06
09
2019

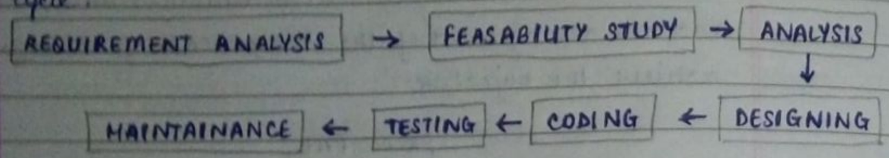
DATE : / /
PAGE NO. :

SDLC (System Development Life Cycle)

* A System Development Life Cycle is the process that is followed in implementing a computer based system or a sub-system.

* SDLC is a framework consisting of a series of various different tasks to achieve the specific centralised objective in the development of a system.

* Cycle :



I) Requirement Analysis :

- The base of this phase is to survey or initially investigate to determine whether an alternative system can solve the problem

II) Feasability study :

- It is the detailed study about the system.
- It is a test of a system proposal according to its workability, impact on the org., ability to meet user needs and effective use of resources.
- Types of feasabilities :

a) TECHNICAL FEASABILITY : survey of softwares & hardwares. It is related to the availability of hardware & software to perform the essential computing.

b) ECONOMIC FEASABILITY : the new system should be economically beneficial for that the cost & benefit analysis is performed.

c) TIME FEASABILITY : the system should be implemented within the mentioned time constraint.

d) LEGAL and ETHICAL FEASABILITY : the new system should be existing within the legal and ethical boundaries.

DATE : / /
PAGE NO. :

III) Analysis :

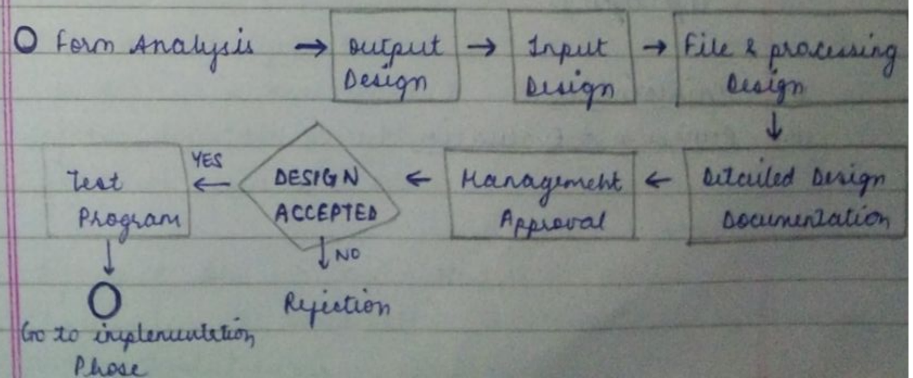
When the feasability study is completed the project team concentrates on the analysis part.
Analysis is a detailed study of the various operations performed by a system & their relationship.
There are also some major objectives :
a) Defines the scope of the new system
b) Understand the old info. of system
c) Review the feasability & cost analysis
d) Develop the structural, functional specification for the new system.

After Analysis phase, this phase generates a document named SRS (Software Requirement Specification)

IV) Designing :

It is done by the project manager.
System design is the determination of the process and data that are required by a system.
It also depends on what are the things that users are expecting.

SYSTEM DESIGN :



- Output design : Procedure to produce, represent Output.
- Input design : Format of Input data, Input data design, sample presentation of the Input data
- File and Processing design : files of database are designed to fulfill the requirement of system. In processing design program construction & testing comes.
- Detailed Design documentation : Documentation of details related to the system. " of estimation which affects the system.
- Mgmt. Approval : All the previous work evolution. Any side effects and risk factors.
- Design Acceptance : checking whether the design is accepted or not.
- Test Program
- Go to Implementation Phase.

v) Coding :
In this phase, development of source code on a specific Technology is compiled.

vi) Testing :
After developing the code of the system, it enters into the testing phase which is called as STLC (system Testing life cycle).

vii) Maintainance :

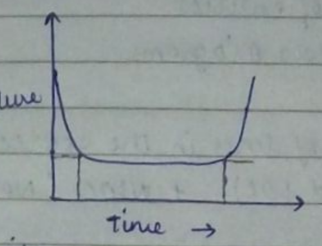
viii) Review and Evaluation :

SOFTWARE

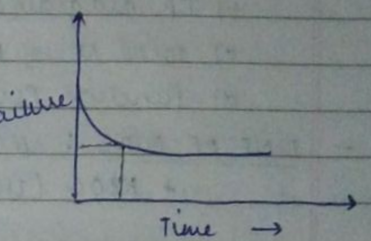
Software is defined as a collection of programs, procedures data and documentations.

