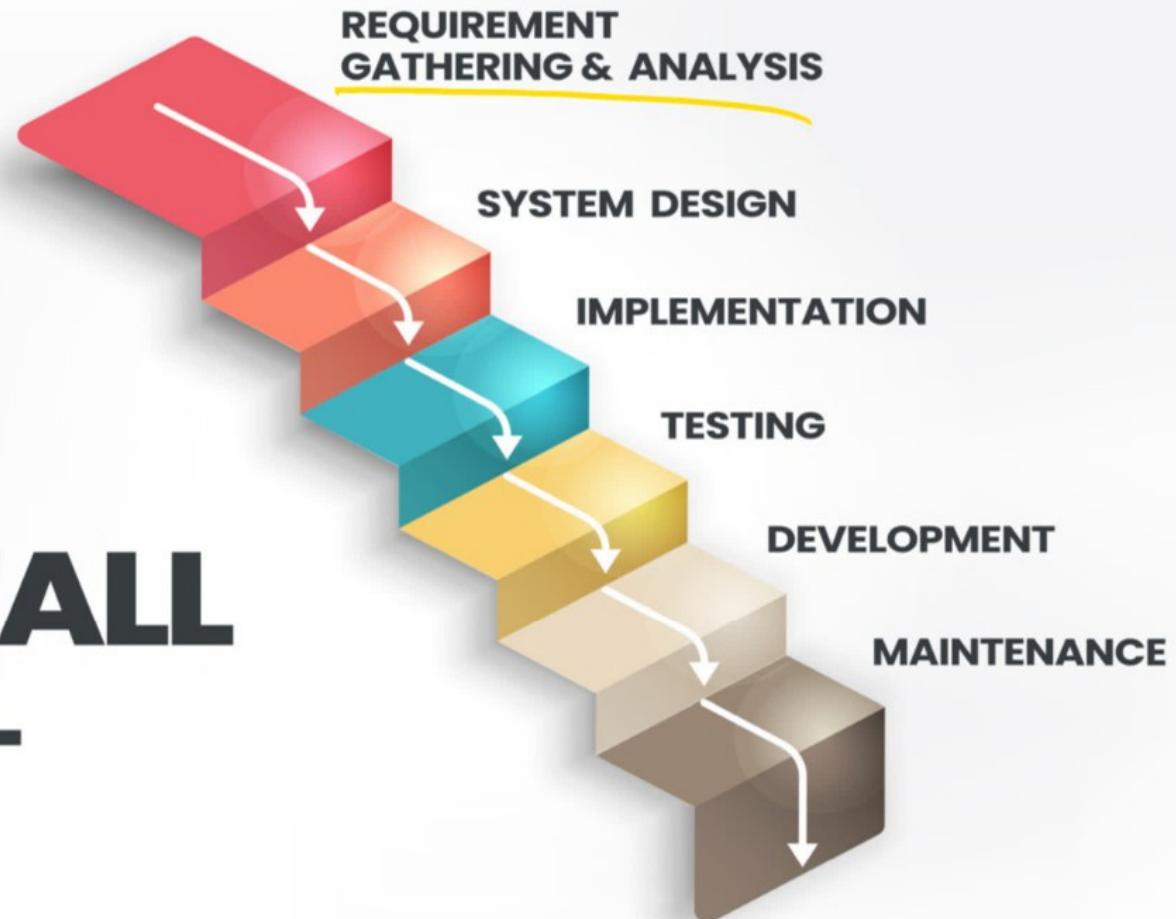
## \* Disadvantages of Agile Model =>

1. complex dependencies को handle नहीं कर पाता है।
2. formal documentation नहीं होने की वजह से कम्पनी-<sup>2</sup> development में confusion.
3. customer representative पर dependency रखाता, अतः कम्पनी-<sup>2</sup> में गलत information दे डेता है, जिससे software गलत<sup>1</sup> का जाता है।
4. केवल experienced programmers की decision सकते हैं।
5. इसमें software development की cost & time starting में पता नहीं चलती है।

## 2. Waterfall Model $\Rightarrow$

- Linear & Sequential Model
- इसमें एक development phase तब तक start नहीं होता है, जब तक Previous Phase end नहीं हो जाता है।
- Phases को overlap नहीं कर सकते हैं।
- जब एक Phase को Complete करके Next phase में चले जाते हैं, तो फिर Previous Phase में काम स नहीं जा सकते हैं।
- एक Phase का output = next phase का input

# **WATERFALL MODEL**



## \* Advantages of Waterfall Model →

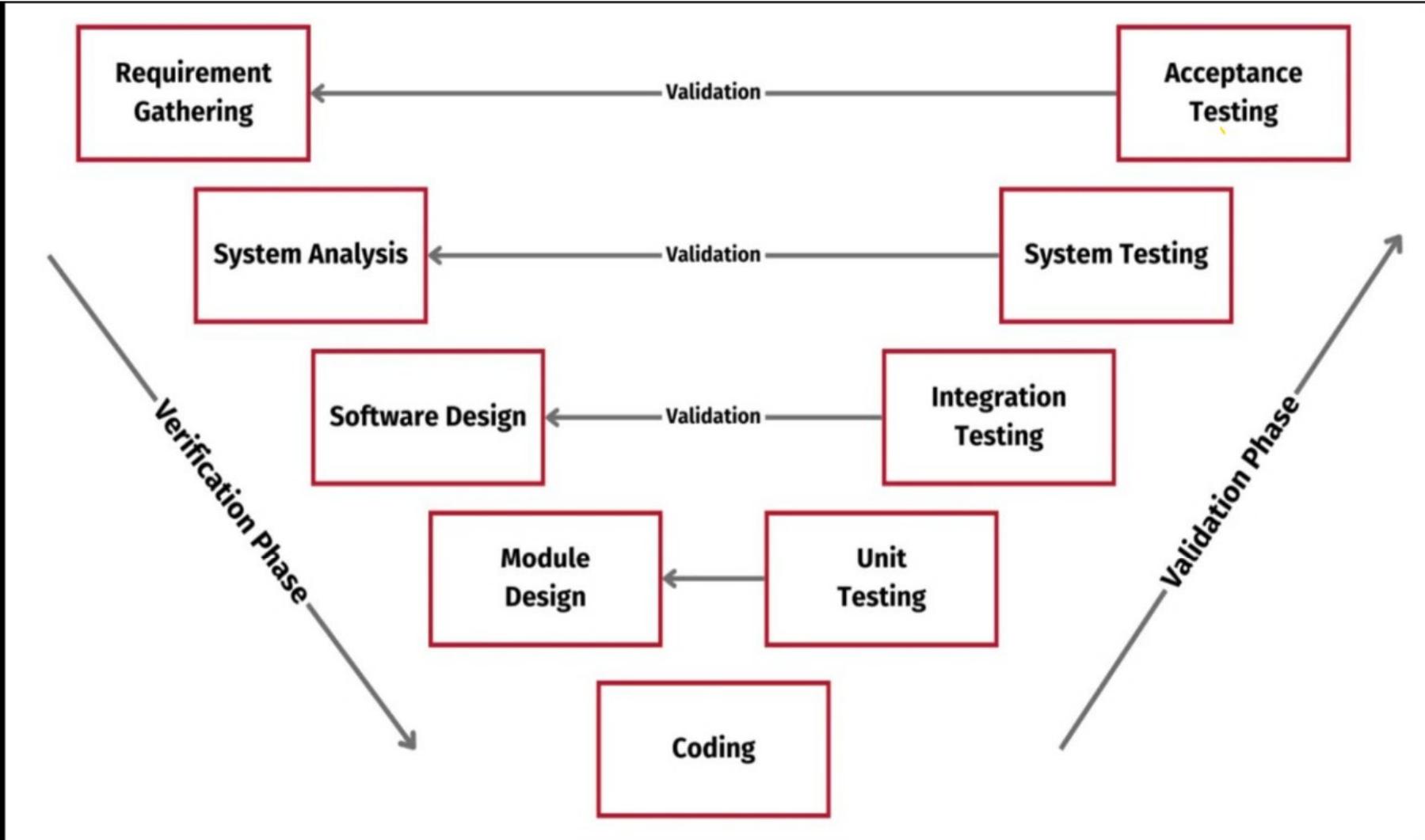
1. Easy to understand.
2. Clear documentation at every step.
3. प्रत्येक step की Timeline & deliverables fix होते हैं, इसलिए manage करना आसान होता है।
4. यह sequential project होता है, इसलिए एक ही direction में progress करता है, इसे track करना easy होता है।
5. यहाँ use small projects में कम Requirements वाले projects में किया जाता है, इसलिए complexities बहुत कम होती है।

