DFD.

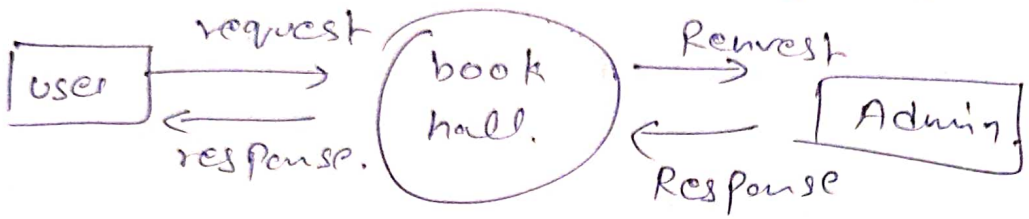
(1) show the data flow b/w various elements

(2) client requests s/w → Give it to Manager

↓ Test

deploy. ←

(3)



(4) shows the requirement of system & how current sys. is implemented.

(5) overview of what data a system process, what transformation are performed, what data are stored, produced, where it flows.

(6) Good communicat tool b/w user & system designer.

Symbols

→ circle ○ Process.

→ open rectangle ⊏ data store.

→ Rectangle ▭ Entity.

→ Arrow → ← data flow.

logical.	physical.
how data flows in system.	how data actually implemented.
what data moved from one entity to another.	how data moved.
implementation independent	Dependent
how system works nor how sys. is implemented.	how system operates & implements

0 level DfD :- Abstract view / Minimal view.

→ whole system as a sigle process communicating with ext. entities.

1 level. decomposed 0 level into multiple components & define each component in detail.

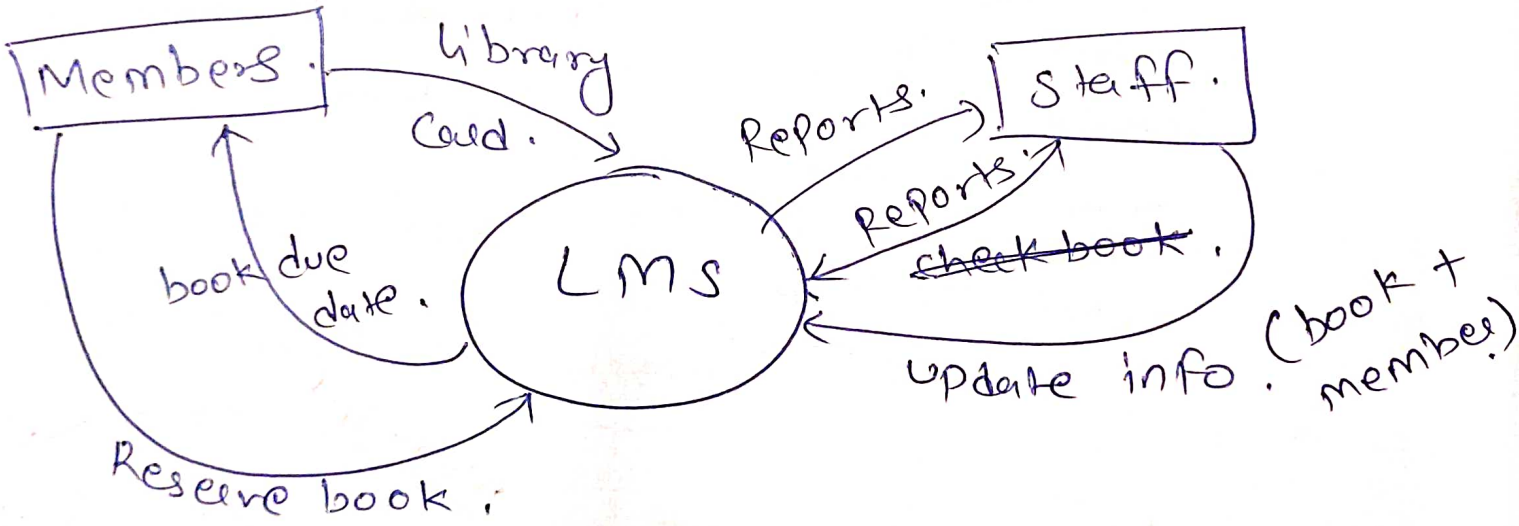
2 level decomposes level - 1 . further

DfD.