* Characteristics of Software Engineering :

1) Software is developed or engineered not manufactured.
ii) Software doesn't wear out with time.



failure curve of Hardware
(Bath tub curve)



failure curve of software.

- In this H/W product, it can be observed that the failure ratio/rate is initially high but it decreases as the components are identified and removed after some time the failure rate is again increased. this gives the plot of H/W realiability over the time is called the BATH TUB CURVE.
- At the other side for S/W failure curve, the failure curve rate is high at the initial level, more errors are identified and after some time after removing these error the curve will come in a stable state.

* Software Life Cycle Method (SLCM) : A SLCModel (Process Model) is a descriptive & diagramatic representation of the software lifecycle. A life cycle represent all the activities required to make a software product.

* Difference B/w life cycle Models are there :

1) Classical ii) Waterfall iii) Iterative iv) Prototype
v) Evolutionary vi) Spiral

DATE : / /
PAGE NO. :

* PROJECT SIZE ESTIMATION TECHNIQUES

1) Estimation of the siz of the software is an essential part of software project mgmt.

- It helps project managers to predict the efforts & time which will be needed to build the project.
- Various measures are used. They are:
  a) Line of Code
  b) ER diagram's No. of entities
  c) Total No. of Data Flow Diagrams
  d) Function Points

a) - <u>LINE OF CODE</u> : total no. of lines in the source code.
+ KLOC (Thousand LOC) + NLOC (Non Comment loc)

* SOFTWARE QUALITY ASSURANCE : (SQA)

Software Quality Assurance is a set of activities for ensuring quality in software Engineering process.
It ensures that developed softwares meets and compile with the defined or standardized quality specifications ;
Software Quality Assurance Activities :-

i) setting the checkpoint
ii) creating an SQA Mgmt. Plan.
iii) Apply Software Engineering techniques.
iv) Executing formal technical Review.
v) Hapling a multitasking strategy.
vi) Controlling Change
vii) Performing SQA audit
viii) Maintaining records & reports.
ix) Manage good relations.

* SOFTWARE TESTING :

04
10
2019

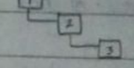
DATE : / /
PAGE NO. :

PROCESS MODELS

i) <u>Linear Sequential or Waterfall Model</u>
ii) <u>Prototyping Model</u>
iii) Rapid Application and Development Model (RAD)
iv) Evolutionary software Process Model
v) Component based development Model (CBD)

(iv) a) <u>Incremental development Model</u>
* b) <u>Spiral Model</u>

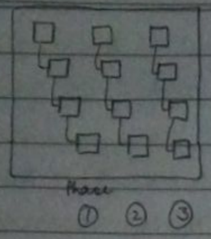
* i) The most basic process Model. All the steps are sequentially attached and flows in one direction only.
+ step by step



Drawback :
- If the developer comes on 3rd step and the user tells to modify and change anything this model fails. because it is not possible to again perform 1st two steps.

* ii) Depends on a formula. User tells requirements and a prototype is made on the base of requirement and modifications are done to satisfy the requirements. and a physical original model is made afterwards.
- User can deny about the nonfamiliarity of the model, this step / model fails.
↓
requirement fulfillment

iii) Best when a user provide a specific period of time for completion of model.
The best model under a specific period of time / time limits.

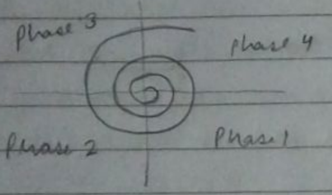


* combination of Waterfall & Prototyping

IV) Consideration of Production is available here.
focus on : first incremental analysis.

every increment / upgradation is studied & analysed.

SPIRAL MODEL
b) Hybrid model (Prototyping + sequential)
but based on Risk Driven Approach.



Mgmt Risks
Technical Risks
Economical Risks
Financial Risks

V) Developed because of getting rid of making models continuously.
fully Object Oriented technology based.

SOFTWARE COST ESTIMATION PROCESS

* Factors affecting Cost estimation :
1) Experience in application domain.
II) Product Complexity - application programs
- utility programs
- system programs
III) Project size
IV) Available Time
V) Programmer Ability
VI) Level of technology
VII) Required level of technology

Software Cost estimation process

I) Project Objectives and Requirements
↓
II) Plan Activity (WBS) — WORK BREAKDOWN STRUCTURE
↓
III) Estimate size — SMALL, MEDIUM, LARGE
↓
IV) Estimate Cost and effort — COCOMO MODEL
↓
V) Estimate Schedule — SCHEDULING SHOULD BE DONE
↓
VI) Risk Assessment
↓
VII) Inspect / Approve — TESTING and APPROVING AFTERWARDS
↓
VIII) Track estimates
↓
IX) Process Measurement and improvement

(x) a) Process effective Metrices

b) Process cost Metrices

DATE: / /

PAGE NO.:

S/W cost estimation is a form of problem solving and in most cases the problem to be solved are too complex to be considered in a single form.