## \* Disadvantages of Waterfall Model →

1. जब कोई एक step end हो जाये, तो उसके पश्चात needs की changes नहीं कर सकते हैं, इसी Lack of flexibility कहा जाता है।
2. Testing process, development के बाद होती है, इसलिए multiple errors को initially catch नहीं किया जा सकता है।
3. user involvement कम।
4. Risk management नहीं।

### 3. V Model

- Verification & Validation Model
- Extension of waterfall Model
- इसकी shape 'V' pattern में होती है, अर्थात् इसमें Development & Testing दोनों Parallel चलते हैं।



## \* Structure of V Model ⇒

- A. Verification Phase
- B. Validation Phase

### A. Verification Phase ⇒

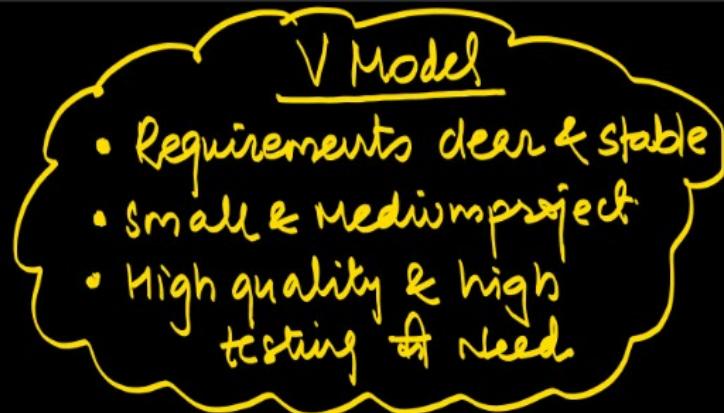
- Requirement Gathering ⇒ Customer की Needs की समझता |
- System Analysis ⇒ Requirements की Analyze कर Architecture को plan किया जाता है।
- Software Design ⇒ Software के Important modules की design को तैयार करना |
- Module Design ⇒ प्रत्येक Modules की detailed design को तैयार करना |
- Coding ⇒ Design के according Software code लिखा जाता है।

## B. Validation Phase ⇒

- Unit Testing ⇒ Code के प्रत्येक छोटे module को individually test करना।
- Integration Testing ⇒ इनी modules को एक साथ मिलाकर Test करना।
- System Testing ⇒ पूरे system की working को Test करना।
- Acceptance Testing ⇒ customer के द्वारा Test किया जाता है, कि मह Software उसकी Requirements को fulfill कर रहा है या नहीं।

## \* Features of V Model → (Advantages)

1. ER step के लिए dedicated testing step होता है।
2. Debugging easy.
3. Documentation पर ज्यादा जोर
4. Easy to understand.
5. प्रत्येक phase में Testing के कारण इसकी quality & reliability better होती है।



## \* Disadvantages of V Model ⇒

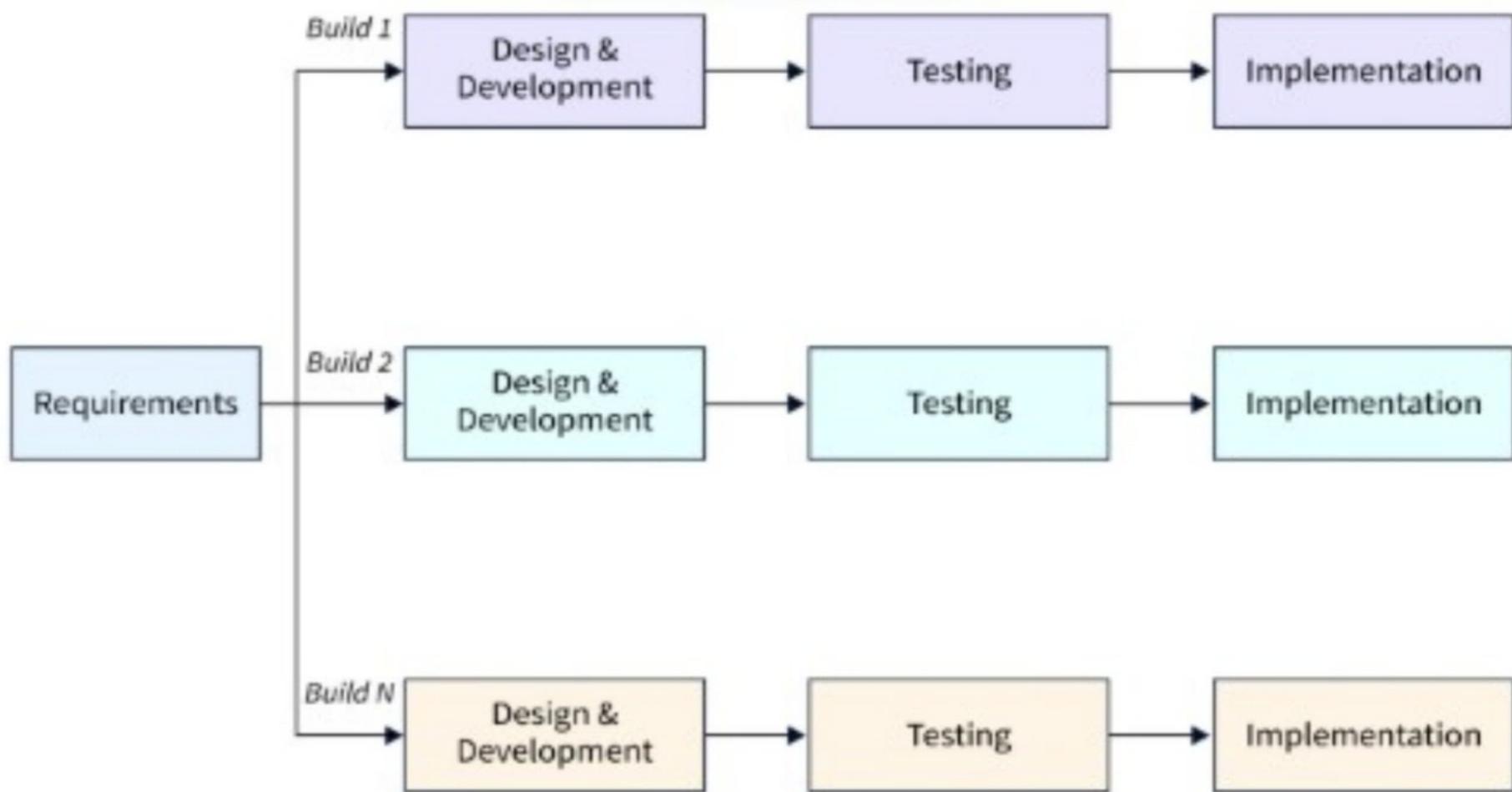
1. Flexible नहीं।
2. Large & complex projects के लिए useful नहीं।