DfD level 0./context level.

→ It shows whole system as a single process & shows diff. entities & data exchage b/w process and entities.

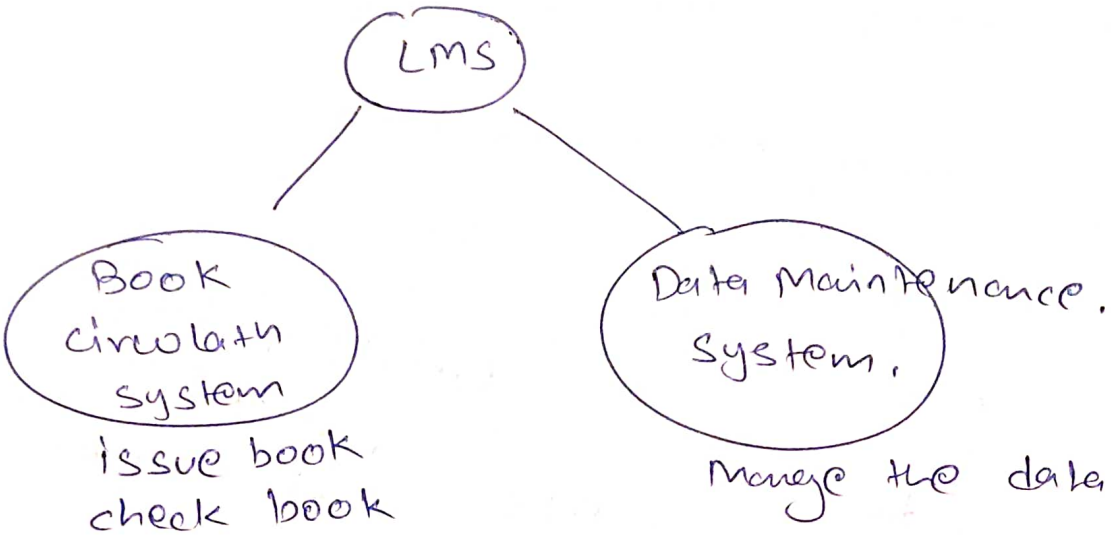
Ex: Library Mgt. System

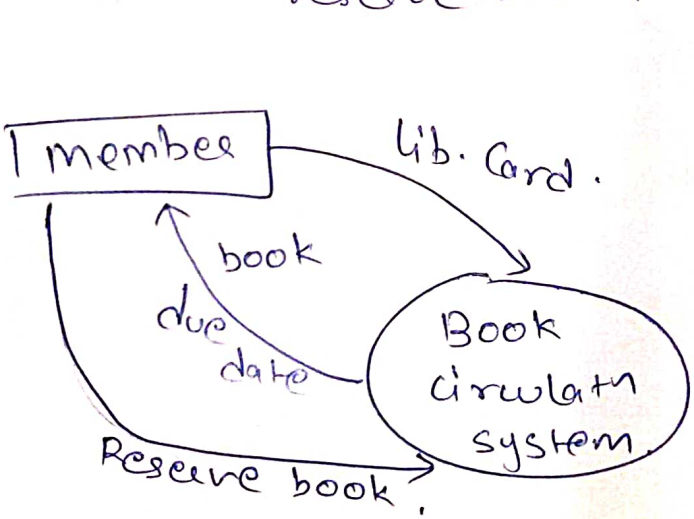


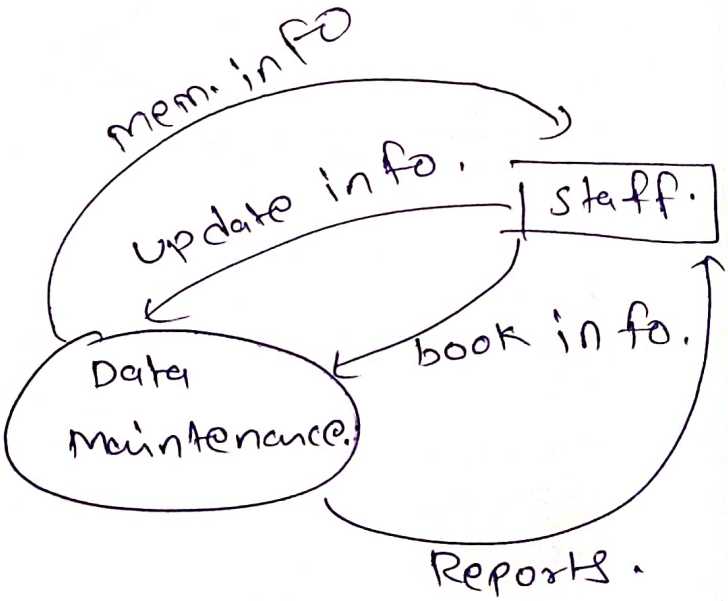
DfD level 1.

