Classification of Software

1. Organic (appn)
2. Semi detached (utility)
3. Embedded (system)

Parameters -

1. Information determinancy.
2. Information content

* **Information determinancy** — concerned with the time at which I/p enters & the time at which o/p is obtained.

* Information Content — Concerned with Size(P)

1K = KLOC

$P < 2K$ — Small project

$2K < P < 8K$ — Medium project

$8K < P < 32K$ — Large project

$P > 32K$ — Very large projects

* Organic Mode (OM)

HLL - High level language

- Requirement is simple.
- HLL skills are required.
- Duration: a month to few years.
- Team size: 1-50.
- Eg:- Simple inventory system
  Business s/w.
  Scientific s/w.
  Small compilers
  os for laptops / desktops, etc.

* Semi-Detached Mode (SDM)

- Requirement is simple & composite.
- Mixed mode skills. (HLL & LOW level skills)
- Duration: months to years.
- Team size: 2-100
- Eg:- Small transaction processing system
  OS for medium level system (minisystems)
  simple command & control s/w.

* Embedded Mode (EM)

- Composite. Requirement.
- HLL skills.
- Duration in years.
- Team Size : 100 - 1000
- Eg:- Very large transaction processing system
  complex command & control s/w.
  OS for mainframes & supercomputers.

Problems with Vista OS.

Performance of Vista wrt size

DM – 12GB

SDM – 100GB

EM – 400GB

Performance of Windows 7

DM – 4GB

SDM – 60GB.

EM – 100GB

EMPIRICAL MODELS.

* COCOMO – Constructive Cost Model.
  - Basic COCOMO
  - Intermediate cocomo
  - Advanced cocomo

* Basic Cocomo — basic idea of the project.

— Estimation is done based on the size of project (KLOC).

