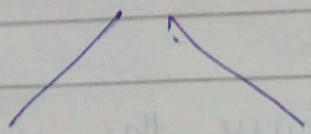
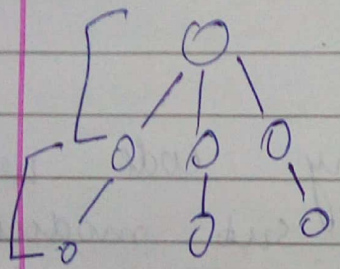
Integration Testing

Integration testing is a systematic technique for constructing the software architecture the objective is to take unit testing components and build a program structure that has been dictated by design.



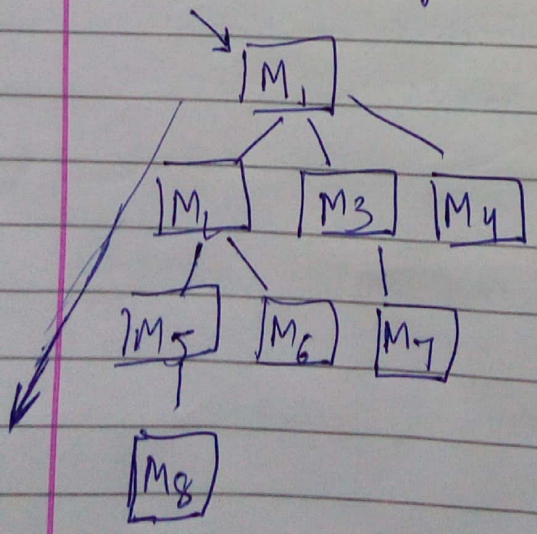
. Top down integration

bottom up integration



depth first

breadth first



M1, M2, M3, M4

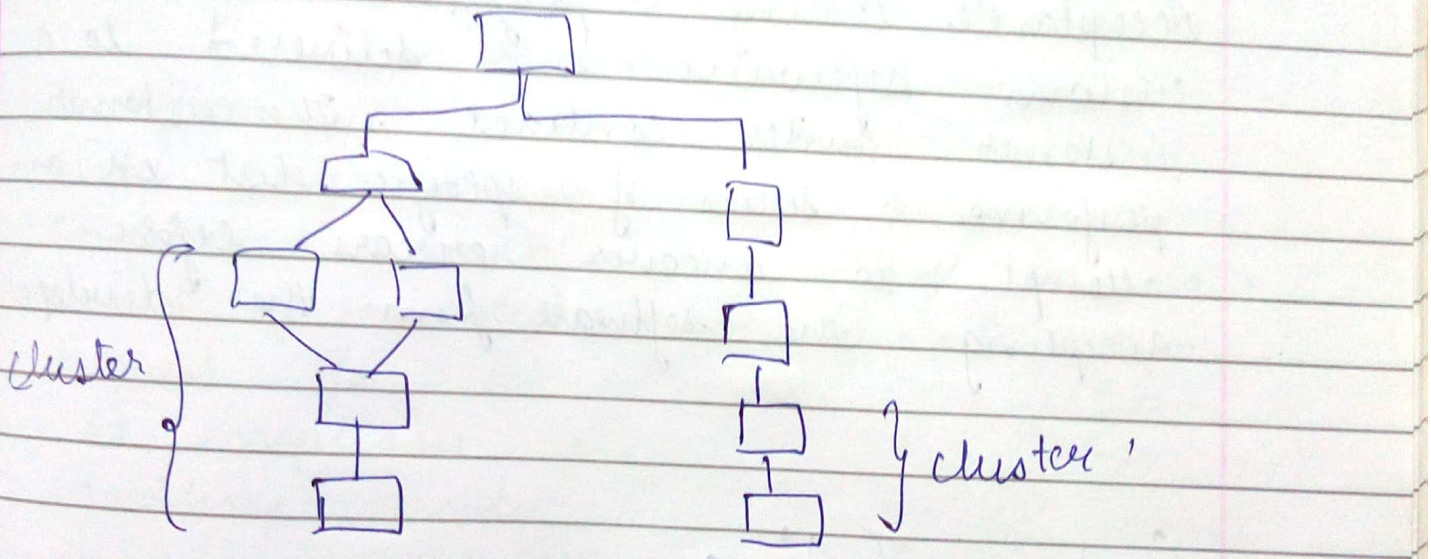
M1, M2, M5, M6, M8

[top down] (5 steps :->

Regression testing.

Regression testing is the reexecution of some subset of Test that has already been conducted to ensure that changes have not propergated unintended side effects.

Bottom up integration :-



Smoke testing

It is an integration testing approach that is commonly used when product software is developed. in another words smoke testing refers to testing the basic functionality of the build. a build includes all data files libraries, reusable modules and engineered components that are required to implement one or more build.

(3 activities)

Date :
Page No. :

Validation Testing

- Acceptance testing (anybody can test)
- Alpha testing (highly qualified person of company)
- Beta testing (customer level)

Acceptance testing is performed when custom software is delivered to a customer under contract. The customer performs a series of specific test in an attempt to uncover errors before accepting the software from the developer

System Testing

- Recovery testing
- Security testing
- stress testing
- Performance testing.

