Date

5.10.12

- Pressman
- K.K. Aggarwal

- What is software?

Ans. Software is a program, when it is executed the associated functionality must be satisfied.
S/w is a data structure used to manipulate the information.
S/w is a document used to maintain the operational guidelines of the program.
∴ S/w is a combination of:-

(program + documentation + operational procedures)
- program: code
- documentation: analysis, design, coding, testing
- operational procedures: System overview, guidelines to naive users.

S/w is a logical entity rather than physical entity.

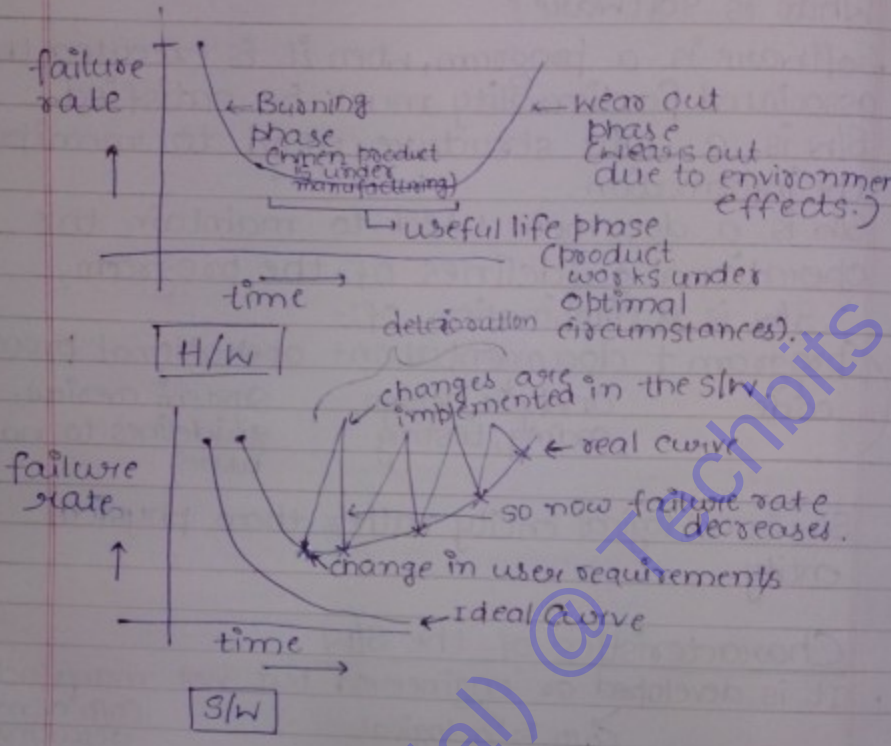
Characteristics of the S/w

(i) · It is developed or engineered but not manufactured.
- Co/p is a logical entity. [It contains analysis, design, coding, testing, etc.]
- Co/p of manufacturing is a physical entity. [It contains analysis, design, manufacturing process, testing]

- S/w is a logical component & h/w is a physical component.
After implementing the design, if we get physical component as o/p, then that process is called manufacturing process,
& if after implemention of design, the o/p we get is a logical component, then process is called development.

Cosmos (Atal) @ Techbits

(ii) S/W doesn't wear out, but it deteriorate.



★ When S/W undergoes deterioration, we have to re-engineer the product.

Note:- Ideal curve :- When customer requirements are ~~station~~ static
Real curve :- When customer requirements are dynamic.

H/W undergoes wear out, which means at early stage of development the uncovered errors causes the failure rate of product high. This state is called burning state. After maintainence of product, the defects are removed from the produce, so failure

Cosmos(Sajal) @ Techbits

rate decreases. Then, the product will be working in the specified levels. This phase is useful life phase. Over a period of time, due to environmental changes (dust, temp., etc.) the product failure rate increases. This phase is wear-out phase. When product undergoes wear-out, replace the component with new component.

Deterioration :-

- S/w undergoes deterioration, which means user requirements are not static, due to the new requirement changes, the product failure rate increases. After
- After maintainence, the defects will be minimized but due to the frequent changes, the product maintainence cost becomes twice or thrice of the development cost. This condition is called as deteriorate.
- When S/w deteriorates, then, re-engineer the S/w.

(iii) Industry moving towards component based development, but still S/w is custom-built. (components are available in the market)

- Component is an error free code or reusable code or fully tested code.
- In the physical component development, the components are directly used in the process w/o changes (diff. IC's)
- But in the S/w development, the components are modified acc to the problem objective.

Cosmos (Sajal) @ Technits

A Slw develop components are divided into 4 types:-

- Off-the-shelf
- Fully experienced
- Partially experienced
- New components

- Off-the-shelf:-
components are extracted from the 3rd party libraries & used directly in the current project.

- Fully or Partial experienced components are extracted from current organisation library & used in the project.

- The new components functionality are developed from the base line.

Software Applications :-

- There are diff. application domains that are using the slw to satisfy the objective within the run time period.

(i) System Software:-
It is a program used to manage the other program e.g. compiler, OS, device drivers.

(ii) Real time Slw:-
This slw satisfies the high level objectives of the real world entities, e.g. weather forecasting, aircraft simulation.
(If there is a latency in user action & slw reaction, then it may cause serious damages.)

Cosmos(Sajal) @ Techbits

(iii) Business S/w :-
It is used to generate the reports. Various programming languages are used in this domain, e.g. COBOL, OLAP Tools

(iv) Engg. & scientific S/w :-
It is used to perform the high level computation, e.g. CAD/CAM, MATLAB, NS2. (within limited time.)

(v) Embedded S/w :-
This s/w is stored in the permanent memory of the devices (ROM).
When the device is in the operational state, then the s/w is executed, so the associated functionality ~~must be~~ is satisfied.

(vi) Web S/w :-
By using the web s/w, we can develop user interfaces e.g. web services, HTTP etc.

(vii) Artificial Intelligence :-
In this domain, symbolic computations are performed rather than numeric computation, e.g. knowledge representation, image processing, speech recognition, machine learning.
To perform the symbolic computation, functional programming languages are used, e.g. LISP (LIST Processing).
Prolog (Logic Programming), Haskel, etc.

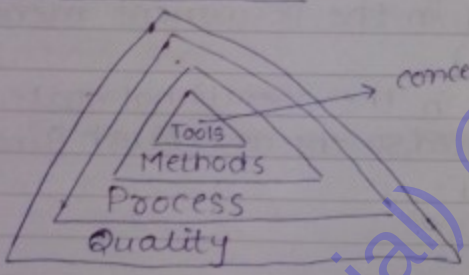
☆ To satisfy various high-level objectives, there is a need to develop efficient s/w. S/w engg. practices are required to develop the quality s/w.

Cosmos (Sajal) @ Techbits

• S/w Engg:-
The application of systematic, disciplined & quantifiable approach to develop, operate & maintain the s/w.