#### 4. Incremental Model ⇒

- पुरा System एक बार में नहीं बनाकर अलग 2 increments में develop किया जाता है।
- इसमें हर increment में कोई part को पूर्ण रूप से बने software में जोड़ किया जाता है।

#### \* Features of Incremental Model ⇒

1. प्रत्येक step में special feature & functionality को develop किया जाता है।
2. प्रत्येक increment के बाद client से feedback लिया जा सकता है।
3. Risk को जल्दी identify किया जा सकता है।
4. कुछ ही increments के बाद Client को early working software display किया जा सकता है।



## \* Advantages →

1. Working software जबकी रैम्पर हो जाता है।
2. Client की Needs के अनुसार change करना easy.
3. Less Risk.
4. Testing Easy.

## \* Disadvantages →

1. क्षयिक design & planning
2. कमी-2 integration में complexities भी जाती हैं।

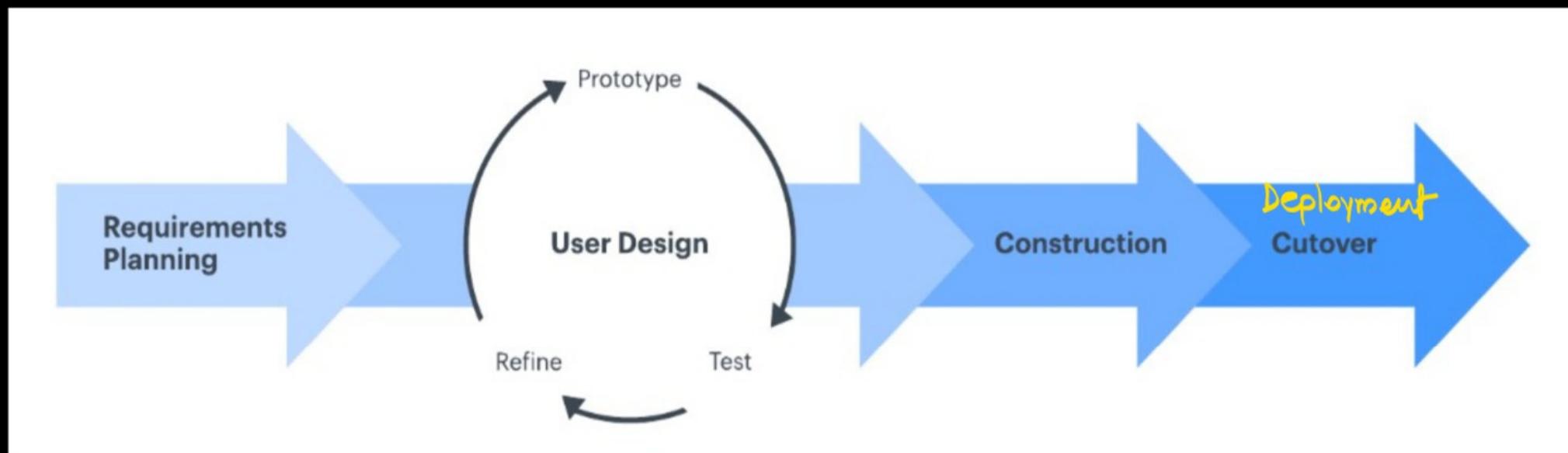
## 5. RAD Model ⇒

- Rapid Application Development
- कम time में Quality software बनाते हैं use.
- Based on prototype.
- इसका Main Objective users की Requirements को अल्दी-<sup>2</sup> समझना व नेट्री हो सफ्टवेर develop करना होता है।

## \* Features of RAD Model =

1. कम समय में Software Development.
2. User involvement.
3. User का feedback जरूरी।
4. प्रत्येक Step पर Prototype बनाये जाते हैं।
5. Iterative process होते के कारण बार-<sup>2</sup> Corrections कर Final product ready किया जाता है।
6. अन्य Models की तुलना में कम Documentation Requirement.

9610991981  
Book



## \* Phases of RAD Model →

### 1. Requirement Planning Phase →

- Customer से Meeting कर Requirement collect करता |
- Customer & Developer दोनों निलंबन Software के scope को understand करते हैं।

### 2. User Design Phase →

- Prototype बनाते जाते हैं।
- एक बार Prototype बनाकर customer से feedback लेकर Design की सही किया जाता है।
- Interactive Phase

### 3. Construction Phase ⇒

- Main software development phase.
- Design phase से प्राप्त information के आधार पर Developers Coding Start करते हैं।
- Tools & Automation का use कर Rapidly development किया जाता है।

### 4. Cutover/Deployment Phase ⇒

- Software को user end पर deploy करता।
- Final Testing, Training & Maintenance इसी phase में होते हैं।

## \* Advantages of RAD

1. Development में कम Time लगता है।
2. User की Requirements को easily understand & apply किया जाता है।
3. High quality software जास्ती बन जाता है।
4. बार-2 feedbacks से software क्षमता बढ़ता है।