1. Effort, $E = a_b (KLOC)^{b_b}$ person-month

2. Duration, $D = c_b (E)^{d_b}$ months.

3. Number of people, $N = E/D$ persons.

Barry Boehm

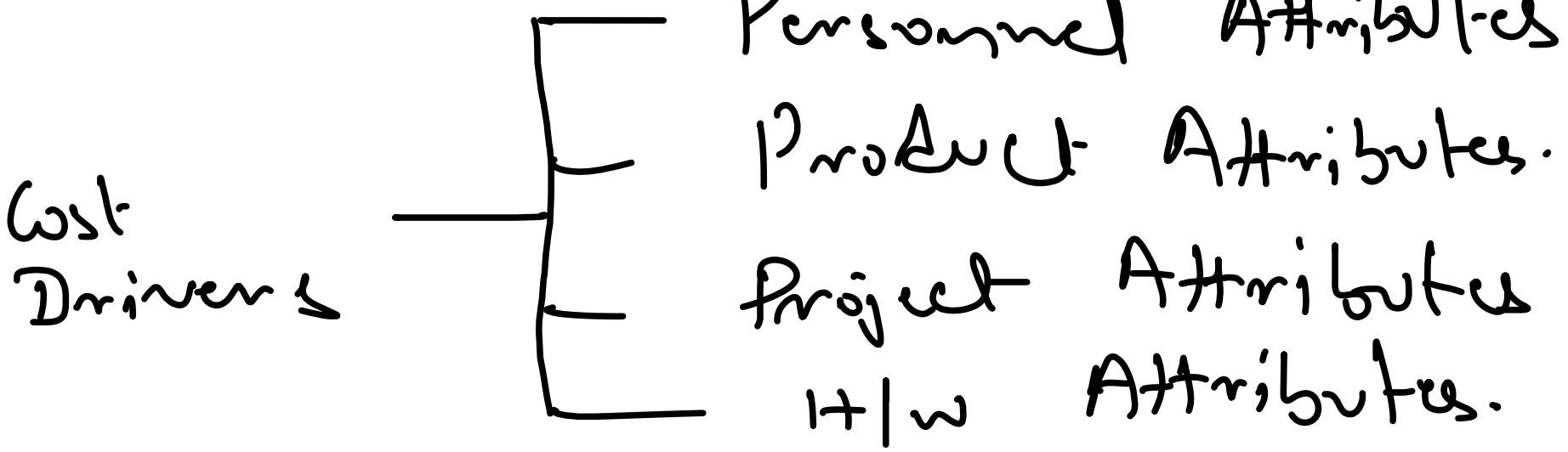
b - basic

	a_b	b_b	c_b	d_b
OM	2.4	1.05	2.5	0.38
SDM	3.0	1.12	2.5	0.35
EM	3.6	1.20	2.5	0.32

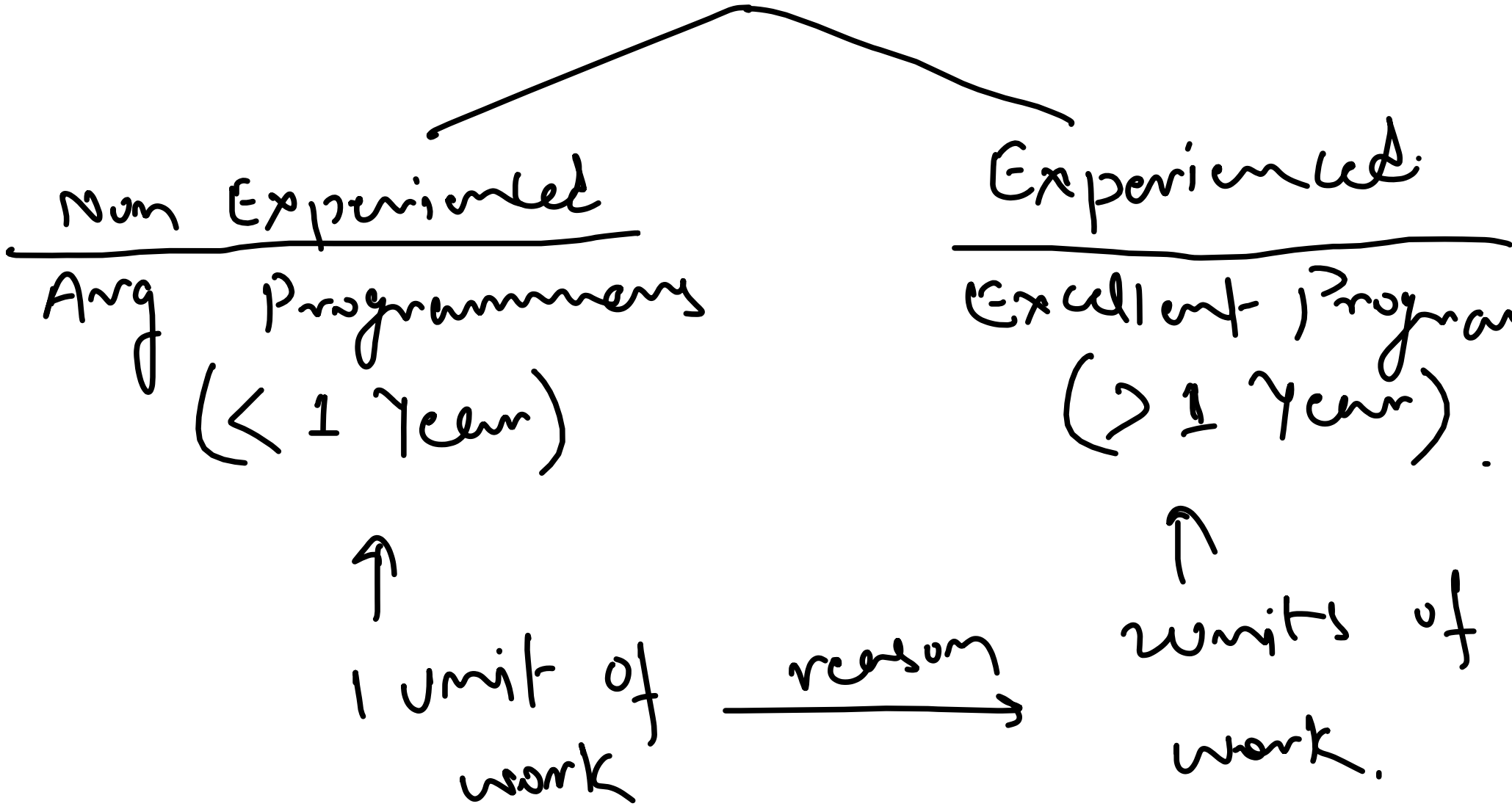
Basic Columns

* Intermediate COCOMO — Actual estimation.

- Along with size, cost drivers are essential in a project.