→ More elaborated dfD.

→ level 0 PfD Process is divided into multiple components.







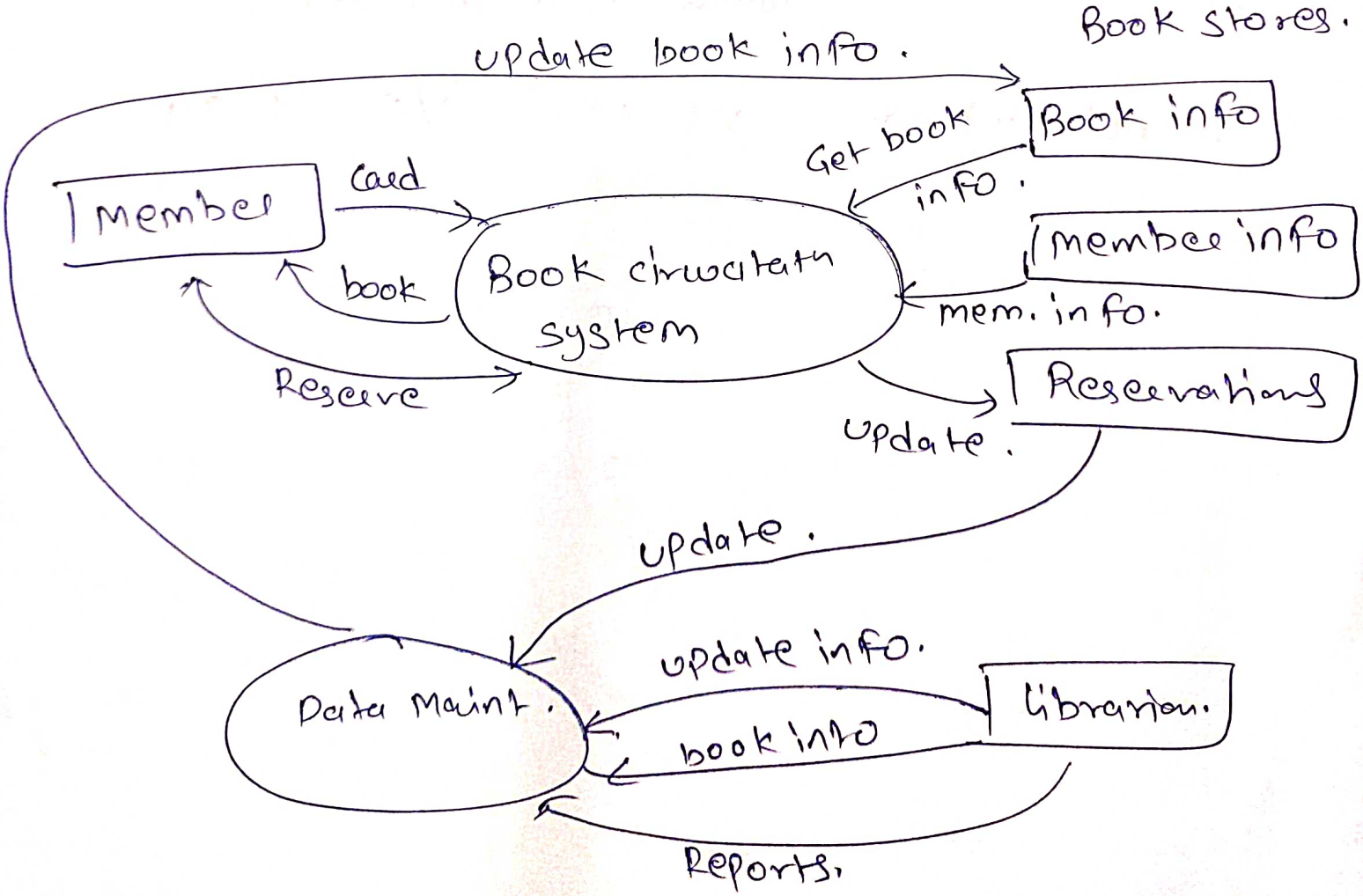
→ Decomposing system into subsystems.