Therefore, the problem is decomposed into components in order to achieve an accurate cost estimate.

* DECOMPOSITION TECHNIQUES

I) Problem Based Techniques

II) Process Based Techniques

* MATRICES

– Approaches:

* LOC (Line of Code) : $S = (S_o + 4S_m + S_p)/6$

– lines in the Source Code of the system.

Factors	S_p	S_m	S_o	Esize
UI	1400	1800	2200	1800
WP word processing	-	-	-	-
FSOF	-	-	-	-

Total = LOC

* FP (Function Point) : Measuring the functionality delivered by the system. effort ∝

PARAMETERS

EI	external Input (Numbers)	
EO	external Outputs (Numbers)	
ILF	Internal logical Files	DATABASE
EIF	External Interface files	INTERFACE

DATE: / /

PAGE NO.:

FP = Count * ~~waiting~~ weighting Factors

* COST ESTIMATION MODELS

I) Algorithemic - COCOMO Model

II) Non-Algorithemic Models - Failure Models

* COCOMO Model - PROBLEM BASED

Constructive Cost Model

Developed by BARRY BOEHM in 1980's

- categories under which to project lies:

a) Organic

b) EMBedded

c) semi-Detached

SIZE PARAMETER depends on KDLOC.

EXAMPLES

a) Size ≤ 50 KDLOCs

b) Size ≥ 300 KDLOCs (complex projects)

c) Size ≤ 300 KDLOCs

→ Ex: Os, compiler designs

→ Ex: Software used in Military Hardware

→ Ex: Small Business System, library management

COCOMO Model Hierarchy

I) Basic

II) Intermediate

III) Advanced.

$E = A \times (Size)^B$

E → Efforts

Size → KDLOC

A, B are constants

units (PM) person months

small sized projects

Intermediate → software Reliability, software complexity

Project Types	A	B
Organic	2.4	1.05
semi - Detached	3.0	1.12
Embedded	3.6	1.20

Ex: Pt = Organic

Size = 30 KDLOCs

$E = 2.4 \times (30)^{1.05}$

$E = \underline{85\ PM}$ PY Person Year

MID - TERM II SYLLABUS

→ Software Cost Estimation → LOC

→ COCOMO Model → Data flow diagrams

→ Software Architecture → Object Oriented Modeling language

→ Software design → Project Planning

→ Decomposition techniques

→ Functional Points

ii) Intermediate Model consider s/w reliability & complexity as parameters along with Size and effort.

Total 4 parameters.

step ① calculate an initiate estimate of effort.

↳ calculate E

step ② identify a set of 15 parameters, all parameters are rated against a numerical value called multiplying factors → calculate EAF

step ③ : calculate total effort = $\underline{EAF \times E_i}$

EAF : concerned particular value

	Parameters	Very low	Low	Medium	Normal High	High
①	RELY			4.6		

Example: $3.2 \times (45)^{1.05} = \underline{174\ PM}$

P T = Organic

Size = 45 KDLOC

Pt	A	B
Organic	3.2	1.05
S-D	3.0	1.12
Embedded	2.8	1.20

$\underline{EAF \times Ei}$

$$\underset{EAF}{0.8895} \; (\underset{\text{Reliability}}{1.15} \times \underset{\text{complexity}}{0.85} \times \underset{\text{Application Example}}{0.91} \times \underset{\text{programme Capability}}{1.00})$$

iii) Advanced Model : Here effort is calculated as a program size and a set of cost drivers for each phase of software engineering. this model incorporates all characteristics of the intermediate model and provides procedures for adjusting the phasewise distribution of the development schedule. It has 4 phases:

a) Requirement planning & Project Design. (RPPD)

b) Detail Design (DD)

c) Code and Unit test (CUT)

a) Integration and test (IT)

- for all these rating, cost drivers are assigned multiplying factors. these are called multiplying factors for analyst capapality (ACAP).