Recovery testing :- It is a system testing that forces the software to fail in a variety of ways and verify that recovery is properly performed.

Date :
Page No. :

Security testing :- Any computer based system that manages sensitive information or causes actions that can improperly harm individual either target for improper or illegal penetration.

→ It attempts to verify that protection mechanism build into a system will protect it from improper penetration.

Stress testing :- It refers to the testing of software to determine whether its ~~software~~ performance is satisfactory under any extreme and unfavourable condition which may occur as a result of process loading, maximum request for ~~an~~ resource utilization.

→ It emphasis availability and error handling under extreme heavy loads to ensure software doesn't crash due to insufficient resources.

Performance Testing :- It is designed to test the run time performance of software within the context of an integrated system it occurs throughout all steps in the testing process.

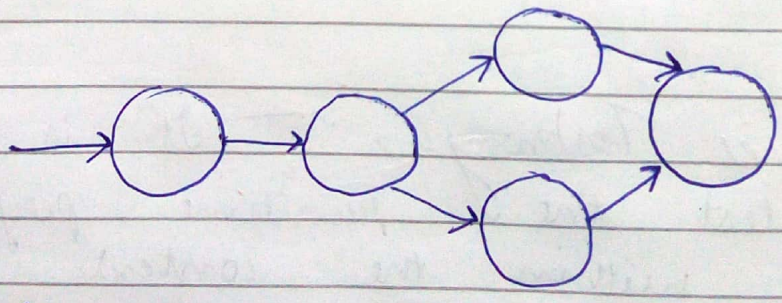
White Box Testing

To test the internal structure
It is the method of testing software that test internal structure or working of an application

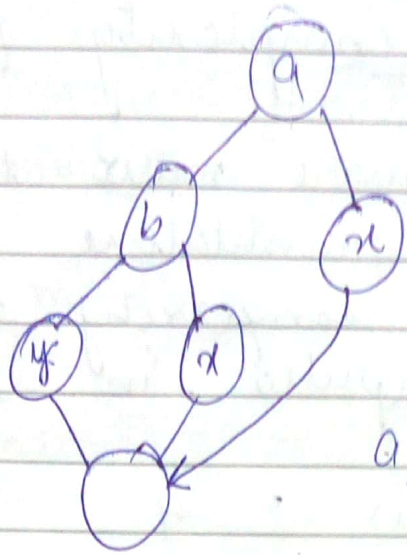
1) Basis Path Testing
↓
independent path

- It is the white box testing technique
- Test cases derived to exercise the basis set are garuented to execute every statement in the program atleast one time during testing.

→ Flow Graph Notation
It depicts logical control flow



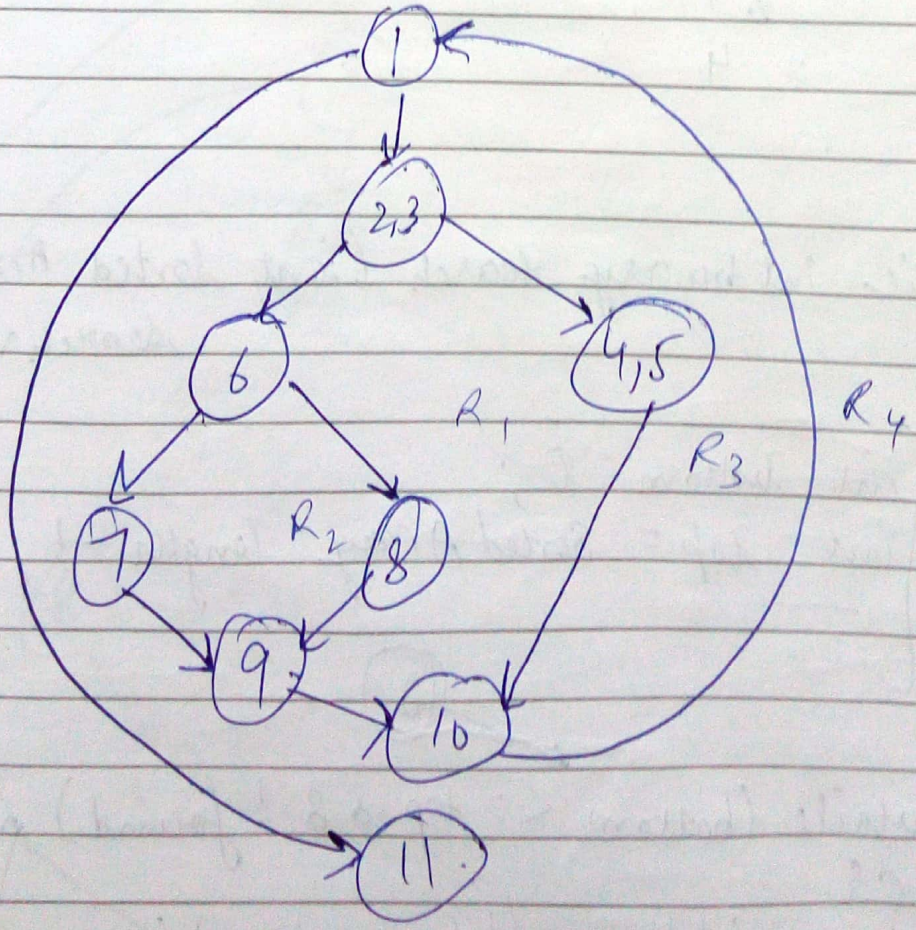
Predicate node
Each node that contains a condition is called Predicate node



a, b are Predicate node

Independent Path

In terms of a flow graph an independent path must move along atleast one edge that has not been traversed before the path is defined.