→ Do not introduce any new inputs/outputs with the external entities.

→ In level 1 DfD, data stores are used.

→ In level 0 DfD, No data stores are there.

→ Ex: Data stores
Loans. – All info related to issues book
Mem. info table –
Reservation table –
Book info table –



level 2 DfD

→ what if we divide subsystem of level into various process.

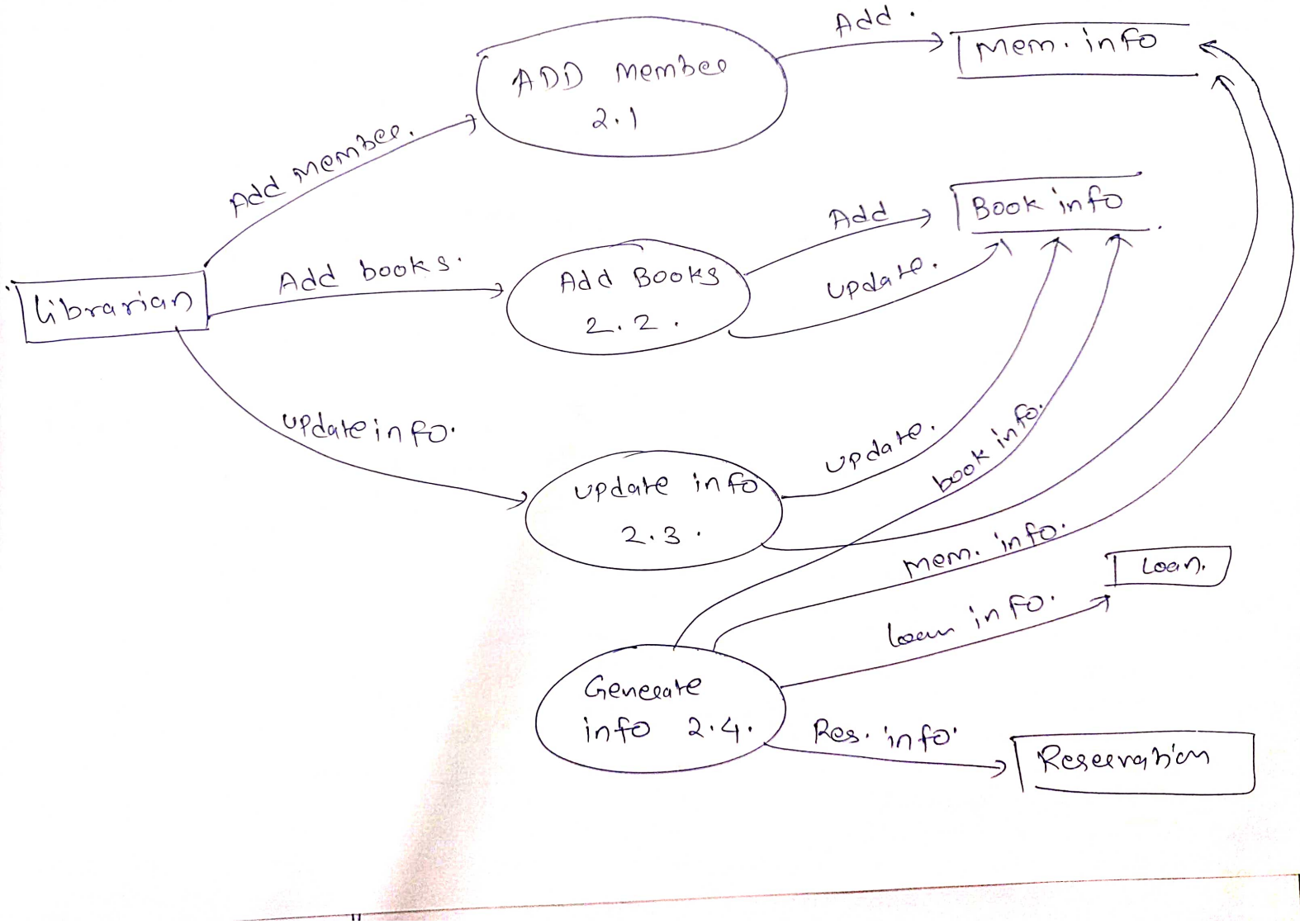
→ where each process responsible for each func