1. Layered Approach
2. Generic Approach

Layered Approach



Layers to develop Quality S/w.

(i) The quality layer is a bedrock of the s/w. Quality is defined as confirmation to explicitly stated requirements.
Quality is an indirect measurement, so it depends on defects.

$$\text{Quality} \propto \frac{1}{\text{defect}}$$

• The uncovered error becomes the defect during the operational state of the s/w.

(ii) Process layer focus on key process area, i.e. the product, project, people, process to trace the cost, effort & schedule.
(There is no measurement related to 'people'.)

Cosmos (Sajal) @ Techbits

(iii) Method layer focus on the framework i.e. process model to develop the quality S/W.
(iv) Tools layer focus on the automated & semi-automated tools present in the 4th generation techniques (focus on 4th generation language.)

Generic Approach :-

Generic approach comprises of 3 phases of operations
(i) Definition → focus on 'What' [e.g. what behaviour is expected from system i.e. analysis & info. gathering (i/p & o/p) of the system.]
(ii) Development → focus on 'how' ['how' to convert i/p to o/p → (Designing, Coding & Testing)]
(iii) Support.
↓
For maintainence purposes. (In operational mode)
(first maintainence is corrective maintainence)

Definition :-

This stage focuses on 'what', i.e 'what' info. is processed, 'what' functionality or performance is designed & 'what' behaviour is expected. 'What' constraints are imposed & 'what' validation criteria is used.
∴ In this stage requirements analysis & req. gathering activities are performed.

Validation :- "Did we build the right system ?"
The validation process is injected into the early stage of the S/W development to minimize the maintainence effort & cost.

Verification :- "Did we build the system rightly"
Verification activity is performed in the testing

Cosmos(Saiaj) @ Techbits

stage of the S/W development life cycle.

Development:-
This stage focuses on "how" (design), i.e. how to create the data structure, 'how' (design) to define the interfaces, 'how' (code) to translate the procedural description into machine readable format & 'how' to define & implement the test cases.

→ In this stage design, coding & testing activities are performed.

Support:-
After deploying product at the customer place, support stage is required to maintain the product from various side effects.
There are 4 types of support present:-

(i) Correction:- In this support, the uncovered errors are corrected, so corrective maintainence is required to cover the uncovered errors.

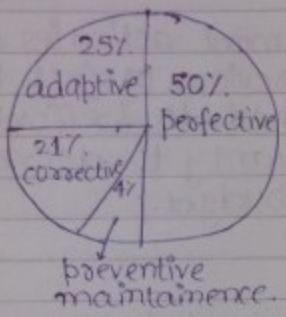
(ii) Adaption:- This support requires adaptive maintainence to maintain the product due to the platform changes, e.g. due to CPU/ Mem./OS upgradation.

(iii) Enhancement:- This support is required when the functional requirement changes. Perfective maintainence is required to maintain the S/W from the functionality changes.

(iv) Preventance:- This support uses preventive maintainence to maintain the S/W from frequent changes.

Cosmos(Sajal) @ Techbits

The maintainence effort is distributed as follows:-



Software Process

- Process is a framework, there are diff types of the framework (process models) are used to develop the quality s/w.

Software engg. institution define one assessment model to assess the s/w organisation based on their level of process development.
The assessment model is named as CMM (Capability Maturity Model).
This model consists 5 levels of assessment, i.e.:-

(i) Level-0 :- Initial. (No standard model to develop the s/w.)
(ii) Level-1 :- Repeatable (management activities are defined)
(iii) Level-2 :- Defined
(iv) Level-3 :- Managed
(v) Level-4 :- Optimized

→ Initial:-
In this level, adhoc procedures are used to develop the s/w, ∴ the process becomes chaotic. So success/failure depends on the individual

Cosmos(Sajal) @Techbits

effort

→ Repeatable:-
In this level, management activities are defined based on the previous project knowledge to trace the cost, schedule & effort of the project, ∴ no guaranty to deliver the product on time & in-budget.

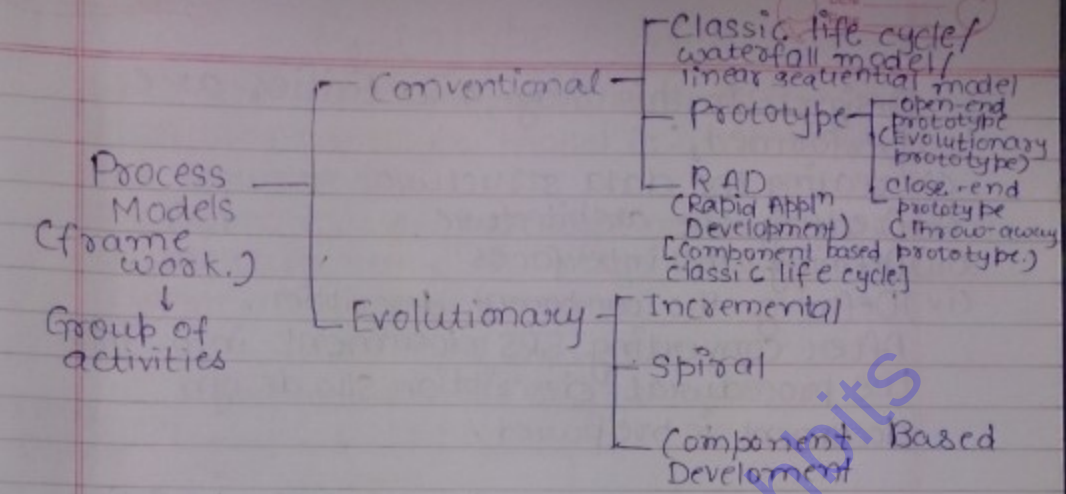
→ Defined:-
In this level, management & engg activities are defined but not properly documented, ∴ no soln. for the anticipated situation.

→ Managed:-
In this level, engg. & management activities are qualitatively documented & controlled. Due to lack of continuous improvement min. quality will be assured.

→ Optimiced:-
Gain more profit with min. effort is the objective of the optimized level. In this level, innovative tools & techniques are used & continuously taking the qualitative feedback from the customers.

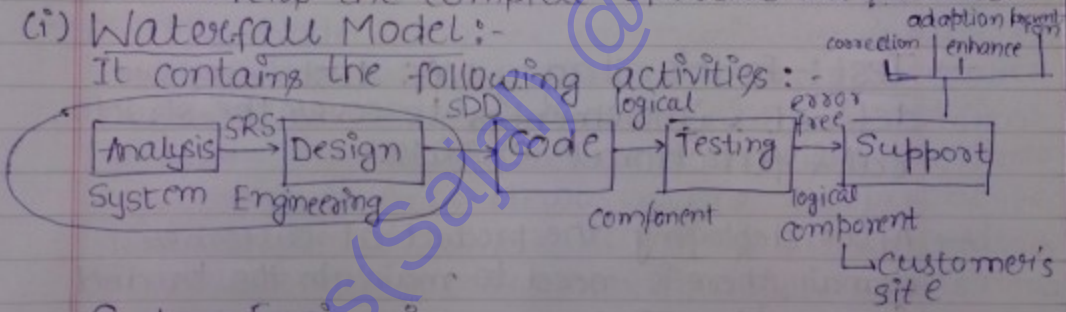
Process Models:-
The classification of the process models are shown below:-

Cosmos(Sajan) @ Techbits



- Conventional Model:-
By using the conventional process model, we can develop the complete operational product.