Control Structure Testing

It is a group of white box testing methods

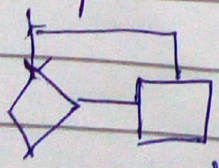
(1) Branch testing
(2) Condition testing
(3) data flow testing
(4) loop testing

Condition testing

- It is a test case design method that exercises the logical conditions in a program module
- Condition testing method focus on each condition in the program to ensure that it doesnot contain error
- Error in conditions can be due to boolean operator error, boolean variable error, boolean parantheses error, relational operator error, aisthmatic exp error

loop testing

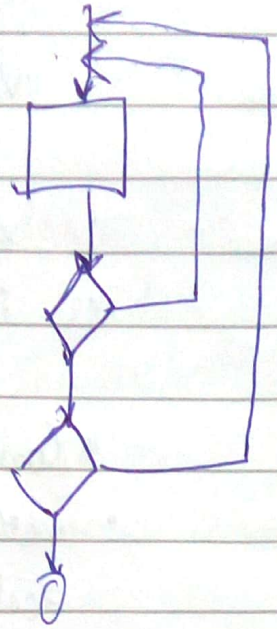
- It is a white box testing technique that focus exclusively on the validity of loop construct
- Simple loop



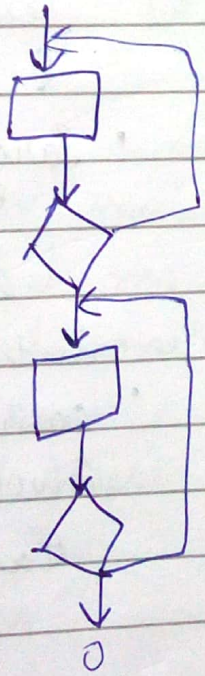
for (i = 1 ; i < 10 ; i++)

Date :
Page No. :

Nested loop.



Concatenated loop



Unstructured loop.

Page No.:

Diff b/w Validation & Verification.

	Verification	Validation
1.	* Static	• Dynamic
2.	doesnot involve executing code.	• involves
3.	human based checking	• Computer based
4.	methods like inspection, reviews etc.	• methods like black box testing, white box testing Grey box testing.
5.	It is to check whether the software conforms to specification	• To s/w meets the customer expectation & requirements
6.	Target is requirement specification, application & s/w architecture.	• Target is actual product - a unit, a module, integrated modules, final product
7	It is done by quality assurance team.	• carried out with the involvement of testing team.

⇒ Management Spectrum

4 main P's

People
Product
Process
Project

Effective Project managment focus on the four P's.

1. People
   - stakeholders
   - Team leader
   - S/w Team
   - Agile Team
   - Coordination and communication issues.

• Stakeholder

The software process is populated by stakeholder who can be categorised into one of 5 constituencies

↓

⇒ senior manager
→ Project manager
→ Practitioner
→ Customer
→ End users.

• Team leader

Jerry Weinberg suggests a MOI model of leadership

Motivation — Organisation — Ideas

Problem solving
Management Identity.
Achievement.

- Software Team

Constantine Suggests 4 organizational paradigm for s/w engg Teams:-

1) closed Paradigm
2) Random paradigm
3) open Paradigm
4) Asynchronous paradigm

- A closed paradise structures a along a traditional hierarchy of authority. Such team can work well, when producing s/w that is quite similar to pass efforts but they will be less likely to be innovative when within a closed paradigm

- Random paradigm structure a team loosely and depends on individual initiative of team members when innovation or technological is break-through is required, teams following the random paradigm will excel.

- open paradigm attempts to structure a team in a manner that achieves some of the control associated with the closed paradigm but also much of the innovation that occurs when using random paradigm.

Date :
Page No. :

- Synchronous Paradigm relies on the natural compartmentalisation of the Problem & organises team members to work on PC's of the Prob with little active communication among themselves.

formal communication → meeting, structured meetings.
informal → members of software team share ideas on adopt basis.

Product

1) s/w scope
The first s/w Project management activity is the determination of s/w scope
It is defined by answering the following ques.

→ Contacts
How does the s/w to be build fit into a larger system and what constraints are imposed as the result of the contacts.

→ Information objectives
what customer visible data objects are produced as output from the software ?
what data objects are required for input?

→ function & Performance
what functions does the s/w performs to transform input data into output.