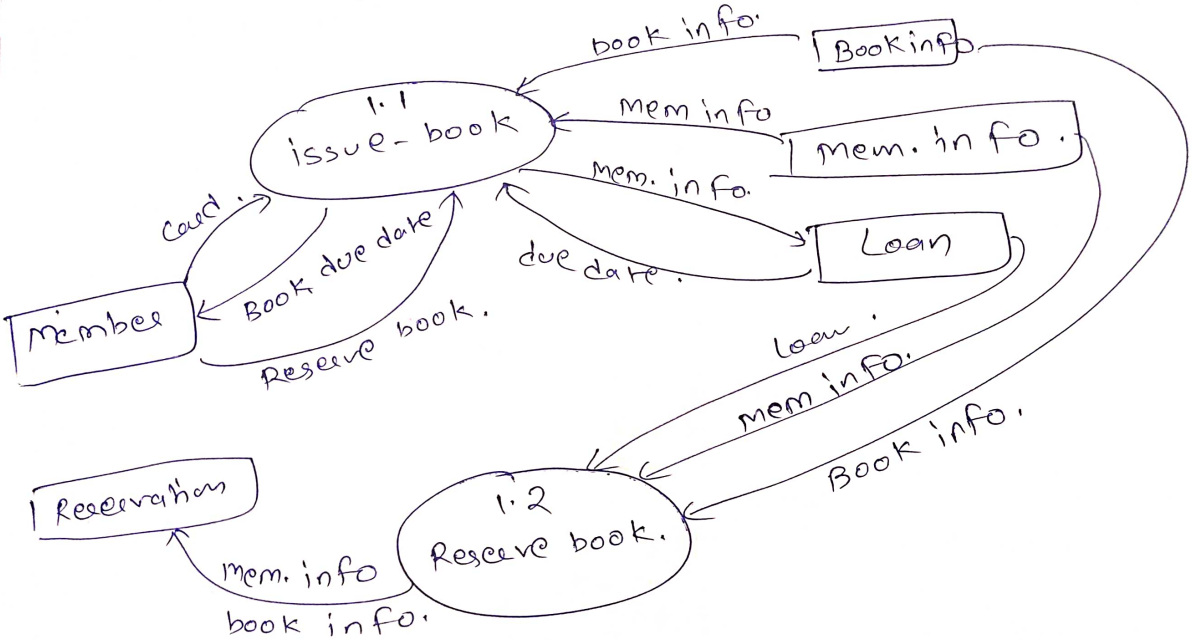
Ex. one process to issue book

one process to reserve book

one process. to send notificatn to custom when book available.

→ Do not introduce any new i/o & o/p.


Librarian
Add member.
ADD member 2.1
Add.
Mem. info
Add books.
Add Books 2.2.
Add
Book info
Update.
Update info.
Update info 2.3.
Update.
book info.
mem. info.
Loan.
Loan info.
Generate info 2.4.
Res. info.
Reservation



~~1.1 & 1.2.~~

* 1.1 & 1.2 are the division of book - circulation system.

~~send notification.~~

Software Metrics

→ for a project proper planning, for goal setting. we use metrics.

Ex: Agile Process metrics.

→ lead-time → time taken to deliver.

→ velocity :→ no. of o/p. per sprint.

→ cycle time → time to implement changes.

→ efficiency.

→ Mean time to Recover (MTTR)
↳ After any failure/change.

Size oriented metrics

→ KLOC (Kilo lines of code).

function oriented metrics

→ function Points.

How can S/W Metrics be Used

(A) Project Planning.

→ Gives a ideal picture of how team performed in previous iteration.

→ Now he/she can better estimate budget, cost, time, resources for future.

(B) Project overview

Easily can access the current states, challenges etc.

(3) Task Prioritization.

— which task to finish in which sequence to maximize value.

Ex. User satisfaction is low due to some quality s/w problem.
so instead of delivering new features, why not focus on problematic areas.