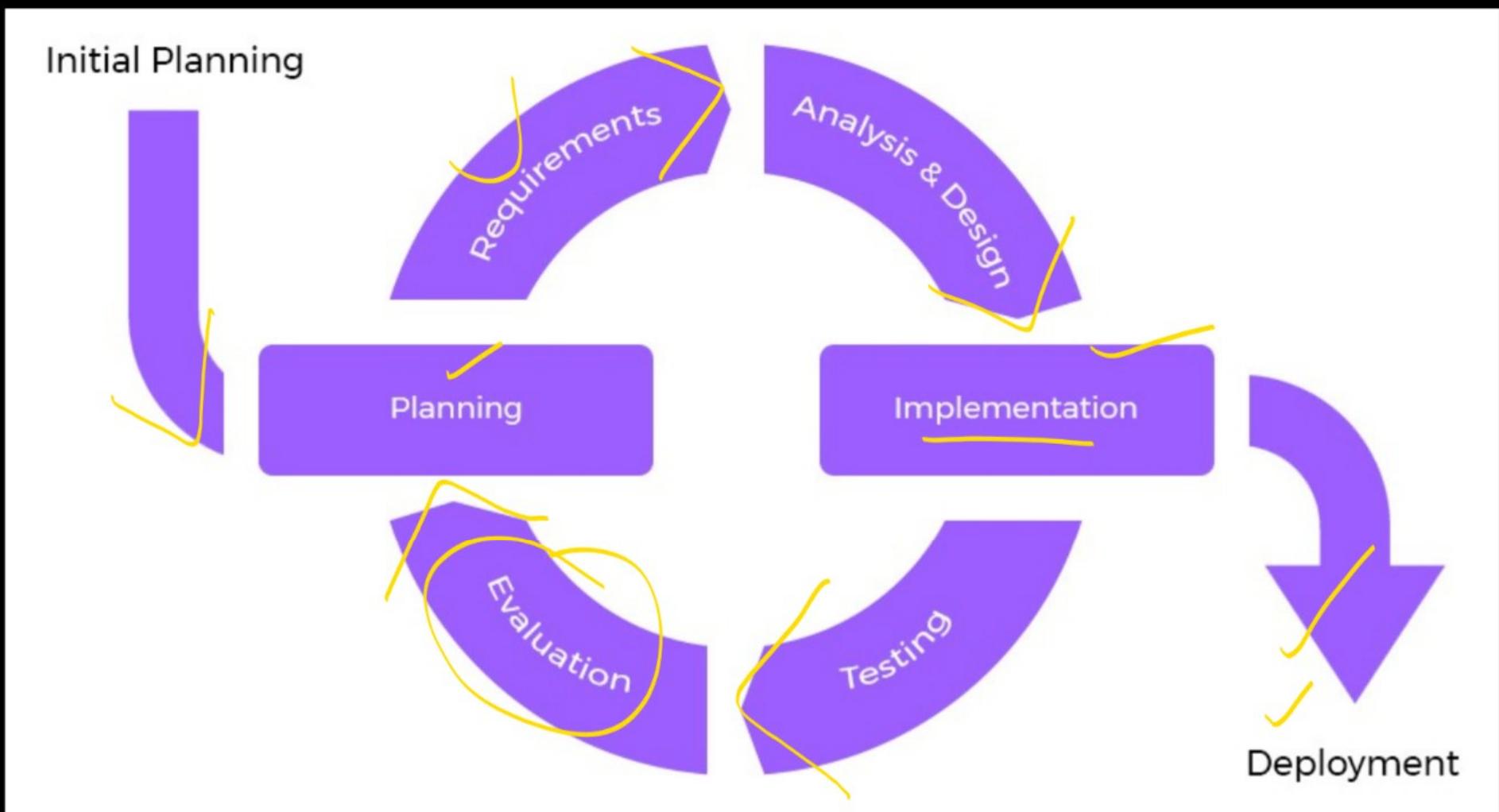
implement

## \* Disadvantages of RAD

1. Only small & Medium sized projects के लिए ही useful.
2. कम experience कोले developers के लिए मुश्किल।

## 6. Iterative Model ⇒

- Software को छोटे-<sup>2</sup> parts में develop किया जाता है।
- प्रत्येक step पर Testing & Review किया जाता है।
- इसमें Software को एक बार में पुरा नहीं बनाया जाता है, जबकि प्रत्येक iteration में छोटे-<sup>2</sup> करके develop किया जाता है।



## \* Characteristics of Iterative Model =>

### 1. Phased development =>

→ प्रत्येक iteration में system का एक part बनाकर Test कर लिया जाता है।

### 2. Customer Feedback =>

→ प्रत्येक iteration में developed part पर customer का feedback लिया जाता है, ताकि next iteration में उसे follow किया जाता है।

### 3. Reduced Risk =>

→ Risks को initial phases में ही पकड़ लिया जाता है।

## \* Phases of Iterative Model =>

### 1. Requirement Gathering =>

- Customer की Basic Requirements collect करता |
- पुढ़ी Requirements एक साथ नहीं लेकर parts में ली जाती हैं।

### 2. Design =>

- Requirements के Base पर design रेस्मार की जाती है।
- प्रत्येक iteration के बाद design change होती है।

### 3. Implementation =>

- Design के आधार पर Coding.
- छोटे-से parts में execution.

#### 4. Testing ⇒

- प्रत्येक Iteration के बाद Testing.
- Bugs & errors की संख्या किसे जाता है।

#### 5. Evaluation ⇒

- प्रत्येक Iteration के बाद Prototype को customer द्वारा evaluate किया जाता है।
- Feedback के आधार पर necessary changes किये जाते हैं।

### \* Advantages of Iterative Model ⇒

- Customer needs से कठीं से समझा जाता है।
- जल्दी working software मिल जाता है।
- Errors को जल्दी ठीक किया जाता है।
- bड़. Software को छोटे-2 parts में develop करना easy होता है।

### \* Disadvantages of Iterative Model ⇒

- प्रधिक iterations के कारण Time & cost increase हो जाता है।
- Requirements clear नहीं होने से system complex हो जाता है।

## 7. Spiral Model ⇒

- Given by = Barry Boehm (1986)
- Mainly यह model Risks को कम करते हैं और project की cost & Time को Manage करने में इसका use किया जाता है।
- Regular iterations के कारण इसे Spiral कहा जाता है।

## \* Main Elements of Spiral Model ⇒

### 1. Four Major Activities ⇒

#### A. Determine Objectives & Alternatives ⇒

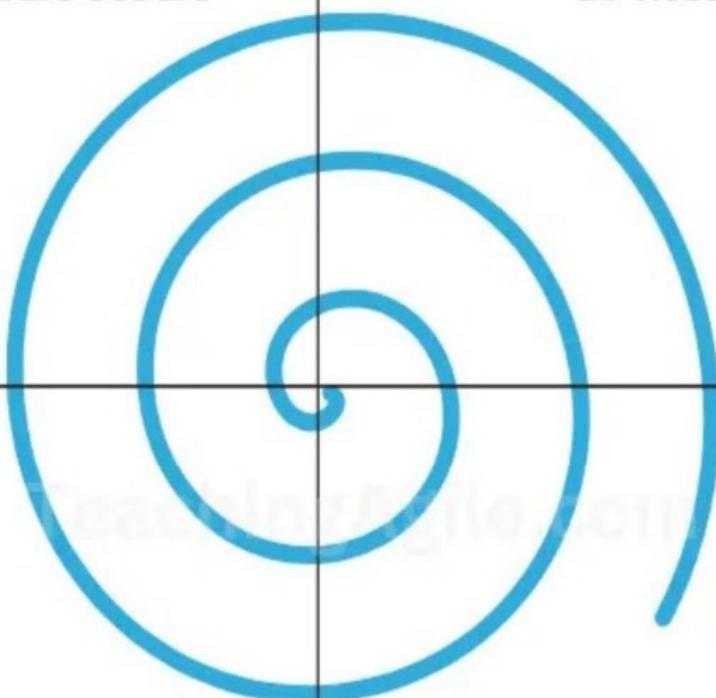
- इस step में project के objective को define किया जाता है और Available Alternatives की Evaluate किया जाता है।
- यह Risk Analysis & objectives के बीच priority define की जाती है।

**1. IDENTIFY OBJECTIVES**

**2. RISK ANALYSIS**

**4. EVALUATION**

**3. PRODUCT DEVELOPMENT**



### B. Risk Analysis ⇒

→ प्रत्येक step के last में Risks को Analyze किया जाता है तथा possible problems की determine किया जाता है और उनके लिए रणनीतियाँ (Plans) तयार की जाती हैं।

### C. Development & Testing ⇒

→ इस step में software को Design, Coding & Testing किया जाता है।

### D. Evaluation & Planning ⇒

→ Project के execution को evaluate किया जाता है।

→ इसमें project की progress & Plans की समीक्षा की जाती है।

→ इसी के आधार पर next plan बनाया जाता है।

## 2. Cycle of Spiral Model

- इस Model की Cyclic pattern में display किया जाता है, जिसमें प्रत्येक Cycle के बाद project development के क्रमे steps बनते हैं।
- प्रत्येक Cycle में चारी Activities की Repeat किया जाता है।
- प्रत्येक Cycle के बाद Software, more mature & more complete होता है।
- अद्य Cycles तक तक चलते हैं, जब तक Software पुराणा Tested नहीं हो जाता है।
- Waterfall + Iterative का Combo.