- Problem decomposition
It is called partitioning or problem elaboration.
It is an activity that fits at the core of the software requirement analysis. During the scoping activity no attempt is made to fully decompose the problem. rather decomposition is applied in 2 major areas. 1) The functionality that must be delivered 2) The process that will be used to deliver it.

⇒ Process

1) Melding the Product & Process

{ diagram from book }

Project Planning begins with the melding of the product and the process.
Each function to be engineered by the Software team must pass through the set of framework activity that have been defined for a software organisation.
Assume that the organisation has adopted the following set of framework activities.

- Communication
- Planning
- Modeling
- Construction
- Deployment

The team members who work on a product function will apply each of the framework activities to it.

2) Process Decomposition.

Process decomposition begins when the project manager ask how do we accomplish this framework activity?
for eg. a small simple project might require the following work task for the communication activity
- ↳ develop list of clarification issues.
- ↳ Need with customer to address clarification issue
- ↳ Jointly develop a statement of scope
- ↳ Reviews the statement of scope with all concern
- ↳ Modify the statement of scope as required

⇒ Project

Reel suggests 5 part common sense approach to software project.
- ↳ Start on the right foot.
- ↳ Maintain momentum
- ↳ Track the progress
- ↳ Make smart decisions
- ↳ Conduct a Postmortem Analysis.

⇒ W^5HH Principle

why — How will
what — How much
when
who
where

⇒ Size Estimation.

Two ways.

① Line of code (LOC)
② Function Point.

Size estimation is used to estimate the size of the software application. It helps the project manager to further predict the effort and time which will be needed to build the project.

Following are the measures used in Project Size ~~est~~ Estimation

① line of code (LOC)

As the name suggests LOC count the total number of lines of source code in the project

```
1    int sort(int x[], int n)
2    {
3        int i,j, save, im1;
4    /* This function sorts array x in ascending order */
5        if (n<2) return 1;
```

```
6   for(i=2, i<=n; i++)
7   {
8       ind = i-1;
9   for(j=1; j<=ind; j++)
10      if(x[i] < x[j])
11        {
12          Save = x[i];
13          x[i] = x[j];
14          x[j] = Save;
15      }
16    }
17      return 0;
18.   }
```

LOC = 18
LOC = 17 (without comments & blank spaces)
LOC = 13 (executable code)

A line of code is any line of program text that is not a comment or blank line, regardless of the no of statements on the line.
This specifically includes all ~~lines~~ lines containing program headers, declaration and executable and non executable statements.

Advantage

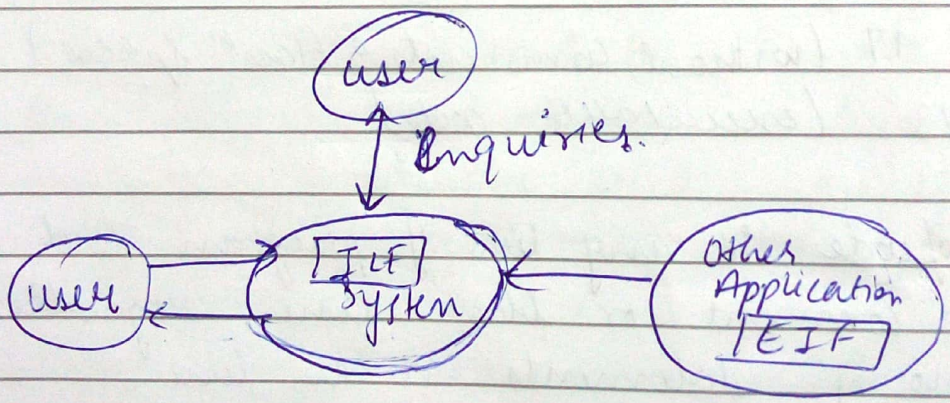
1. Simple to use
2. Universally accepted and is used in many models like COCOMO.

Disadvantage

1. Different programing languages contain different no. of lines.
2. It is difficult to estimate the size using this technique in early stages of project.

(2) Function Point

It measures functionality from the user point of view i.e on the basis of what the user request and receive in return therefor it deals with the functionality being delivered and not with the LOC, source modules, files etc.



ILF → Internal logical files
EIF → External Interface files.

Principle of Albrechet's Function Point Analysis

It is that a system is decomposed into functional units

1. External Input
   functions related to data entering the system

2. External output
   functions related to data exiting the system.
3. External enquiries
   They leads to data retrieval from system but donot change the system.
4. Internal logical files
   logical files maintained within the system.
5. External interface files
   These are logical files for other app which are used by our system.

These 5 functional units are divided into 2 categories
1. Data function type.
   (i) Internal logical files
   (ii) External interface files
2. Transactional function type
   (i) External input
   (ii) External output
   (iii) enquiry.