(i) Waterfall Model:-
It contains the following activities:-



System Engineering :-

(i) Software is a subset of the system, so to develop the s/w there is a need of analysing the existing system to finalise the goals of the product.

→ Analysis :- In this stage, based on the goals of the product, diff. requirements are finalised. All of the right requirements are placed in SRS document. SRS document consist goal of the s/w, functional requirement, non-functional requirements.

Cosmos(Saiel) @ Technits

→ Design:- In this stage, 4 activities are performed,:-
(i) creating the data structure.
(ii) Creating the architecture.
(iii) Defining the interfaces
(iv) Defining the component description.
After converting SRS document info. into the procedural description, s/w design document is prepared.

→ Code:- In this stage, procedural description is converted into machine readable format by taking the appropriate language from 4GL.

→ Test:- In this stage, diff. test cases are defined & implemented to cover the structural & functional errors.

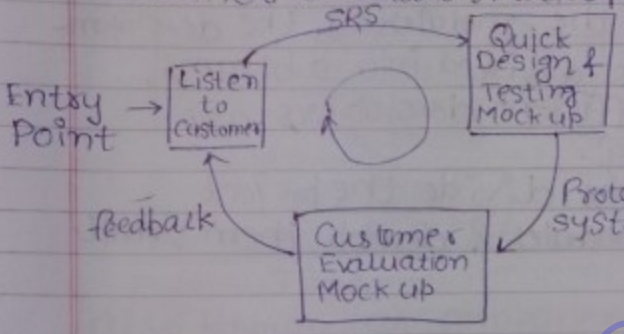
→ After deploying the product at customer domain, there is need to maintain the product from side effects

ote: Waterfall model is suitable when the custome requirements are clear initially & customers can wait because in this model, the final product is available after a long schedule. It is not suitable to develop complex systems because it is very diff. to determine all requirements at a time.

Cosmos(Sagal) @ Techbits

Prototype Model

When the customer requirements are not clear, then prototype model is used to finalize the requirement.
The framework of prototype model is to listen to the customer, Quick design & testing mockup, & customer evaluation mock up.



- In this model, customer communication is allowed. Based on the customer requirements, the developer prepares the SRS document. This SRS is not final. Based on the SRS, the developer develops the prototype product. After demostrating the prototype product, the developer accepts the feedback from the customer. When feedback consists new requirements the SRS document is modified, otherwise the corresponding SRS is finalized.
- After finalizing the SRS document, if we still use the same framework to develop the final product is open-end/evolutionary prototype.
- After finalizing SRS, if we use diff. process model to develop the product is called throw away prototype model.

Cosmos(Gajal) @ Techbits

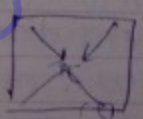
RAD

(i) RAD model is used when the customers req. are more clear & the project schedule is very short

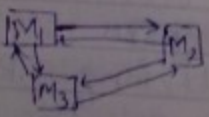
(ii) To develop the product in short schedule, there is a need of using the modularity concept
Before taking the modularity, the development team is divided into subteams, that means efficient developers are req. in the orgn.

(iii) Modularity means divide the project into smaller modules & develop them simultaneously.

(iv) Two factors
→ Cohession
→ Coupling
Cohession is a major measure of to what extent the module is independent from other module
→ Coupling is a measure of what extent the module is depend of the other module
The effective modulating maintains the high cohession & low coupling.



Cohession

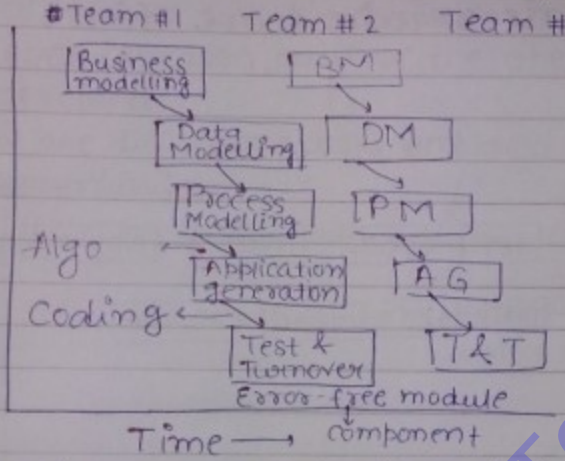


Coupling

→ With respect to Modularity concept, all the modules are developed simultaneously. After development, there is a need of

Cosmos(Sajal) @ Techbits

integration to develop the final system.
The framework of RAD model is



also known as component based classic-lifecycle.

- In the Business modelling, the goals of the product are finalized.
- In the data modelling, diff. data objects are required to satisfy the goal of the projed.
- In the process model, the info. flow is created by adding/deleting/modifying the data objects.
- In the application generation, the procedural description of problem statement (algo) is converted into machine language.
- In Test & Turnover stage, various test cases are implemented to cover the diff. errors.

Note: The objective of the testing is to prove the system upto what extent the system
At the end of test & turnover, we have an error-free module (components), ∴ integrate all the components to develop the final product.

Cosmos(Sajal) @ Techbits

Note: RAD model maintaing less effort during testing phase, whereas in waterfall model, testing phase require more effort.

Cosmos(Sajal) @ Techbits

Date

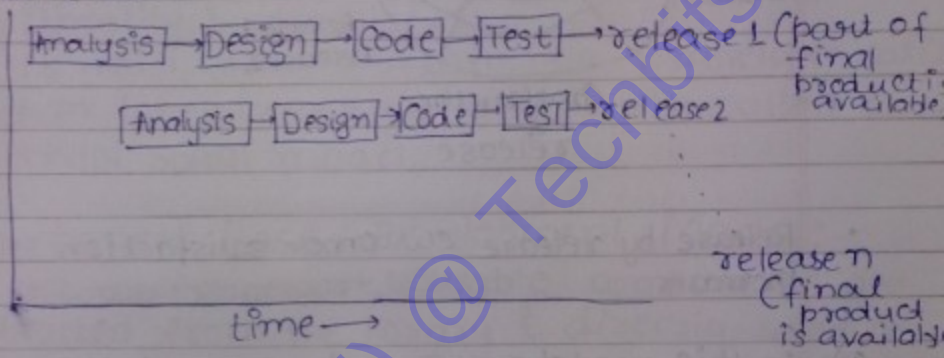
07.10.12

Evolutionary Process Model :-

In the evolutionary process model, the final operational product will be available after some iteration in each iteration, part of the operational product is developed.