↳ Calculated using 25 parameters

- Total Effort = $E_i \times ACAP$

↓

174 × ACAP = $\underline{174(ACAP)\ PM}$

* SYSTEM ANALYSIS :

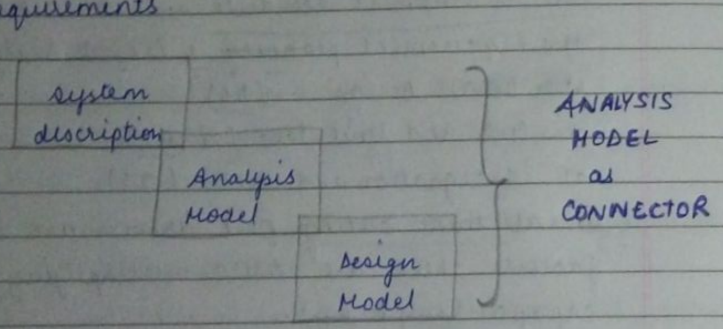
Q: What is software requirements?
Requirement is a cond^n or a capability posed by s/w or system component in order to solve a real world problem. Requirements describe how a system should act, appear or perform. for this when users requests for s/w they possess an approximation of what the new system should be capable of doing.

- **Requirement Analysis :**

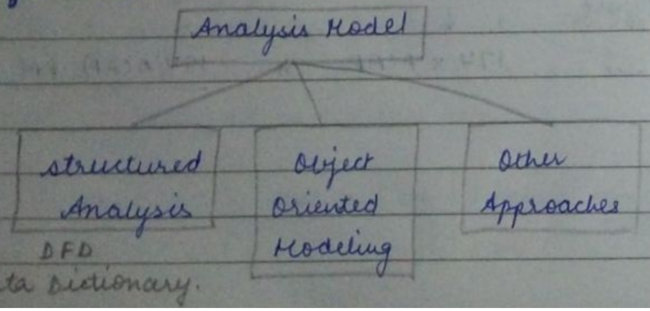
IEEE defines :

I) the process of studying user needs to arrive at a definition of system, H/W or S/W requirements.

II) the process of studying and refining system, H/W, S/W requirements.



- **Analysis Model :**



A) Structured Analysis :

I) Data Flow Diagram (DFD) → Level 0, 1, 2, 3

II) Data Dictionary

B) Object Oriented Modeling :

C) Other approaches.

- **Data Flow Diagrams**

IEEE defines DFD (a.k.a. Bubble Chart, workflow diagram) as a diagram that depicts data sources, data sinks, data storage and processes performed on data as nodes and logical flow of data as links b/w the nodes.
DFD should not be confused with a flowchart. A DFD represents the flow of data whereas flowchart depicts the flow of control.

	NAME	NOTATION	DESCRIPTION
I)	External entities	Rectangle	
II)	Data Flow	→ ←	DOWN BELOW ↓
III)	Data Store	═	
IV)	Processes	Circle / Oval	

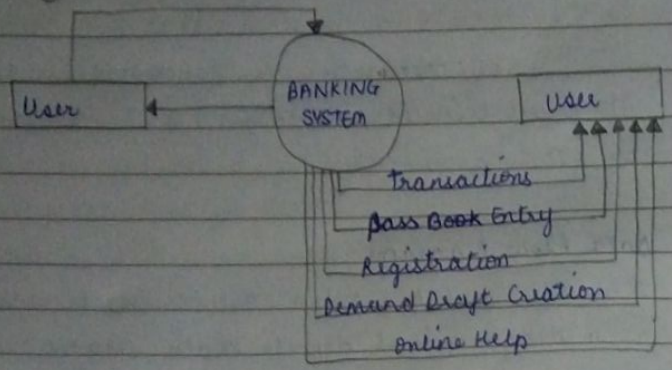
I) represents the source & destination of data within the System. Each EE is identified with a meaningful and unique name.

II) represents the movement of data from its source to destination within the system.

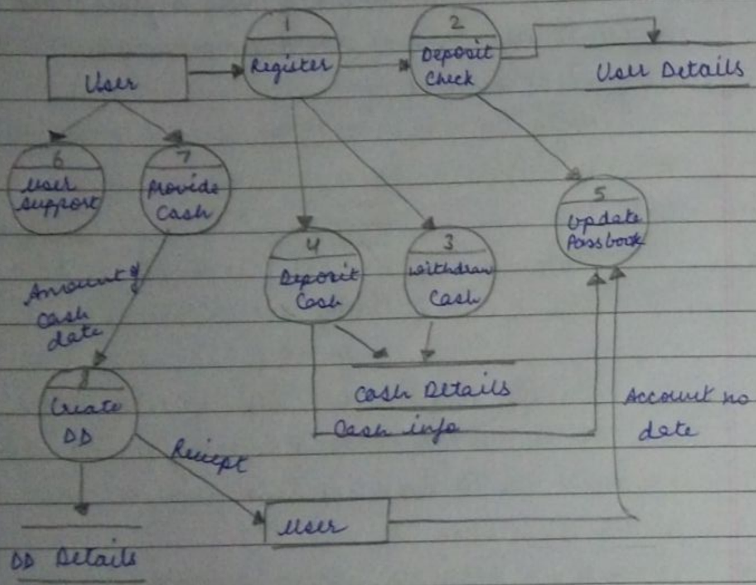
III) indicates the place for storing information within the system.