⇒ Counting function point
The 5 functional units are ranked acc to their complexity i.e low, average or high

Functional Units	Weighting factors.		
	Low	Average	High
External I/P	3	4	6
External o/p	4	5	7
External Inquery	3	4	6
Internal logical files	7	10	15
External interface files	5	7	10

$$FP = UFP * CAF$$

- FP → Function Point
- UFP → unadjusted function point
- CAF → Complexity adjustment factor

UFP

functional unit	Count	Complexity	Complexity Total	Functional unit total
External i/p	20	low × 3	= 60	
	20	Avg × 4	= 80	
	20	High × 6	= 120	Add = 260
External o/p		low ×	=	
		Avg ×	=	
		high ×	=	

UFP = → +

$$UFP = \sum_{i=1} \sum_{j=1} z_{ij} w_{ij}$$

CAF

factor from book (14)

0 — 1 — 2 — 3 — 4 — 5

0: no influence; 1: Incidental; 2: Moderate; 3: Avg; 4: Significant; 5: Essential

multiply 0-5 range with 14

$$CAF = [0.65 + 0.01 \times \Sigma F_i]$$

Q Consider a project with the following functional units

(i) number of user input = 50
number of user enquiry = 35
logical user files = 6
user output = 40
External inputs = 4

Assume all complexity adjustment factors and weighting factors are average. Compute the function point for the project.

→ UFP

50 x 4	=	200
35 x 4	=	140
6 x 10	=	60
40 x 5	=	200
4 x 7	=	28
		628 = UFP

CAF.

$0.65 + 0.01 \times 3 \times 14$
$= 1.07$

FP = UFP * CAF
$= 628 * 1.07$
$= 671.96$

Ques An application has following
10 low external I/P, 12 high external o/p,
20 low internal logical file, 15 high
external interface files, 12 average external
Inquiries & a value of CAF of 1.10.
What are unadjusted & adjusted function
point count?

→ UFP

10 x 3	=	30
12 x 7	=	84
15 x 10	=	150
20 x 7	=	140
12 x 4	=	48
		452

497.2

CAF

1.10.

FP = 452 x 1.10
= 497.2

Ques Consider a project with the following parameters
External i/p. → (a) 10 with low complexity
(b) 15 with Avg complexity
(c) 17 with high complexity.
External o/p → (a) 6 with low complexity
(b) 13 with high complexity
External Inquiry → (a) 3 with low
(b) 4 with Avg
(c) 2 with high

Internal logical files → (a) 2 with Avg
(b) 1 with high
External interface → (a) 9 with low

In addition to above System requires
(i) significant data communication
(ii) Performance is very critical
(iii) Design code may be modularly reusable
(iv) system is not designed for multiple installation in diff organisations

Other complexity adjustment factor (CAF) are treated as Avg.

Compute the FP for the project.

(424).

UFP

1.	10 × 3	= 30		
	15 × 4	= 60	= 192	
	17 × 6	= 102		
2.	6 × 4	= 24	= 102	
	13 × 6	= 78		
3-	3 × 3	= 9	= 37	
	4 × 4	= 16		
4.	2 × 6	= 12		
4.	2 × 10	= 20	= 35	
	1 × 15	= 15		
5.	9 × 5	= 45	= 45	
			424.	

CAF

$\Sigma F_i = 4+5+2+0+(3\times10) = 41$

$CAF = 0.65 + 0.01 \times 41$

$= 1.06$

$FP = 424 * 1.06 \Rightarrow 449.44$

⇒ Cost Estimation

COCOMO Model

↓

(Constructive Cost Model)

1. Basic Model

Cocomo is a heirarchy of software cost estimation model which include basic, intermediate, detailed model

1. Basic Model

The basic model is aims at estimating, in a quick and rough fashion, most of the small to medium sized software projects three models of software development are considered in this model.

(a) Organic
(b) Semi detached
(c) Embedded.

Mode	Project size	Nature of Project	Innovation	Deadline	Development
• Organic	2-50 K LOC	Small Size Project eg. lib manag. system	little	not tight	Familiar & inhouse
• Semi-detached	50-300 K LOC	Medium sized eg. Database system	medium	medium	medium
• Embedded	300 K LOC & above	Large sized eg. ATM, Air traffic control	Significant	tight	Complex h/w interface

$E = a_b (KLOC)^{b_b}$ [Efforts applied in Person months]

$D = c_b (E)^{d_b}$ [Development time in months]

Product	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Average staff size $= \frac{E}{D}$ Persons.

Productivity(P) $= \frac{KLOC}{E}$ KLOC/PM

Ques Suppose that a project was estimated to be 400 KLOC. Calculate the effort and development time for each of the 3 modes that is organic, Semi-detached and Embedded.

Organic $\Rightarrow$ $E = 2.4(400)^{1.05} \Rightarrow 1295.311$

$D = 2.5(\quad)^{0.38} \Rightarrow 38.075$

Semi $= E = 3.0(400)^{1.12} \quad 2462.79$

$D = 2.5(\quad)^{0.35} \quad 88.45$

Embedded $E = 3.6(400)^{1.20} \quad 4772.81$

$D = 2.5(\quad)^{0.32} \quad 38.7$

Ques A project size of 200 KLOC is to be developed. Software development team has average experience on similar type of projects. The project schedule is not very tight. Calculate the effort development time, average staff size and productivity of the project.