There are diff frameworks employed under this category:

① Incremental:-



This model uses the classic life cycle framework to develop the product version by version (or part by part).

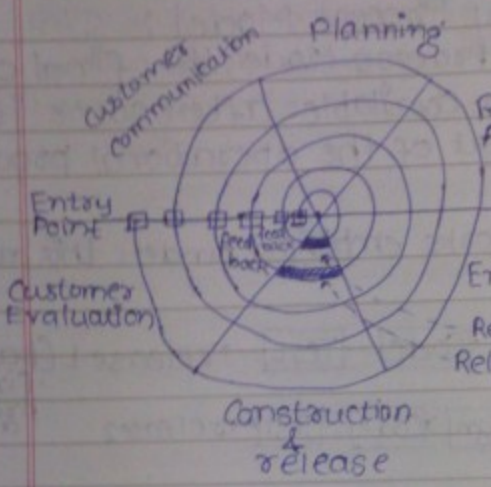
② Spiral:- Spiral Model :-

This framework consists diff stages:-

① Customer communication (produces SRS)
② Planning (perform estimations)
③ Risk Analysis
④ Engineering
⑤ Construction & Release
⑥ Customer Evaluation

Cosmos(Sajal) @ Techbits

WIN-WIN Spiral Model



- Release by release customer satisfaction increases.

(i) In this model, customer involvement takes place, i.e. continuously taking the qualitative feedback from the customer.
(ii) The entry point in this model to develop the product is customer comm^{n}.
(iii) Based on the customer req. SRS document is finalised.
(iv) During the planning stage, diff. parameters of the S/W undergoes estimation, e.g. size, effort, schedule, cost, etc.
(v) In this model risk impact is calculated during the risk analysis stage.
(vi) After taking the Risk impact, it is added to the project budget to develop the SW on-time & in-budget.
(vii) In the engineering stage, the design practices are defined (Data structure, Architecture, etc.)

Coshos (Saral) @ Techbits

(viii) After converting SRS into procedural description (Algo) use. By using diff. tools & techniques, they develop the code

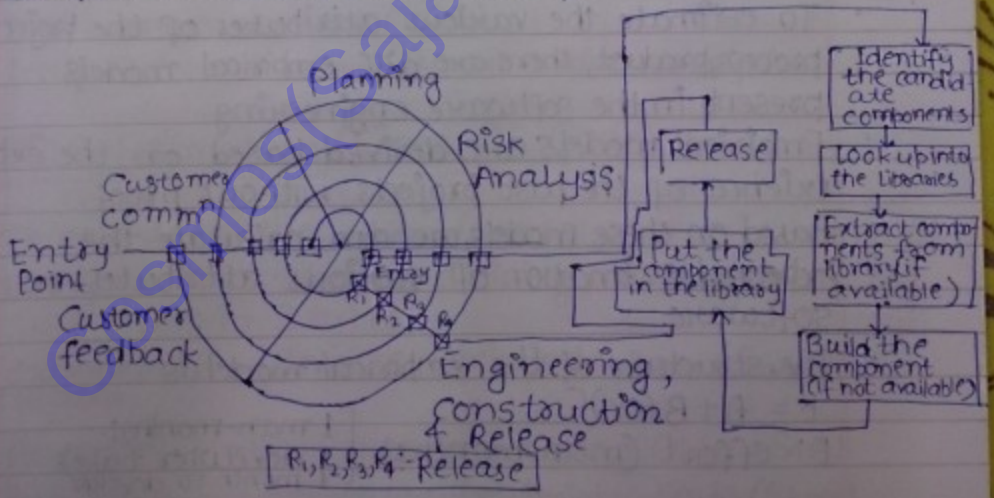
(ix) After implementing diff. test cases, the product will be released at customer's site

(x) After releasing the product, continuous feedback is accepted from the customer to improve the S/W functionality & reliability.

(xi) In this model, customer satisfaction & developer satisfaction levels are high. So, it is also called as WIN-WIN Spiral model.

Component Based Development Model:-

- In this framework, reusable components are extracted from the library & directly used in the project development. The framework of the component based development is



Cosmos(Sajal) @ Technoits

- This process model uses the spiral model framework, but during the development diff. components are extracted from the libraries. If the library is a 3rd party library, then the extracted components are called off-the-shelf components.
- If the library is own-organisation library, then extracted components are called as fully/partially experienced components.
- The new component functionality are developed from the base line.
  After developing new components, place them in the library, so that they may become reusable components for the future projects.

Estimation Models:-

- To estimate the various attributes of the project process, product, there are diff. empirical models present in the software engineering.
- Empirical models are derived based on the experience of the past projects without proof.
- Based on these models, we can calculate the expected estimation of various attributes w.r.t. software.
- The structure of the empirical model is

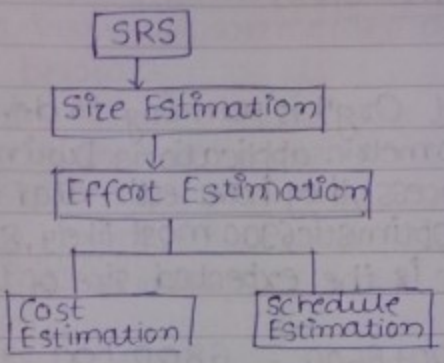
  $E = A + B(ev)^C$

  E:- effort (man-months) [1 man-months:- 1 developer takes 1 month to develop OR 12-man month:- 1 developer takes 12 months to develop]

  A, B & C:- empirically derived constants
  ev:- estimated value (size).

Cosmos(Sajal) @ Techbits

The sequence of S/W attributes estimation is:-



Size Estimation :-

- The size of the S/W can be estimated based on the SRS document.
- The size of the S/W can be measured in 2 ways:-
  - → Direct measurement (LOC)
  - → Indirect " (Function Point Analysis).

LOC Estimation (Lines Of Code) :-

To calculate the expected size of S/W in terms of LOC, there is a need of dividing the programming skill levels in 3 types:

(i) Experienced → develops Optimistic Code (S_{opt})
efficient

(ii) Average → develops most likely Code (S_m)

(iii) Below Average → develops pessimistic code (S_{pess}).

After taking above skill levels, we can use the following formulas to calculate the expected size of S/W :-

Cosmos(Saral) @ Techbits

$$S = \frac{S_{opt} + 4S_m + S_{pess}}{6}$$

Q1. Slw development Orgn is planning to develop the slw for geometric applications. During the development process, they have estimated the code as 4600 optimistic, 6900 most likely, 8600 pessimistic, what is the expected size of the slw?

Ans. $S = \frac{4600 + 4 \times 6900 + 8600}{6} = 6800$ LOC

(b) If the above slw is expected to develop within 12-man-months effort, what is the productivity?