### 3. Risk Management ⇒

- Risk identify & Analysis इस Model का important element है।
- इसमें प्रत्येक Step में आने वाली Risks को identify कर उनके Solution की planning कर ली जाती है।

### 4. Interactive & Iterative Development ⇒

- यह iterative Model है, जिसमें बार-<sup>2</sup> Repeat किया जाता है।
- प्रत्येक Step के बाद Software का नया version मिलता है।
- यह Prototype development को छापा है, जिससे जल्दी-<sup>2</sup> feedbacks लेकर Software को बेहतर किया जा सकता है।

## 5. Suitability ⇒

- Large & complex projects के लिए suitable तरीका Risk Management is big factor.
- इसका use वहाँ किया जाता है, जहाँ consumer की Requirements clear रही होती है।

## \*Advantages ⇒

1. Better Risk Management.
2. Software steps में Ready होता है, इसलिए More mature, Reliable & Accurate होता है।
3. Regular customer feedback.
4. Flexibility (New Requirements can be added.)

## \* Disadvantages =>

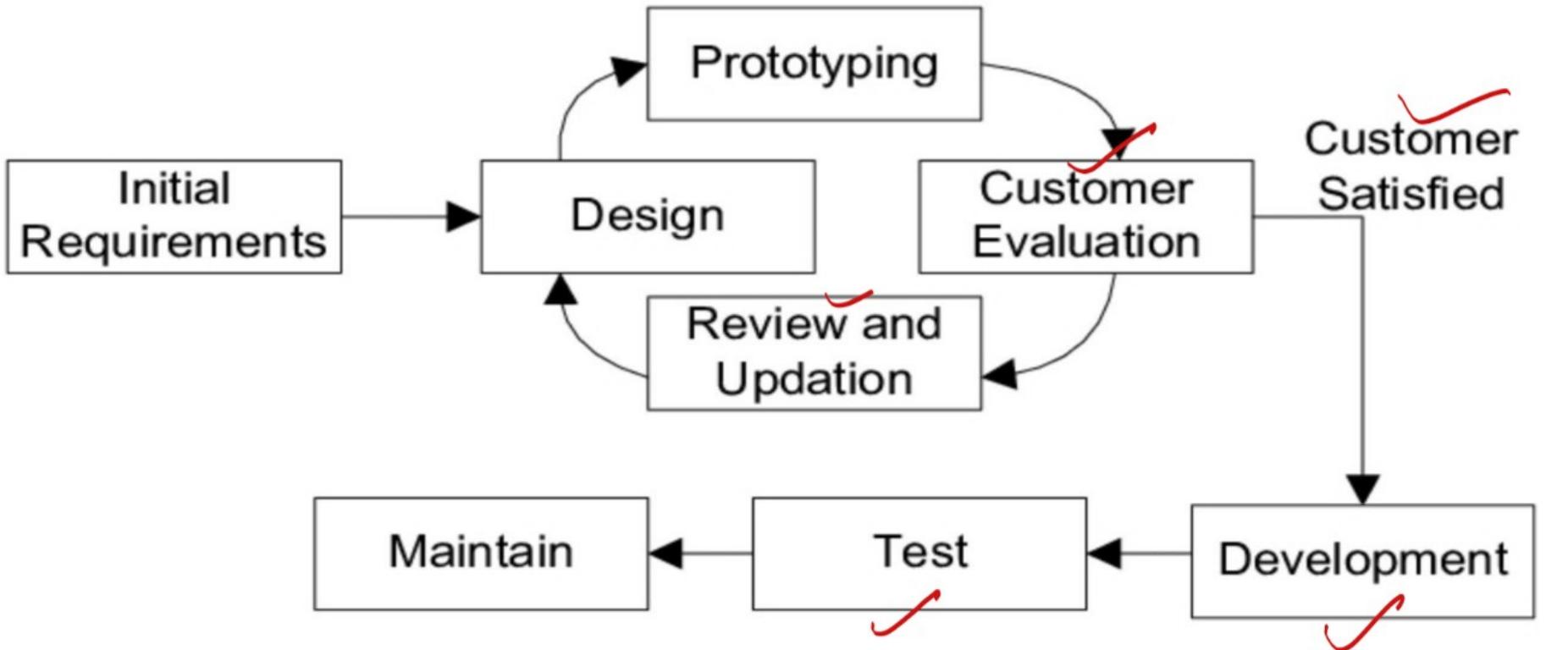
1. More Time, Cost & Resources Required.
2. Not suitable for small projects.
3. High level of expertise required

## B. Prototype Model

- Customer की Requirements clear रही होने पर USE.
- इसमें एक working / dummy model बनाया जाता है, ताकि customer इसकी working को समझ सके और necessary changes कर सके।
- customer की Requirements को बीटार तरीके से समझने में USE.

## \*Process Steps

1. Requirement Gathering
2. Prepare quick design
3. Build working prototype
4. User Evaluation & feedback
5. Refinement & improvements in prototype
6. Final Product deployment



PROTO TYPE MODEL

## \* Advantages of prototype Model ⇒

1. Customer satisfaction अधिक |
2. Requirements को clearly समझा जा सकता है।
3. Time की बचत।

## \* Disadvantages of Prototype Model ⇒

1. Large & complex project के लिए उपयोगी नहीं।
2. Customer dummy prototype को final product समझ नहीं सकते।

## 9. COCOMO Model

- Constructive Cost Model
- Developed by = Barry Boehm (1981)
- इसका प्रयोग Software develop में लगने वाले Efforts, Cost & Time का अनुमान लगाने में किया जाता है।

## \* Types of COCOMO Model

### A. Basic COCOMO Model

- सबसे सरल Model
- ये केवल Software की size ( KLOC = Thousands of Lines of Code ) पर Based होते हैं।
- तीन Categories :-
  - (i) Organic = Small & Easy product ( Small & experienced Team )
  - (ii) Semi detached = Moderate Model ( Average experienced Team )
  - (iii) Embedded = Complex & Hard Model ( जटिल - Military software )