→ Semi-detached.

$$E = 3.0(200)^{1.12}$$
$$= 1133.117 \text{ PM}$$

$$D = 2.5(E)^{0.35}$$
$$= 29.30 \text{ PM}$$

$$\text{Staff} = \frac{E}{D} = \frac{1133.117}{29.30} = 38.672 \text{ Person}$$

$$P = \frac{200}{E} = \frac{200}{1133.117} = 0.176 \text{ KLOC/PM}$$
$$= 176 \text{ LOC/PM}$$

Intermediate Model

The basic model allowed for a quick and rough estimate but it resulted in a lack of accuracy Boehm introduced an additional set of cost drivers (Predictors) in a intermediate model to take account of the software development environment. Cost drivers are used to adjust the nominal cost of the project to the actual project environment, hence increasing the accuracy of the estimate.

The cost drivers are grouped into four categories:

1) Product attributes
2) Computer attributes
3) Personnel attributes
4) Project attributes

- Analys. + Capability (ACAP)
- Application experience (EXP)
- Programmer capability (PCAP)
- Virtual Machine experience (VEXP)
- Programing language experience (LEXP)

Cost Drivers	Ratings					
	very low	low	Nominal	High	very high	Extra high

Earned Value Analysis :-

It is an industry standard method of measuring a project's progress at any given point in time forcasting its completion date and final cost and analysing variencies in the schedule and budget as the project proceeds.

1) BAC = Budget at completion

2) Planned (PV) or Budget Cost work schedule) = Planned value
Planned completion (%) * BAC

3) Earned value (EV) or Budget cost work performed (BCWP) = Actual completion (%) * BAC

4 SPI (Schedule Performance Index) = $\frac{BCWP}{BCWS}$

5 Schedule Variance (SV) = BCWP - BCWS

6 Cost Performance Index = $\frac{BCWP}{ACWP}$

7. Cost variencee (CV) = BCWP - ACWP

Actual Cost or Actual cost Work Performance (ACWP)

Ques Assume you are a software project manager and you have been asked to compute earned statistics for a small software project. The project has 56 planned work task that are estimated to require 582 person days to complete at the time you have been asked to do the earned value analysis 12 tasks have been completed however the project schedule indicates that 15 tasks should have been completed The following scheduling data in person days are available

Compute the SPI, schedule variance, % schedule for completion, ~~person~~ % Complete, CPI and cost variance.

Task	Planned effort	Actual effort
1	12.0	12.5
2	15.0	11.0
3	13.0	17.0
4	8.0	9.5
5	10.0	19.0
6	4.0	19.0
7	12.0	14.0
8	6.0	~~6.5~~ 10.5
9	5.0	~~4.0~~ 6.5
10	14.0	~~14.5~~ 4.0
11	16.0	- 14.5
12	6.0	-
13	8.0	-
14		-
15		-
+	BCWS	

BAC = 582 Person days.

Earned value = Add first 12 Planned effort (127.5)

Actual cost = Add Actual effort. (127.5)

Planned value = Add Planned effort (156.5)

$$SPI = \frac{BCWP}{BCWS} = \frac{\text{Earned value}}{\text{Planned value.}}$$

$$= \frac{127.5}{156.5} = 80.8\ \%$$

$$SV = BCWP - BCWS$$

$$= -30.$$

-ve means project is overbudget

CPI = 99.2 %

CV = -1

Q 1

Numerical of Intermediate Model

Date :
Page No. :

Ques A new project with estimated 400 KLOC embedded system has to be. Project manager has a choice of hiring from two pools of developers

(i) very highly capable with very little experience in the programming language being used

(ii) developers of low quality but a lot of experience with the programming language.

What is the impact of hiring all developers from one or the other pool

→ 1st → Effort ↓
2nd → Effort ↑

(i) EAF (Effort adjustment factor) = 0.82 [AEXP → Very high] × 1.14 [LEXP → Very Low]