Ans 1 developer will work 12 months to develop 6800 Lines of Code.

Productivity = Lines of code developed in one month

$\therefore$ Prod = $\frac{6800}{12} = 566.67 = 566.67$

(c) If 12-man months effort project is developed in 5-months duration, what is the man-power.

Ans. $1 \times 12 = n \times 5$

$n = 3$

3 developers are req.

Q2 A company needs to develop the strategy for developing the slw in DSP, in which 2 diff. programming languages are planned. The LOC developed using lang 2 is estimated

Cosmos(Sajal) @ Technite

to be twice the LOC developed using lang. 1. The product will have to be maintained for 5 years. Various parameters of the product is given below :-

	L1	L2
man-years req for development	$\frac{LOC}{10,000} = x$	$\frac{LOC}{10,000} = 2x$
dev. cost/ man-year	Rs. 10,00,000 [total → x × 10 lakh]	Rs. 7,50,000 [total → x × 15 lakh]
cost of maintence per year	Rs. 1,00,000 [total → 5 lakh]	Rs 50,000 [total → 2.5 lakh]
Maintainence year	5 years	5 years

Total cost of the product includes cost of the development & maintainence, what is LOC for L1 for which the total cost of project using L1 = total cost of project using L2

ans. $x \times 10 \text{ lakh} + 5 \text{ lakh} = 15 \text{ lakh} \times x + 2.5 \text{ lakh}$

$2.5 \text{ lakh} = 5 \text{ lakh} \times x$

$x = \frac{1}{2}$

$LOC = x \times 10,000$

$= \boxed{5,000}$ [for L1]

$\frac{1}{2}$-man year effort is req. [for L1]

(means 6 months req. to develop 5,000 LOC)

$Prod = \frac{5,000}{6}$ [i.e LOC developed in 6 month].

Cosmos(Sagar) @ Tech Bros

Function-Point Analysis:

(i) FP Analysis is an indirect measurement, used to measure the size of the s/w.
To calculate the expected size of the s/w in terms of FP, there is a need of other parameters.

(ii) Five functional parameters & 14 non functional parameters are req. to calculate the FP of the s/w.

Functional parameters are directly involved in the project:-

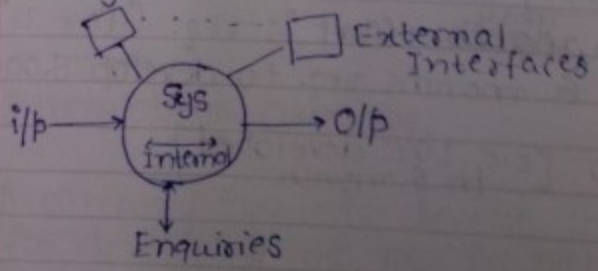
① no. of i/p's:- it indicates no. of inflows to the system.

② no. of o/p's :- it indicates no. of outflows from the system.

③ no. of enquiries - it indicates no. of ways to get the quick info. from the system.

④ no. of files :- it indicates the logical connections within the system.

⑤ no. of external interfaces:- it indicates no. of the other modules interfaced with our system.



Cosmos(Sajal) @ Techbits

- The five parameter values can be calculated by using :-

$$S = \frac{S_{opt} + 4 \times S_m + S_{pess}}{6}$$

(i)	S_{opt}	S_m	S_{pess}	(z) $S = \frac{S_{opt} + 4 \times S_m + S_{pess}}{6}$
# of i/p:				
# of o/p:				
# enquiries:				
# of files:				
# of external interface:				

- To calculate the count value of the 5 functional parameters, there is a need of weighing factor.
- Weighing factor is derived empirically based on the level of complexity of the project, i.e.

Weighing factor :-

(j) simple (1)	average (2)	complex (3)	
3	4	6	W
4	5	7	
3	4	6	
7	10	15	
5	7	10	

$$\text{Count} = \sum W_{ij} \times Z_{ij}$$

i :- type of the i/p parameter
j :- level of complexity
w :- weighing factor.
z :- value of the parameter.

★ Non-functional requirements are indirectly involved in the project there involvement in the project

Cosmos(Sajal) @ Techbits

Involvement	Impact
If no influence	0
If Incident	1
If moderate	2
If average	3
If significant	4
If essential	5

VAF (value adjustment factor) =
$0.65 + 0.01 \times \sum_{i=1}^{14} F_i$

now,

FP = count value × VAF

460
0.65+0.01×3×14
0.42
tp.65
1.07

Q. Consider the s/w project with no (weighting factor) of i/p's, o/p's, enquiries, files & external interfaces as 30(4), 25(5), 20(4), 10(10), 5(7). Assume all complexity AF are constant i.e. 3 What is the FP for the project

Ans. Count = 30×4 + 25×5 + 20×4 + 10×10 + 5×7
= 460

1.07
×46
642
428×
49.22

VAF = 0.65 + 0.01 × 3 × 14
= 1.07
FP = 460 × 1.07
= 492.2

(b) If the above project requires 2-man years effort, what is the productivity.

Ans. 492.2 / 24 — 24 man-months → 492.2
1 man-month → 492.2/24 (prod.)

Cosmos(Gaal) @ Techbits

Q If in a S/w project no. of i/p's, no. of o/p's, enquiries, files & external interfaces are 30, 60, 20, 10 & 5 respec., with complexity & weighing factors are avg., what is the prod. of the project if the effort is 100 man months.

Ans. Count = 120 + 300 + 80 + 100 + 35
= 635

VAF = 0.65 + 0.01 × 3 × 14
= 1.07

FP = 635 × 1.07
= 679.45

prod. = $\frac{679.45}{100}$ = 6.7945 (FP's developed in 1-month).

Q A company has to develop S/w for an application. The size of the project can be estimated by using the FP analysis with the following values:-

	Sobt	Smp	Sbess	S
# of i/p's	20	24	30	24.33
# of o/p's	12	15	22	15.67
# of enquiries	16	22	28	22
# of files	4	4	5	4.167
# of ext. interfaces	2	2	3	2.167

with complexity & weighing factors as avg, what is the productivity of project, if effort is 2 man-years?

Ans. Count = 4 × 24.33 + 5 × 15.67 + 22 × 4 + 41.67 + 7 × 2.167
= 97.32 + 78.35 + 88 + 41.67 + 15.169
= 320.509

VAF = 1.07

FP = 342.94

∴ prod = $\frac{342.94}{24}$ = 14.28 person-month

Cosmos(Sajal) @ Techbite

Halstead size estimation :-

1. To measure the various attributes of the program, Halstead defined diff formula based on the token mechanism.
2. In the LOC count, all the comments & blanks & hash directives are included, so it doesn't gives the accurate estimation, i.e token mechanism is used to estimate the size of the slw. In the token mechanism each statement is decomposed into operands & operators. Variables & constants are treated as operands.
3. Function calls & different unary operators & binary operators are treated as operators.
4. The reserved words like return, break, continue, etc. are considered as operators.
5. Diff control structures like if, switch, while, do while are considered as operators.
6. The pair symbols i.e {}, (), [], special symbols & termination operators are considered as operators.
7. Goto label, in this Goto → operator
   label → operand.
8. In the array accessing
   array_name[index], array_name & index → operand
   [] → operator.
9. Comments & function declarations & hash directives are excluded.