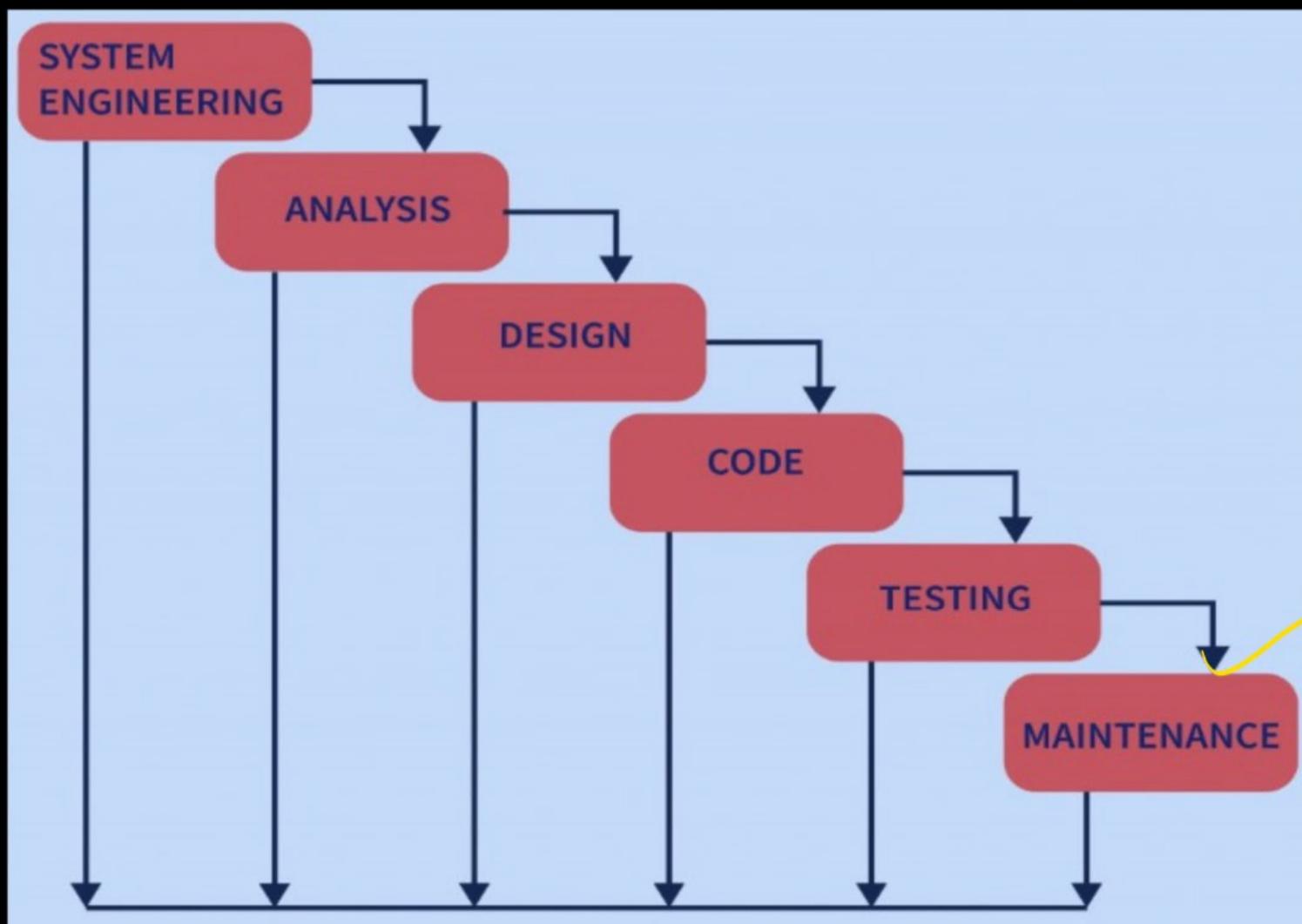
### B. Intermediate COCOMO Model $\Rightarrow$

$\rightarrow$  Same as basic model but इसमें 15 costdrivers add किये जाते हैं (Team - Experience, Team की Capacity & Tools)

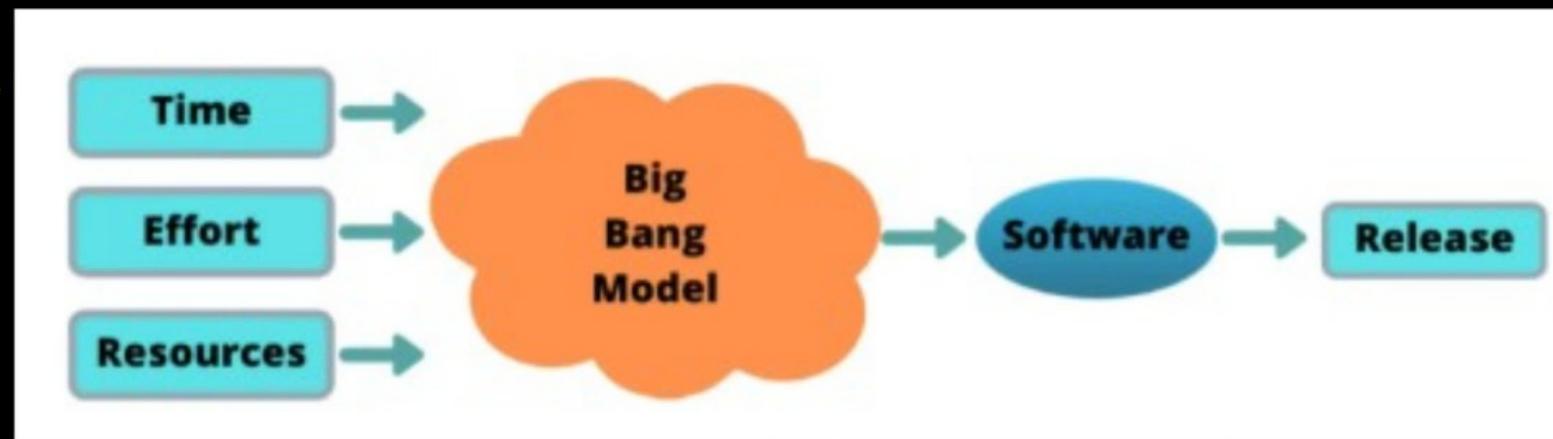
### C. Detailed COCOMO Model $\Rightarrow$

$\rightarrow$  Intermediate Model का expansion.