EAF =

$E = a_i (KLOC)^{b_i} \times EAF$ a_i : Embedded table

=

$D = c_i (E)^{d_i}$

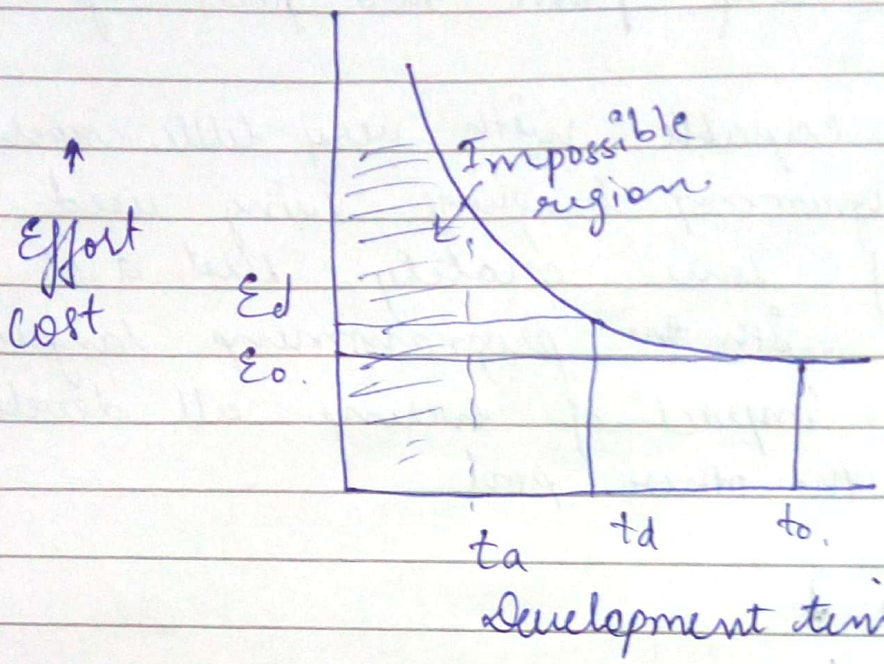
(ii). EAF = 1.29 × 0.25 =

E = ?

D = ?

⇒ Detailed Model

⇒ PNR Curve (Putman Norsden Rayleigh)



$E_d = m(E_d^4 / E_a^4)$
E_d = Effort in PM
t_d = Nominal delivery
t_o = optimal development
t_a = actual time development

Earned value

Ques Suppose you are managing a software development project. The project is expected to be completed in eight months at a cost of $10000/- per month. After 2 month you realise that the project is 30% completed at a cost of $40,000. You need to determine weather the project is on time and on budget after 2 months.

Date :
Page No. :

1 month = $10,000

8 months = $80,000 = budget at completion

Actual cost = $40,000

Planned completion = $\frac{2}{8}$ = 25 %.

Actual Completion = 30%.

PV = Planned Completion * BAC

= 25% * 80,000.

PV = $20,000.

Earned value = 30 % * 80,000.

EV = $24,000

Budget = CPI = $\frac{BCWP}{ACWP} = \frac{EV}{AC} = \frac{24000}{40000}$.

CPI = 0.6

SPI = $\frac{EV}{PV} = \frac{24000}{20000} = 1.2$

CPI < 1 → Project is over budget

SPI > 1 ⇒ ahead of schedule.

Page No. :

Q) Suppose you have a budgeted cost of a project at $9 lakh. This project is completed in 9 months After a month You have completed 10% of the project at a total expense of $1 lakh. The planned completion have been 15%.

→

BAC = $9,00,000

AC = $1,00,000

PC = 15%

PV = PC * BAC

= 15% * 9,00,000

PV = 135000

EV = 10% * BAC

= 10% x 9,00,000

EV = 90,000.

$$CPI = \frac{EV}{AC} = \frac{90000}{100000} = 0.9 < 1$$

$$SPI = \frac{EV}{PV} = \frac{90000}{135000} = 0.67 < 1$$

Date :
Page No. :

Categories of Matrix

1) Product Matrix.
it describes the characteristics of the product such as size, complexity, design, features, performance, efficiency, reliability, portability, etc

2) Process Matric :- Describe the effectiveness & quality of the processes that produce the s/w product. Eg - effort reqd in the process, time to produce the product, effectiveness of defect removal during development no of defects found during testing, maturity of the process.

3) Project Matric :- It describes the project characteristics and execution. Eg. No. of s/w developer, staffing pattern over the life cycle of the s/w. Cost & Schedule, productivity

Notes

Notes

(1) Product Matric - It describes the characteristics of product such as size, complexity, design, features, performance, efficiency, reliability, portability etc.

(2) process ~~project~~ matric describes the effectiveness that produce the s/w product.
eg: 1) effort reqd. in the process,
2) time to reduce the process,
3) effectiveness of defect removal during development,
4) No. of defects found during Testing.
5) Maturity of process.

(3) project matric describes the project and characteristics and execution. eg:
1) No. of s/w developer.
2) Staffing pattern over the lifecycle of the s/w.
3) Cost and schedules.
4) Productivity.

May 2018
मई

01 | Tuesday मंगल

02 | Wednesday बुध

Equivalence class Partitioning :

Q. Quadratic equation (say a, b, c) & value from interval [0, 100]. The program o/p may be have [Note a quadratic equation, Real roots, Imaginary roots, qual roots. Disign the equivalence class and boundary value analysis.

$ax^2+bx+c=0$

→ Roots are real → $b^2-4ac>0$
o/p → imaginary $b^2-4ac<0$
equal $b^2-4ac=0$
not quadratic → $a=0$

$O_1 = \{a, <b, c>\}$ Not a quadratic equation if $a=0$,
$O_2 = \{b^2-4ac) > 0$
$O_3 = \{b^2-4ac) < 0$
$O_4 = (b^2-4ac) = 0$

Testcase	a	b	c	expected o/p
0	0	50	50	Not a Quadratic
1	1	50	50	Real Roots
2	50	50	50	Imaginary roots
3	50	100	100	Equal roots

April'18

S	M	T	W	T	F	S
01	02	03	04	05	06	07
08	09	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