Cosmos(Sagar) @ Technite

e.g. a = a + (b * c);

Operand	Operator
a	=
b	+
c	*
	( )
	;

after decomposition the statement into token, database is required.

The Database consists of following :-

# of unique operators	# of occurrences of the unique operator in the program	# of unique operand	# of occurrences of unique operand in the program
=	1	a	2
+	1	b	1
*	1	c	1
( )	1		
;	1		
$\Sigma = n_1$	$\Sigma = N_1$	$\Sigma = n_2$	$\Sigma = N_2$

- After making the database, we can calculate diff attributes of the program.
- the vocabulary of the program :- $n = n_1 + n_2$
- the length of the program :- $N = N_1 + N_2$
- the volume of " " :- $V = N * \log_2 n$
- the estimator level of the program :- $L = \frac{2}{n_1} \cdot \frac{n_2}{N_2}$ (program level is in the range of 0...1) if L=0, then program is optimistic if L=1, then the size of the program is low & optimistic code is maintained.
- diff of program is :- $D = \frac{1}{L}$
- effort of program :- $E = \frac{V}{L}$ (not in man-months)
- the estimated length of program :- $\hat{N} = n_1 \log_2 n_1 + n_2 \log_2 n_2$
- the programming time is calculated as $T = E/\beta$, $5 \le \beta \le 20$ (normally $\beta = 18$). time is in seconds.

Cosmos(Sajal) @ Techbits

Q1. Consider the following program & calculate diff. attributes of the program.

```
int sort(int x[], int n)
{int i, j, save, im1;
 if (n<2) return 1;
 for (i=2; i<=n; i++)
  {im1 = i-1;
   for(j=1; j<=im1; j++)
     if (x[i]<x[j])
       {save = x[i];
        x[i] = x[j];
        x[j] = save;
       }
  }
 return 0;
}
```

Operators	occurences	Operands	occurences
int	4	x	7
[]	7	n	3
()	5	i	8
{}	3	j	7
;	11	save	3
if	2	im1	3
<	2	2	2
return	2	1	3
for	2	0	1
,	4	sort	1
=	6	10	38
<=	2		
++	2		
-	1		
14	53		

Cosmos(Sajal) @ Techbits

(n) vocabulary = 24 ($n_1 + n_2$).
(N) length = 91 ($N_1 + N_2$)

$\hat{N} = 14\log_2 14 + 10 \times \log_2 10 = 86.5$

$V = 91 \times \lceil \log_2 24 \rceil$
$= 91 \times 5$ bits
$= \boxed{455 \text{ bits}}$

$L = \frac{2 \times n_2}{n_1 \cdot N_2} = \frac{2 \times 10}{14 \times 38} = \frac{20}{14 \times 38} = 0.037$

$D = \frac{1}{L} = 26.6$

$E = \frac{V}{L} = \frac{455}{0.037}$ 12,297

$T = 683.16$ (when $\beta = 18$).

Effort Estimation :-

- Based on the size of the S/W, we can estimate the effort required to develop the S/W.

Diff. empirical models are used to estimate the effort. The structure of effort is :-

$\boxed{E = A + B(ev)^C}$

A, B & C :- empirical constants
ev : - estimated value size (KLOC/KDSI/FP)
* (ev will accept Line of Code in KLOC) Delivery of source ins^n

There are 3 empirical models used to estimate the effort :-

- SEL (S/w Engineering Lab)
- W-F (Walstonald-Felix) [developed by IBM]
- COCOMO (Cost Constructive Model) - Boehm's

SEL :-

In this model, the effort & duration, both the parameters are calculated as

$E = 1.4(KLOC)^{0.93}$ man-months $[A = 0]$
$D = 4.6(KLOC)^{0.26}$ months

Cosmos Sajal @ Techbits

W-F Model :-

In this model, the effort & duration req. to develop the S/W is calculated as

$$E = 5.2(KLOC)^{0.91} \quad [A = 0]$$
$$D = 4.1(KLOC)^{0.36}$$

Q. S/w development expected to have 8-man years of the effort. Calculate LOC, Duration, Productivity & avg manning by using SEL & W-F models

Ans. SEL :-

$$96 = 1.4(KLOC)^{0.93}$$
$$68.57 = (KLOC)^{0.93}$$
$$94.26 = KLOC$$
$$\therefore LOC = 94260$$
$$D = 4.6 \times (94.26)^{0.26}$$
$$= 15 \text{ months}$$

$$Prod = \frac{96 \text{ man-month}}{15 \text{ months}} \frac{(LOC)}{(E)} \frac{94.26}{96} = 0.9819 \text{ KLOC/month}$$

Avg. Manning = $\frac{E}{D} = \frac{96 \text{ man-month}}{15} = \boxed{7}$

(How many developers are req. to develop S/w of 8-man months into 15 months)

Cosmos(Sajal) @ Techbits

COCOMO - MODEL :-

- Cocomo model was designed by the Bohem, acc. to Bohem's analysis the projects are classified into 3 types, i.e.

→ Simple
→ Average
→ Complex

Based on the level of complexity, diff empirical constants are maintained in 3 diff. modes :-

① Organic model (2 - 50 KLOC)
   2. Simple
   3. ~~Similar~~ Familiar Appⁿ Domain
   4. Experienced developers
   5. Deadlines are not tight

② Semi-Detached: 1. 50-300 KLOC
   2. Medium
   3. Deadlines are also medium.

③ Embedded-Mode: 1. >300 KLOC
   2. Complex
   3. Deadlines are very tight.

Cocomo model is divided into 2 types :-

1. Basic Cocomo
2. Intermediate Cocomo

Basic Cocomo is used for quick & rough estimation, ∴ in this model accuracy is not possible. The structure of the effort & duration estimation is

$$E = C_1 (KLOC)^{P_1} \text{ or } C_1 (FP)^{P_1}$$

$$D = C_2 (E)^{P_2}$$

Mode	C_1	P_1	C_2	P_2
organic	2.4	1.05	2.5	0.38
semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.2	2.5	0.32

Cosmos (Saroj) @ Techbits

Q. Suppose that a project was estimated to be 400 KLOC, then calculate the effort & duration by using three modes

Ans. Organic :-

$E = 2.4(400)^{1.05} = 1295.31$ man-months

$D = 2.5(1295.31)^{0.38} = 38.07$ months

5 Dev

Semi-Detached :-

$E = 3(400)^{1.12} = 2462.7$ man-months

$D = 2.5 \times (2462.7)^{0.35} = 38.45$ months

65 Dev

Embedded

$E = 3.6 \times (400)^{1.2} = 4772.81$ man-months

$D = 2.5 \times (4772.81)^{0.32} = 37.6$ months

$\frac{E}{D}$ = 128 Dev

Note :- In the above SW development, the effort req. to develop the SW in the embedded mode is 4 times effort req. in organic mode & 2 times " " " semi-detached mode, but the duration of the 3 modes is approx. equal.