Personnel Attributes.



$$E = a_i * (KLOC)^{b_i} \times EAF$$

i — intermediate

EAF — Effort Adjustment factor

	a_b	b_b	c_b	d_b	a_i	b_i
OM	2.4	1.05	2.5	0.38	3.2	1.05
SDM	3.0	1.12	2.5	0.35	3.0	1.12
EM	3.6	1.20	2.5	0.32	2.8	1.20

Intermediate COCOMO

* LOC Oriented Estimation Models.

1. $E = 5.2 \times (KLOC)^{0.91}$ (Watson Felix Model)
2. $E = 5.5 + 0.73 (KLOC)^{1.16}$ (Bailey Basic Model)
3. $E = 3.2 \times (KLOC)^{1.05}$ (Simple Boehm Model)
4. $E = 5.288 \times (KLOC)^{1.047}$ (Doty Model, KLOC > 9K)

* fp Oriented Estimation Models.

1. $E = -13.39 + 0.0545 \times fp$

(Albrecht & Gaffney Model)

2. $E = 60.62 \times 7.728 \times 10^{-8} \times fp^3$

(Kemerer model)

3. $E = 585.7 + 15.12 * fp$

(Matson, Barnett, Mellichamp Model)

* Software Eqn.

$$E = \left[LOC \times \frac{B}{P}^{0.333} \right]^3 \times \left(\frac{1}{t^4} \right)$$

B — Special skill factor

P — Productivity.

t — Duration.

E — Effort.

LOC — Lines of code.

* NOTE :-

1. Size of s/w ∝ Complexity.
2. Complexity — design, testing, etc.

Size	B
5-15K	0.16
>70K	0.39

Appn	Productivity
① Real time System	2000
② System s/w, Telecommunication s/w	10000
③ Scientific s/w	12000
④ Business s/w	27000

*. Putnam & Meyer Model.

$$t_{min} = 8.14 \left(\frac{LOC}{P}\right)^{0.43} \text{ months.}$$

↳ min time required to develop a s/w.

$t_{min} > 6$ months generally.

Effort, $E = 180 Bt^3$ person-years.

where, t — years, $E \geq 20$ person-years.

Example.

Given:- Size = 33.2 KLOC

EAF = 10.2

(i)

① $E = 2.4 \times (33.2)^{1.05} = 94.69$ person-months

② $D = 2.5 \times (94.6)^{0.38} = 14$ months.

③ $N = E/D = \frac{94.69}{14} = 6.7$ persons.

(ii) $E = 3.2 \times (33.2)^{1.05} \times 10.2 = 1291$ person-months.

SOFTWARE METRICS

1. Productivity metrics o/p.
2. Quality metrics.
3. Technical metrics (logical complexity of s/w, degree of modularity)

Why measure s/w.?

1. To indicate quality of product.
2. Productivity of people.
3. Assess the benefits (quality & productivity)
4. Form baseline for cost estimation of resources & of schedules.
5. Justify requests for new tools / additional training.

Types of measures

Direct
(eg:- length of bolt)

1. Lines of code (LOC)
2. Execution speed.
3. Memory size
4. Defects reported over some set period of time

Indirect
(eg:- quality of bolt)

1. Functionality.
2. Quality.
3. Complexity.
4. Efficiency.
5. Maintainability.

Categories of metrics.

1. Productivity metrics. — focuses on o/p.
2. Quality metrics
3. Technical metrics.

Another category.

1. Size oriented metrics.
2. Function oriented metrics.

* Size Oriented Metrics. — used to collect the direct measures of s/w engg. o/p & quality. — focus on the size of s/w that is produced.

Project	LOC	Effort	$	pp doc	Errors	Defects	People
Alpha	12100	24	168	365	130	30	3
beta	27000	12	440	700	200	70	5
gamma	20000	63	200	1200	765	43	6
⋮							⋮

pp doc — pages per document.
(page of documentation per KLOC)

Function Oriented Metrics

- Measure of the functionality delivered by the application.
- Proposed by Albrecht.
- Suggested a measure → function points (FP).
- Domains →
  a) Info domain (No. of I/p, No. of o/p, No. of file, No. of inquiries, no. of external interfaces)
  b) Functional domain — No. of transformations.

c) Behavioural domain — No. of transactions.

Measurement Parameter	Count	Weighing factor. Simple	Avg	Complex.	
No. of user inputs.		3	4	6	=
No. of user outputs.		4	5	7	=
No. of user inquiry.		3	4	6	=
No. of files.		7	10	15	=
No. of external interfaces		5	7	10	=
Count →					

FP (Function Points)

- Adjustable FP
- Unadjustable FP

* Adjustable FP :- $FP = Count\text{-}total * [EAF]$

$$EAF = 0.65 + 0.01 * \sum_{0}^{5} f_i \rightarrow (1 \rightarrow 14)$$

* Unadjustable FP :- $FP = count\ total$

$f_i \rightarrow$ complexity adjustment values.