May'18

S	M	T	W	T	F	S
		01	02	03	04	05
06	07	08	09	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

June'18

S	M	T	W	T	F	S
					01	02
03	04	05	06	07	08	09
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Scanned by CamScanner

03 | Thursday गुरु

I/P : a, b, c

$a < 0$

$1 \le a \le 100$

$a > 100$

I_1 $\{a : a = 0\}$

I_2 $\{a : a < 0\}$

I_3 $\{a : 1 \le a \le 100\}$

I_4 $\{a : a > 100\}$

I_5 $\{b : 0 \le b < 100\}$

I_6 $\{b : b < 0\}$

I_7 $\{b : b \ge 100\}$

I_8 $\{c : 0 \le c \le 100\}$

I_9 $\{c : c < 0\}$

I_{10} $\{c : c > 100\}$

04 | Friday शुक्र

Testcase	a	b	c
1	0	50	50
2	-1	50	50
3	50	50	50
4	101	50	50
5	50	50	50
6	50	-1	50
7	50	101	50
8	50	50	50
9	50	50	-1
10	50	50	101

May 2018
मई

05 | Saturday शनि

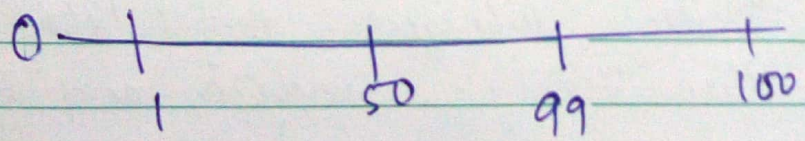
07 | Monday सोम

Boundary Value Analysis

Test	a	b	c	Expected O/P
1	0	50	50	Not Quadratic
2	1	50	50	Real Roots
3	50	50	50	Imaginary Roots
4	99	50	50	"
5	100	50	50	"
6	50	0	50	
7	50	1	50	
8	50	99	50	

06 | Sunday रवि

9	50	100	50
10	50	50	0
11	50	50	1
12	50	50	99
13	50	50	100



July'18 | August'18 | September'18

मई

08 | Tuesday मंगल

09 | Wednesday बुध

	Planning & req.	System design	detailed design
organic small	0.06	0.16	0.26
Z_I	0.10	0.19	0.24

$$E_p = \mu_p E$$
$$D_p = Z_p D$$

Object Oriented Matric:

It suggests the following set of matric for object oriented projects.

1) No. of seniors.
2) No. of key classes (~~derived~~ parent class)
3) No of support classes (derived class)
4) Average no of support classes per key class.
5) No. of subsystems.

Matric for Software quality :-

(1) Correctness (we find defect)
error are identified by coder itself but defect are identified by customer

→ The most common measure for correctness is defect.

→ defects are those problems reported by a user of the program after the program has been released for general use.

(2) Maintainability :-
It is the ease with which a program can be corrected :-

If an error is encountered.
adapted if its environment changes
enhanced if the customer desires a change in requirement.

programs that are maintainable will have a lower mean time to change than program that are not maintable.

Date :
Page No. :

(3) Integrity :

Integrity $= \Sigma [1 - \text{threat} \times (1 - \text{security})]$

threat $= 0.25$

security $= 0.95$

Integrity $= \Sigma [1 - 0.25(1 - 0.95]$

$= 1 - 0.25 (0.05)$

$= 0.9875$

$= 0.99$

(4) Usability - code should be usable

Defect removal Efficiency :- used to remove defects. and it should be more so the defects can be removed.

→ Effort & defect is given

$DRE = E / (E + D)$

$DRE\ i = E_i / (E_i + E_{i+1})$

↳ If $E + D$ is less then DRE is more (inversely prop)

Monte Carlo Simulation

It is a computerised mathematical technique that allows people to account for risk in Quantitive analysis and decision making. monte carlo technique involves simulating the task no of times (using an appropriate distribution) and then calculating respective completion time based on the result.

tmt = most likely time.

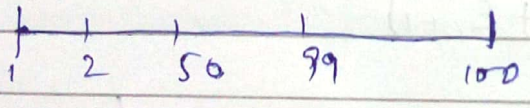
$$P(t) = \frac{1 - (t_{max} - t)^2}{(t_{max} - t_{ml})(t_{max} - t_{min})^2}$$

$$t_{ml} \leq t \leq t_{max}.$$

Ques We have 3 ~~techni~~ triangles: isoceles, scalen, equi. find boundary value analysis and Equivalence Partitioning.

→ Boundary value = $4n+1$ (a, b, c)
$= 4(3)+1 = 13$ (13 test cases will be made)

→ Range $0 - 100$ (x, y, z)
[scalene, isoceles, equilateral, Not a triangle]



Test Case	x	y	z	O/P.
1	1	50	50	I
2	2	50	50	I
3	50	50	50	E
4	99	50	50	I
5	100	50	50	I
6	50	1	50	I
7	50	2	50	I
8	50	99	50	I
9	50	100	50	I
10	50	50	1	I
11	50	50	2	I
12	50	50	99	I
13	50	50	100	I