IV) shows a transformation or manipulation of data within the system. A process is also known as BUBBLE.

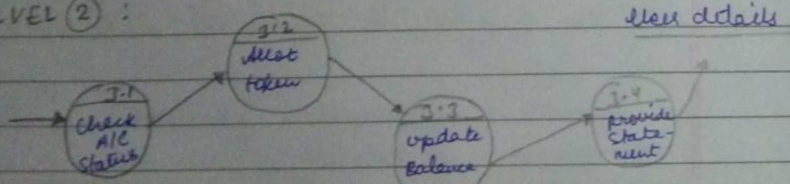
* LEVEL (0):



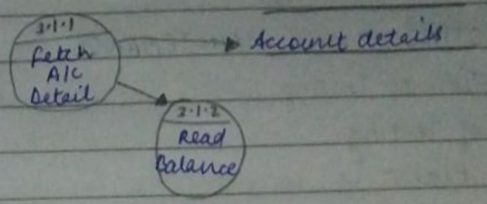
* LEVEL (1):



+ LEVEL (2) :



+ LEVEL (3) :

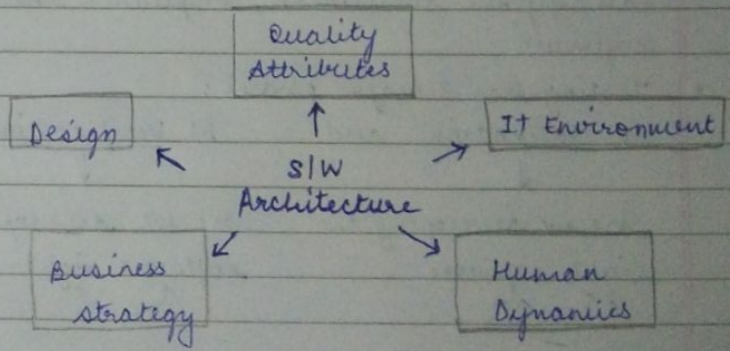


+ OBJECTED ORIENTATION ~~AND~~ MODELING :

Identyf

08.11.19 - SOFTWARE ARCHITECTURE - UNIT 3

* Architecture serves as a blueprint for a system, It provide an abstraction to manage the system complexity and establish a communication and coordinator mechanism among the components.
It defines a structured solution to meet all the technical and operational requirements while optimising the common quality attributes like performance and security.



15/11 2019

DATE : / /
PAGE NO. :

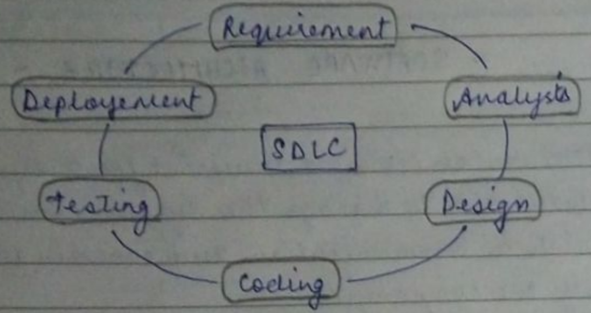
- SOFTWARE DESIGN -

15/11/2019

* - SOFTWARE TESTING -> and MAINTENANCE

i) Introduction to testing
ii) Concepts of Testing
iii) Testing Plans
iv) Testing Principles
v) Testing strategies
vi) Types of testing

*



- testing is an important phase of SDLC. Its main objective is to detect errors in the software.
In IEEE relation, ST is a process which is used to identify the correctness, completeness and quality of the software.

- Testing has 2 major phases:

a) Validation and b) Verification
↓ ↓
Are we developing the right software. Are we developing the software right. // important

▾ Verification refers to checking or testing of items, including software, consistancy with an associated specification.

DATE : / /
PAGE NO. :

ex: (inspection, analysis of all items), Reviews, walkthrough

▾ Validation refers to the process of checking that the developed S/W is according to the requirements specified by the user.

+ Principles of Techniques of Software Testing:

i) Define the expected output,
ii) Inspect output of each test completely,
iii) Include test cases for invalid and unexpected conditions.
iv) Test the modified program to check its expected performance.