Scale

0 — No influence
1 — Incidental.
2 — Moderate
3 — Average
4 — Significant.
5 — Essential.

Example: Compute FP value for a project with following information domain characteristics:

No. of user I/P = 32.
" " o/p = 60
" " inquiries = 24
" " files = 08.
" " external interfaces = 02

Assume complexity adjustment values are avg. & 14 algorithms have been counted.

Measurement Parameter	Count		Weighting factor.		
			Simple	Avg	Complex
No. of user I/p.	32	X	4		= 128
No. of User o/p.	60	X	5		= 300
No. of User inquiries.	24	X	4		= 96
No of files.	8	X	10		= 80
No. of I/F.	2	X	7		= 14
					618

Count total. →

$$FP = \text{Count total} * [0.65 + 0.01 * \Sigma F_i]$$

$$\sum_{0}^{3} F_i = 14 \times 3 = 42$$

So, $FP = 618 * [0.65 + 0.01 * 42]$

$$\therefore FP = 661.26 \approx \boxed{661}$$

* Basic metrics of LOC (Size-oriented)

① Effort = Size / Productivity.

② Productivity = Size / Effort.

③ Quality = Errors / KLOC.

④ Documentation = pp doc / KLOC

⑤ Cost of a line = $ / LOC.

⑥ cost of a slw = Effort x pay.

Metrics for Source Code

(Halstead's Theory/Metrics)

η_1 = No. of distinct operators that appear in the program.

η_2 = No. of distinct operands.

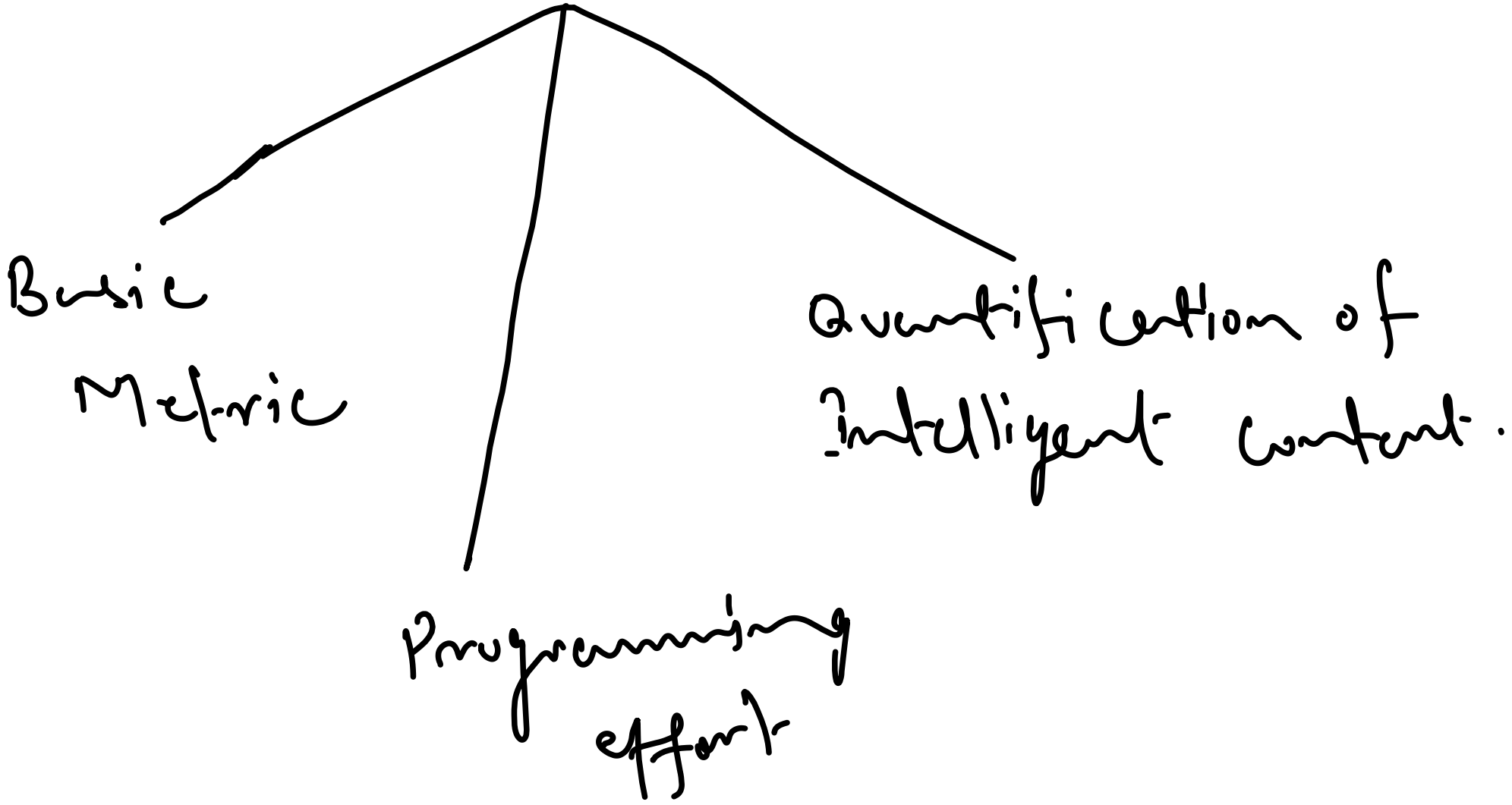
N_1 = Total no. of operator occurrences.