$\rightarrow$  प्रत्येक Software Module के लिए Cost-driver add किया जाता है।



## 10. Big Bang Model ⇒



- इस Model में बिना पुरी Planning के व Customer की पुरी Requirements को समझे बिना Coding Start कर दी जाती है।
- No formal Planning
- Minimum Requirement
- पुरा System एक दी बार में develop कर बिना Testing के deliver कर दिया जाता है।

## \* Advantages of Big Bang $\Rightarrow$

1. Planning में Time नहीं लगता है।
2. नेत्री से व्युहान की जा सकती है।
3. Small Models के लिए ठीक।

## \* Disadvantages of Big Bang $\Rightarrow$

1. High Risk
2. Documentation नहीं
3. Changes करना Tough
4. Large & Complex projects में fail.

## 11. Kanban Model ⇒

- Visual Project Management Tool.
- Workflow को manage करने वाला प्रै.
- Japan में Toyota Company ने start.
- कोई project का start हुआ, चल रहा है, कब खत्म होगा आदि को Visual Signals द्वारा display किया जाता है।



## \* Size Estimation in Software project =>

- किसी Software Project की Complexity, Requirement & Workload का अनुमान लगाता,  
वाकि Time, Cost & Resources की planning की जा सके।
- Project Planning में Help करता।
- Accurate Time & Cost का पता लगाता।
- Required Resources को Identify करता।
- Possible Risks को Identify करता।
- Client के लाभ सही delivery timeline set करता।

## \* Size Measurement Techniques →

### 1. LOC (Lines of Code) ⇒

- Total किसी गड़ी Lines count करता है।
- शब्दों पुरानी & Easy method.
- अगे किसी project में 1000 lines हैं, तो इसे 1KLOC कहते हैं।

### 2. FPA (Function Point Analysis) ⇒

- ये प्रृथम हारा किसी software से expected tasks को मापता है।
- यह Input, Output, User Interface & files को लाभार्थी पर depend करता है।
- किसी Technology पर based नहीं है।

### 3. UCP (Use Case Points) $\Rightarrow$

- Use Case diagrams पर based.
- इसमें Use case की Complexity को उपात ने रखकर Size तम की जाती है।
- Precise Accurate output देता है।

## \* Size Estimation Techniques $\Rightarrow$

A. Expert Judgement  $\Rightarrow$  Expert लोगों को Experience के आधार पर अनुमान लगाना ।

B. Delphi Technique  $\Rightarrow$  कई व्यक्ति<sup>2</sup> Experts से व्यक्ति<sup>2</sup> अनुमान लेकर उल्टा Average निकालना ।

C. Analogy Based Estimation  $\Rightarrow$  Last completed same project के आधार पर अनुमान लगाना ।

D. Top-Down Estimation  $\Rightarrow$  किसी System का अनुमान लगाकर उसे छोटे-2 parts में divide करना ।

E. Bottom-Top Estimation  $\Rightarrow$  किसी System के छोटे-2 parts का अनुमान लगाकर पुरे System का अनुमान लगाना ।

## \* Cost Estimation in Software project $\Rightarrow$

- Project का Budget final कर, Customer से उचित राम लेना।
- Resources का सही Allocation.
- Time & workload का Management.

## \* Types of Cost Estimation $\Rightarrow$

### A. ROM (Rough Order of Magnitude) $\Rightarrow$

- Initial अनुमान।
- इसमें -25 से +50% तक Error हो सकती है।

### B. Budget Estimation $\Rightarrow$

- Planning में उपयोगी।
- Estimated Cost की Accuracy -10 से +20% तक हो जाती है।

### C. Definitive Estimate $\Rightarrow$

→ यदि Project की सारी जानकारी मिल जाती है, तो Definitive Estimate मिल जाती है, जो अंतर -5% से +5% तक Accurate होती है।

### \* Factors Affecting Cost =

- Size of Project
- Complexity of Project
- Team Experience
- Technology used
- Onsite & offShore (Place of work)
- Deadline

## \* Cost Estimation Techniques $\Rightarrow$

### 1. Algorithmic / Parametric Methods $\Rightarrow$

#### A. COCOMO Model $\Rightarrow$

→ Based on LOC (Lines of Code)

→ 3 Types  $\Rightarrow$  Basic, Intermediate & Detailed.

#### B. FPA (Function Point Analysis) $\Rightarrow$

→ Based on number of functional Units (Input, Output, files, interfaces)

### 2. Empirical Methods $\Rightarrow$

#### A. Expert Judgement $\Rightarrow$

→ Expert लोगों की याद लेकर Cost का पता लगाना।

#### B. Delphi Technique $\Rightarrow$

→ बहुत सारे लोडों से idea लेकर उसका Average निकाल लिया जाता है।

### 3. Analogous Estimate $\Rightarrow$

$\rightarrow$  Previously किये गये same project के Data से Estimation किया जाता है।

### 4. Top-Bottom Estimation $\Rightarrow$

$\rightarrow$  पुरे project की complete cost का idea लगाकर उसे छोटे-2 parts में divide कर दिया जाता है।

### 5. Bottom-Top Estimation $\Rightarrow$

$\rightarrow$  किसी project के छोटे-2 parts की cost का idea लगाकर पुरे project की cost का पता किया जाता है।

## \* Staffing Level Estimation ⇒

→ किसी project को पुरा करने में कितने लोग चाहिए वे किस समय पर कितने लोग कार्य करेंगे।

## \* Staffing level को फ़ादूयातिक करना ⇒

### A. WBS (Work Breakdown Structure) बनाना ⇒

→ Project को छोटे-2 parts में divide करना।

→ प्रत्येक Part का Time & Staff को फ़िx करना।

### B. Effort Estimation ⇒

→ किसी भी कार्य को complete करने में कितने hours लगेंगे।

→ जैसे-किसी 40 hours के कार्य को एक week में पुरा करने के लिए 2 manpower लगा दिया है। (One man = 20 hours/week)

## \* Tools ⇒

1. COCOMO Model
2. MS Project
3. Jira / Trello
4. PERT & CPM Charts.

↓      ↗ Critical Path Method

Program Evaluation  
& Review Technique

## \* DFD →

→ Data Flow Diagram

→ एक प्रकार का Graphical Tool जो Data के Flow को बताता है।

→ इसमें 4 things को clear किया जाता है-

A. Data आता कहाँ से है।

B. Data, process कैसे होता है।

C. Data, store कहाँ होता है।

D. Data उत्तर कहाँ है।

→ Also known as Bubble chart.

## \* DFD Components ↗

### 1. External Entity ↗

- ऐसे लोग या organizations जो System के साथ interact करते हैं।
- ये System से Data लेते / देते हैं।
- Rectangle (  ) से Represent करते हैं।
- जैसे - Bank, Customer etc.

## 2. Process →

- System में Data को process करने का कार्य।
- इनपुट को आउटपुट में बदला जाता है।
- इसे वीर्यवृत्त (Ellipse) से display किया जाता है।
- उसे - किसी Order को process करना, Data का Validation करना।

## 3. Data Store →

- वह स्थान जहाँ पर Data को Temporarily या permanent रूप से store किया जाता है।
- उसे - File, DataBase.
- इसे open ended Rectangle (==) से display किया जाता है।

#### 4. Data Flow ⇒

- System के अन्दर Data कैसे घोर रहा हो कहा जाता है, को denote करते हें USE
- Arrow का USE किया जाता है
- Data की Direction, Arrow की Direction से denote की जाती है।

## \* Types of DFD =>

### 1. Level 0 / Context level DFD =>

- Highest level View.
- इसमें पुरे System को एक ही process के रूप में denote किया जाता है।
- External entities & Data flow को ही denote किया जाता है।

### 2. Level 1 DFD =>

- Processes को Sub-Processes में divide कर दिया जाता है।
- इसमें यह display किया जाता है, कि Data किसे subprocesses में flow करता है।

### 3. Level 2, 3 (Detailed DFD) ⇒

- Level 1 को More details के साथ Explain किया जाता है।
- इसका Use अधिक depth में details को जानने में किया जाता है।

### \* Rules of DFD ⇒

- कम से कम एक Input & Output होता चाहिए।
- Data store से Data direct other data store में नहीं डा सकता है।
- Data flow external entities के बीच नहीं होता है।
- एक process और process के बाप्त directly communicate नहीं कर सकती है।
- जब Data process से रहा होता है, तो उसका flow दो directions में नहीं डिव्हिल सकते हैं।

## \* Object Oriented Design →

- इस design में software की objects & classes के रूप में design किया जाता है।
- OOPS concepts पर based.