There is a huge difference in the manpower requirement, ∴ to develop the SW on-time & in-budget, there is a need of mode selection i.e. embedded mode.

Q. A company needs to develop DSP SW for one of its newest invention. The S/W is expected to have 40,000 LOC. The company needs to

Cosmos(Saral) @ Technits

determine the effort in person-months needed to develop the s/w using basic cocomo model. The multiplicative factor for this model is 2.8, while exponentiation factor is calculated as 1.2, what is estimated effort in persons-month.

Ans. $E = 2.8 \times (40)^{1.2}$
$= 234.25$ person-months.

Intermediate Cocomo Model:-

- In this model, 15 predictors are introduced as cost drivers to calculate the accurate effort & duration.
- These 15 cost-drivers are grouped into 4 types:-

① Product attributes
② Project "
③ Personal "
④ Computer "

The cost drivers involvement is rated as:-

Low, Very low $\} < 1$
Nominal $= 1$
High, Very High, Extra High $\} > 1$

* The cost drivers ~~are~~ ratings are never zero, because effort adjustment factor is a multiplication factor of 15 cost-drivers.

- The structure of effort & duration estimation is

$E = C1 \times (KLOC \text{ or } FP)^{P1} * EAF$ [$EAF = \prod_{i=1}^{15} (\text{cost drivers})$

$D = C_2 (E)^{P_2}$

Cosmos(Saial) @ Techbits

Q. Consider a project that consists 5 modules with the estimated sizes as:-
4K, 2K, 1K, 2K & 3K.
Overall cost & schedule estimate based on the following cost drivers impact :-
1.15, 1.15, 0.86, 1.07 & rest others are 1.
What is the expected effort when the multiplicative factor 3.2 & exponentiation factor is 1.05.

Ans. EAF = 1.21
$E = 3.2 \times (12)^{1.05} \times 1.21$
$= 52.61$ man-months.

Quality-Estimation

- Quality is defined as confirmance to the explicitly stated requirements.
- Quality is an indirect measurement, it depends on the defects. Defect is an uncovered error after careful evaluation of the project.
- In the quality estimation we can calculate defect density & defect removal efficiency.
- Defect Density (rate) = $\frac{\text{no. of defects} \times \text{time frame}}{\text{opportunities for defects (LOC)}}$
(for example, when product is in opn, it is in opn for 1 hr)
- DRE = $\frac{E}{E+D}$ [Errors :- E (total no. of errors.)
D :- Defects (subset of errors, also called uncovered errors.)
- The quality attributes are divided into diff. types based on the user pt of

Cosmos(Sajan) @ Techbits

View & developer point of view)

User-perspective:-	Developer's perspective
1. Availability	1. Maintainability
2. Efficiency	2. Portability
3. Flexibility	3. Testability
4. Integrity	4. Reusability
5. Interoperability	
6. Reliability	
7. Robustness	
8. Usability	

- In the Software engg., the time spent & effort req. keeping s/w opn. (Operational) after the release is very significant & it consumes 40-70% of cost of the entire life cycle.
- To minimize the maintainence effort, invest more effort in the early stage of the s/w life cycle.
- To calculate the maintainence effort, Bohem introduced an estimation model.
  (i) ACT (Annual Change
  (ii) AME (Annual Maintainence Effort)
- ACT means the no. of source inst^n or that undergoes change during a year through adding the inst^n/ deleting the inst^n/modifying the inst^n.

$$ACT = \frac{KLOC_{added} + KLOC_{deleted}}{KLOC_{Total}}$$

- AME is calculated as :-

$$AME = ACT * SDE$$

SDE :- S/w development effort.

Cosmos Sajan @ Techbits

Q1. ACT for a s/w system is 15% per year. The development effort is 600-man months. If the life-time of the project is 10 years, what is the total effort of the project.

Ans. In maintainence phase:-

ACT = 0.15

AME = 0.15 × 600

= 90 man-months (for 1 year)

for 10 years = 900 man-months.

now, total effort = 600 + 900

= 1500 man-months.

Cosmos(Sajal) @Techbits

Date

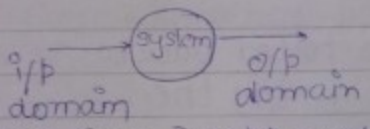
02.12.12

Functional Testing

① In this testing, external behaviour is taken into the account to cover the functional errors

② In this regard, diff test cases are prepared based on the i/p & o/p domain

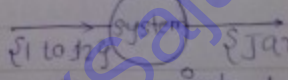
③ The test case development process is



Test Case Development :-

Test Case ID	input	expected o/p
Test-Case 1	i/p	o/p
Test-Case 2	i/p	o/p

e.g.



Test Case ID	input	expected o/p
Test-Case 1	1	Jan
Test-Case 2	6	June
Test-Case 3	0	Invalid i/p
Test-Case 4	13	Invalid i/p

* ① The developed test cases are implemented in the system to cover various functional errors

② The implementation process is

Test Case ID	Test Data	Test Result
• Test Case 1	1 → (sys) → actual o/p Jan	(Actual o/p is compared with expected o/p) Jan CMP Jan Success
• Test Case 2	0 → (sys) → actual o/p June	June CMP Invalid i/p ↳ F-failure → Reported to the development team

Cosmos(Sajal) @ Techbits

- Test Case 4 → 13 → (Sys) → Invalid o/p

invalid i/p CMP
invalid i/p
↳ success

During the test case implementation, the actual o/p of the system is compared with the expected o/p. When both are same, then the test case is successful, otherwise test case is failure.
The failed test cases are reported back to the development team to modify the program.
In the black box testing, diff test plans are used to cover the functional errors:

(i) equivalence partitioning
(ii) boundary value analysis
(iii) robustness test
(iv) comparison test

Equivalence Partitioning :-
In this testing, i/p domain & o/p domain is divided into equal parts known as classes.
The classes are claffied into:

- valid class
- Invalid class wrt lower limit.
- Invalid class wrt upper limit.

Acc to the above example scenario, i/p domain is divided into the following classes:

- class {1 to 12}
- class {<1}
- class {>12}

Cosmos(Sajal) @ Techbits

The various test cases are prepared based on the i/p classes, ie test case 1, {1 to 12}, {Jan to Dec}
test-case 2, {<1} , {invalid i/p}
test-case 3, {>12} , {invalid i/p}

In the test case implementation, test the system with anyone of the i/p of the class, if the test case is successful with the i/p, then no need to test the system with other possible i/p's of the same class.