N_2 = Total no. of operand occurrences

η_1'/η_1^* = No. of single appearance of operators.

η_2'/η_2^* = No. of single appearance of operands.

Halstead's Metrics.



Basic Metric.

1. Vocabulary :- Total no. of unique operators & operands.

$$n = n_1 + n_2$$

2. Implementation length / Program length

$$N = N_1 + N_2$$

3. Length Equation

$$N' = \eta_1 \log_2 \eta_1 + \eta_2 \log_2 \eta_2$$

Quantification of Intelligent Content:

4. Program Volume, $V = N \log_2 \eta$

5. Program level, $L = \frac{\text{Potential Volume}}{\text{Program Volume}} = \frac{V^*}{V}$

6. Program level eqn, $L' = \frac{2}{\eta_1} * \frac{\eta_2}{N_2}$

7. Intelligent Content, $I = L' * V$

Programming Effort.

8. Potential Volume, $V^* = (2+\eta_2^*)\log_2(2+\eta_2^*)$

9. Effort eqn, $E = \frac{V}{L}$ or $\frac{V^2}{V^*}$

10. Time eqn, $T' = \frac{\eta_1 N_2 [N' \log_2 \eta]}{2\eta_2 S}$

where, S = 5-20 seconds is the human tendency to understand.

11. Programming Time,

$$P_T = \frac{\text{Effort Equation}}{\text{No. of discriminations in second}}$$

Example : Source code

```
int i, j, k, l, m;
for(i=0; i<=10; i++)
{
    if(i > 5)
    {
        printf("%d", i);
        break;
    }
}
```

unique operators

```
int , ;          = 3
for(=(=++)       = 6
{                = 1
if  >            = 2
printf "%d"      = 4
break            = 1
}                = 21
```

18

Operators		Operands.	
int , , , , ;	— 6	i j k l m	— 5
for(= ; < = ; ++)	— 8	i 0 i 10 i	— 5
{	— 1		
if(>)	— 4	i 5	— 2
{	— 1		
printf("%d",);	— 8	i	— 1
break;	— 2		13
}	— 1		
}	— 1		
}	32		

$N_1 = 32$, $N_2 = 13$, $n_1 = 18$, $n_2 = 8$

$n_1^* = 12$, $n_2^* = 7$

Soln :-

$n = n_1 + n_2 = 26$

$N = N_1 + N_2 = 45$

$N' = 18 \log_2 18 + 8 \log_2 8 = 99.05$

$V = 45 \log_2 26 = 211.51$

$V^* = (2+7) \log_2 9 = 28.52$

$L = 28.52 / 211.51 = 0.1348$

$$L' = \frac{2}{18} * \frac{8}{13} = 0.068$$

$$T = 0.068 \times 211.51$$

$$E = \frac{211.51}{0.1348} \text{ or } \frac{(211.51)^2}{28.52}$$

$$T' = \frac{18 \times 13 \left[99.05 \log_2 26 \right]}{2 \times 8 \times \underbrace{10}_{5} (\text{assuming time})}$$