## \* Characteristics of Object Oriented Model →

### 1. Class →

- एक प्रकार का template जिसका use objects बनाने में किया जाता है।
- इसमें Attributes & methods रखे रखते हैं।

### 2. Object →

- class का instance.
- class की सभी properties & methods को use करता है।

### 3. Encapsulation ⇒

- Data & उससे related tasks/functions को एक ही class/unit में Bind करना |
- Data को external involvement से Secure रखता है।

### 4. Inheritance ⇒

- Reusability of code
- एक class की properties को दूसरी class में use करना |

### 5. Polymorphism ⇒

- एक ही नाम के function को अलग-2 तरीकों में use में लेना।
- 2 प्रकार-
  - A. CompileTime (Overloading)
  - B. RunTime (Overriding)

6. Abstraction ⇒ System की Complex information को Hide करे Required Information को ही Reveal करना |

\* Principle of Object Oriented Design →

\* SOLID Principles →

A. S = SRP →

→ Single Responsibility Principle

→ एक class = केवल एक कार्य |

B. O = Open/Closed Principle →

→ Software entities Extension के लिए Open तथा Modification के लिए Close होनी है।

C. L = Liskov Substitution Principle →

→ Subclasses को Superclass की जगह use में लिया जा सकता है।

#### D. I - Interface Segregation Principle →

→ किसी Client को हमें interface पर depend नहीं होना चाहिए, जिसका वह use नहीं करता हो।

#### E. D - Dependency Inversion Principle →

→ Client को केवल उसके Abstraction पर ही depend होना चाहिए, ना कि Complete Implementation पर।

## ★ SRQA →

- Software Reliability & Quality Assurance
- Software की ऐसी विशेषता, जो महत्वाती है कि वह User की Requirements की पुरा करता है या नहीं, उसे Software Quality कहते हैं
- विशेषताएँ ⇒
  - A - Functionality
  - B - Reliability
  - C - Usability
  - D - Efficiency
  - E - Portability
  - F - Maintainability
  - G - Correctness

- Quality Assurance एक प्रकार की process है, जो महसूसिष्ट करती है कि Software product, predefined Standards को follow करता है या नहीं।
- QA, Development Process के दौरान apply होता है।
- किसी भी software के दृश्य विधीरित Timeline तक बिना किसी error के गार्मी करता Software Reliability कहलाता है।
- MTBF (Mean Time Between Failure) उस Average Time को denote करता है, जब तक Software बिना fail हुए कार्य करता है।

- Quality Control एक Technical Process है, जो Software की Test कर उसकी Quality को नियंत्रित करती है।
- Software के Quality Standards के रूप में ISO 9001, CMMI (Capability Maturity Model Integration) - Process Correction के लिए, Six-Sigma - Errors / Risks की कम करने में USE, आदि Standards का USE किया जाता है।

## \* Musa's Basic Model =>

- यह Instructional Technology का Model होता है, जिसे Musa ने बनाया था।
- इसका प्रयोग Education के field में Effective communication & Teaching-Learning process को बेहतर बनाने में किया जाता है।
- 4 Components =>  
Source → Message → Channel → Receiver
- Purpose =>
  1. Communication को Effective बनाना।
  2. Teaching Technique को Easy & Simple करना।
  3. Teachers को यह समझाने में help करता है कि Students नके Information कैसे पहुँचनी है।

## \* ISO 9000 Series ⇒

- International Organization for Standardization
- ISO 9000 QMS (Quality Management Systems) के लिए Standards & directions provide करता है।
- Main Characteristics ⇒
  - A. Customer focus
  - B. Leadership
  - C. Engagement of People
  - D. Process Approach
  - E. Continuous Improvement
  - F. Relationship Management
  - G. Evidence based decision making

\* CMM  $\Rightarrow$

- $\rightarrow$  SEI CMM (Software Engineering Institute Capability Maturity Model)
- $\rightarrow$  इसका USE Software Organizations द्वारा Process Maturity का पते में किया जाता है।
- $\rightarrow$  यह कहाता है कि कोई organization, Software Development में कितना Mature & Efficient है।

\* Levels of CMM  $\Rightarrow$  ⑤

A. Level 1 (Initial Level)  $\Rightarrow$

- $\rightarrow$  No fix process.
- $\rightarrow$  No fix manner.

### B. Level-2 (Repeatable) ⇒

- Basic project Management processes
- Success को Repeat किया जाता है।

### C. Level-3 (Defined) ⇒

- Process Documentation defined.
- Organization level पर standardization .

### D. Level-4 (Managed) ⇒

- Processes & Quality management.

### E. Level-5 (Optimizing) ⇒

- Regular improvement .

## \* ISO 9000 v/s SEI CMM ⇒

Features	ISO 9000	SEI CMM
1. Type	QMS (Quality Management System)	Process Maturity Model
2. Focus	Quality Assurance & Customer Satisfaction	Software Process Improvement
3. Certification	By ISO	No certifications, only maturity levels are there.
4. Process Approach	Requirement of Process	Maturity of Process

## \* Software Maintenance ⇒

→ किसी Software में Timely corrections & modification को Maintenance कहा जाता है,  
ताकि Software smoothly work कर सके।

## \* Types of Maintenance ⇒

### A. Corrective ⇒

→ Bugs & errors की correct करना।  
→ Ex= Runtime error, logical error.

### B. Adaptive ⇒

→ किसी SW को नये Hardware या OS पर Run करने लायक बनाना।  
→ Ex= किसी old SW को windows 11 पर Run करना।

### C. Perfective $\Rightarrow$

- किसी old SW का नया Version available करनाता, जिसकी working capability बेहतर हो।
- Ex = User Interface को बेहतर बनाना।

### D. Preventive $\Rightarrow$

- Future की problems & errors से security के लिए available करनामा जग्या update.
- Ex = Code को Scalable बनाना।

## \* Software Maintenance Process Model ⇒

### A. Quick Fix Model ⇒

- directly problem solve करने पर centralized.
- Documentation & design पर no focus.
- Temporary solution.
- Fast.
- Debt increase होता है।

### B. Iterative Enhancement Model ⇒

- Current system में slowly - slowly corrections करता |
- प्रत्येक iteration में नये features & corrections available करवाता |
- Documentation & Testing include.

### C. Reuse Orientation Model $\Rightarrow$

- Available code & content को Reuse करता।
- Larger systems अधिक उपयोगी।
- Time & Cost में बचत।

### D. Boehm's Model / SMW (Software Maintenance Wheel) $\Rightarrow$

- Circular Model which includes identification of change, impact analysis, modification, Testing & Release.

## \* Reverse Engineering :-

- It is a process in which we fetch design, structure, documentation of a currently available software without changing its code.
- पुराने SW को समझने & Documentation prepare करने में वर्ता.
- इह तीन steps Information extraction (Source code & DB Schema), Abstraction (Low level से high level design को समझना), Documentation में काग करता है।