Imp^n: Test-Case 1 — Test Data: 6 → (sys) → June — Test Result: June CMP June → success

no need to test the system with other i/ps in the range of {1 to 12}

The advantage of equivalence partitioning testing is that no. of test cases will be minimized.

* When the valid i/p box (class) size is large, then to get the efficient test result there is a need of dividing the class into valid subclasses.

Boundary Value Analysis

- S/w test experts indentified that most of the errors occurs at boundary levels rather than center point of the i/p domain. ∴ to cover more errors at boundary levels, boundary value analysis is done
- In this testing, various test cases are prepared as follows:
  Lower limit, Just above the lower limit, center, Just below the upper limit & Upper limit

Cosmos(Sajal) @ Techbits

Suppose that variable x is bounded b/w a & b, i.e.

$a \le x \le b \quad x = [a, b]$,

to test variable x, the following test cases are req:-

$a, a+1, \frac{a+b}{2}, b-1, b$

- Program contain 2 variables x & y, both are accepting i/p in the range of [100, 300]. The various test cases req. to test x & y variables are as follows:-

[x, 200] : To test x, keep y in the centre.
test this variable ← keep all other variables (except x) at the centre

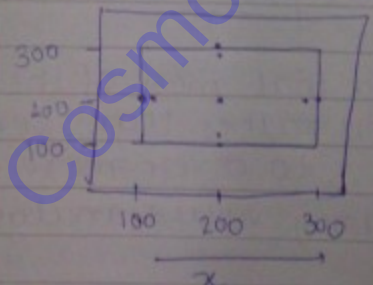
(100,200), (101,200), (200,200), (299,200), (300,200)

[200, y]

(200,100), (200,101), (200,200), (200,299), (200,300)

centre pt test case is common for all the variables.

Symbolic representation



Note: When the program contain n variables then the boundary value analysis is yeilds "4n+1" test cases.

Cosmos(Sajal) @ Techbits

Robustness Testing

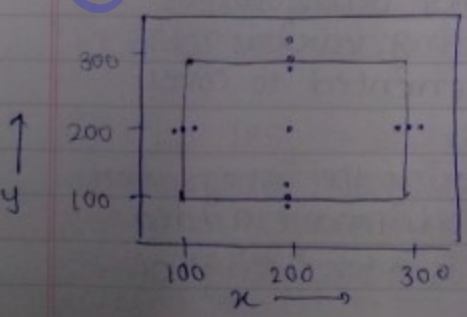
- It is an expansion of boundary value analysis, it is used to conver more errors at the boundaries rather than center point by conducting the different test case implementation.
- Robustness test case development process is:
  Just below the lower limit, lower limit, Just above the lower limit, center, just below the upper limit, upper limit, & just above the upper limit.
- Suppose that variable 'x' is bounded b/w a & b values, i.e. $a \le x \le b$,
  the various test cases req. to test the variable x is $a-1, a, a+1, \frac{a+b}{2}, b-1, b, b+1$

e.g. Prog contain x & y = [100,300]
Test x variable :- [x,200]
(99,200), (100,200), (101,200), (200,200), (299,200), (300,200), (301,200).

Test y variable [200,y]
(200,99), (200,100), (200,101), (200,200), (200,299), (200,300), (200,301)

x not taken Cas already taken while testing

Total Test Cases: 6n+1



Cosmos(Sajan) @ Technirs

* When the prog. contain 'n' variables, then the robustness test yields:

6n+1 test case

Comparison Test

- In some of the applications, accuracy is most imp. factor eg. aircraft simulation.
- In such applications predefined o/p set is maintained based on the i/p component properties & index values.
- After developing the program, the program o/p is always compared with the predefined o/p set to cover the errors called as comparison test.

SDLC Testing:-

- In the s/w development life cycle, various test plans are implemented at diff. levels.
- In SDLC, 3 levels of testing opn is conducted:

(i) Unit test :- Test opn is performed on part of the program. In this test, the module correctness is taken into the account rather than external behaviour of the program. In this testing, various white box test plans are implemented to cover the structural errors.

(ii) Integration test :- During the integration process, when the new modules are added to the existing system, there is a possibility of

Cosmos(Sajal) @ Techbits

new functional errors because functionality changes as the new modules are added), so some kind of the test is required in the intergration process to cover the functional errors
It uses the black box test plans. There are 3 kinds of integration process is present

(i) TOP-DOWN
(ii) BOO BOTTOM-UP
(iii) Sandwich

In top-down, the integration starts from root to leaf in this process drivers are not required but steps are required
In the bottom-up, integration starts from leaf node to root node, in this process steps are not req. but drivers are req.
In sandwich, integration starts from the root node & leaf node simultaneously.

Regression Test - It means retest. It is not a development test It is conducted when the development program is in the operational state.
- During the maintainence phase of the product, if any fraction of code is changed in the module, that introduces additional error in the entire system. So there is a need to retest the entire system to cover the error due to fractional change.

Smoke Test :- It is used in the "wrap-shrink" applications When the developer is developing the S/w acc to the customer's requirement priority list, some of the modules are added & some are deleted to deploy the part of the system to customer location,

Cosmos (Sajad) @ Teenbits

In this process, new functionalities are added so there is possibility of new errors, to cover the functional errors, black box test plans are introduced in this process. known as smoke tests.

(iii) System Test :-
During the system test opⁿ, various attributes of the system is tested

(i) Recovery Test :-
It focuses on the backup functionality, i.e. during the operational state, if any unexpected occurs (system crash, power failure, etc) how to recover the data.

(ii) Security Test :-
It focuses on the system protection, i.e. what kind of the security is provided (password choice, voice recognition, image pattern, etc.) to protect the system from the unauthorised accesses.

(iii) Stress Test :-
It focuses on the load balancing, i.e. how many no. of users can able to use the system at a time

(iv) Performance Test :-
It focuses on the s/w reliability, i.e. how long the system is functioning without failure

or

Validation Test :-

- In s/w engg. practices, customer evaluation is also an imp. factor, so customer conducts the test opⁿ to cover the errors by applying strange i/ps.

Cosmos(Gaja) @ Techbits

- Under this case, two kinds of test opn are performed, α & β test,
- α-test is conducted by the customer at the developer site(functional testing). If any errors are identified during the α-test, then the program is immediately modified before deployment because developer is present..
- β-test is conducted by the end user/customer at the customer's location, i.e. β-test is conducted after the deployment. If any errors are identified during β-test, those are reported to the maintainence team.

Support :- When the developed product is deployed at the customer's location, support stage is req. to maintain the sw from the uncovered error, platform changes, functional requirement changes, frequent requirement changes.

Cosmos(Sarali)@ Techbits