(4) change Mgt. Ex MTTR.

function Point Analysis.

(1) Unit of measurement of software.

(2) what do these measure ?

- functional user requirements.

↓

what a system must do in terms of data transfer, transformation, storage, Processing.

→ funct^n user req. are not funct^n requirem

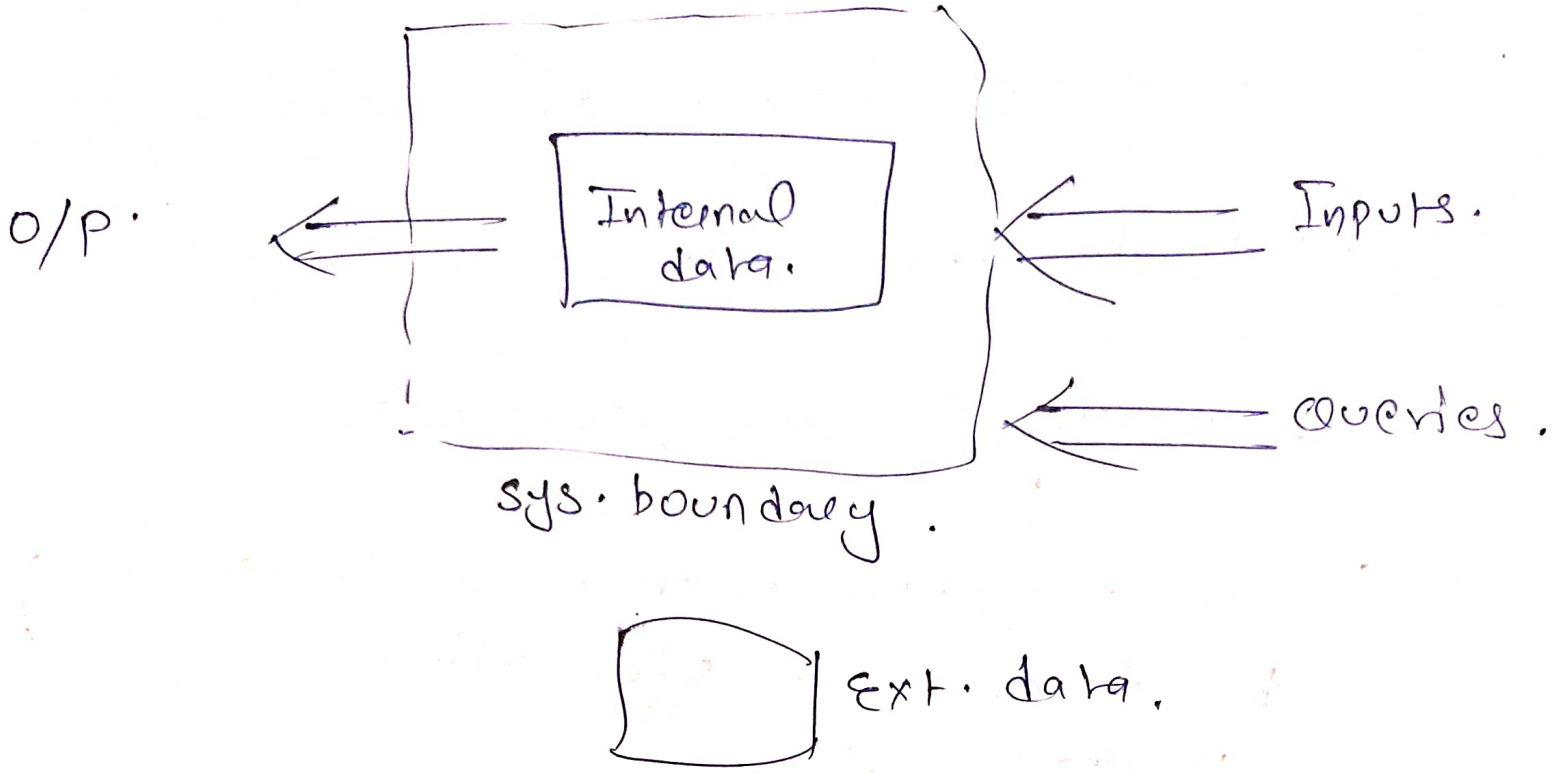
↓

Ex. for a bank, → bank data base

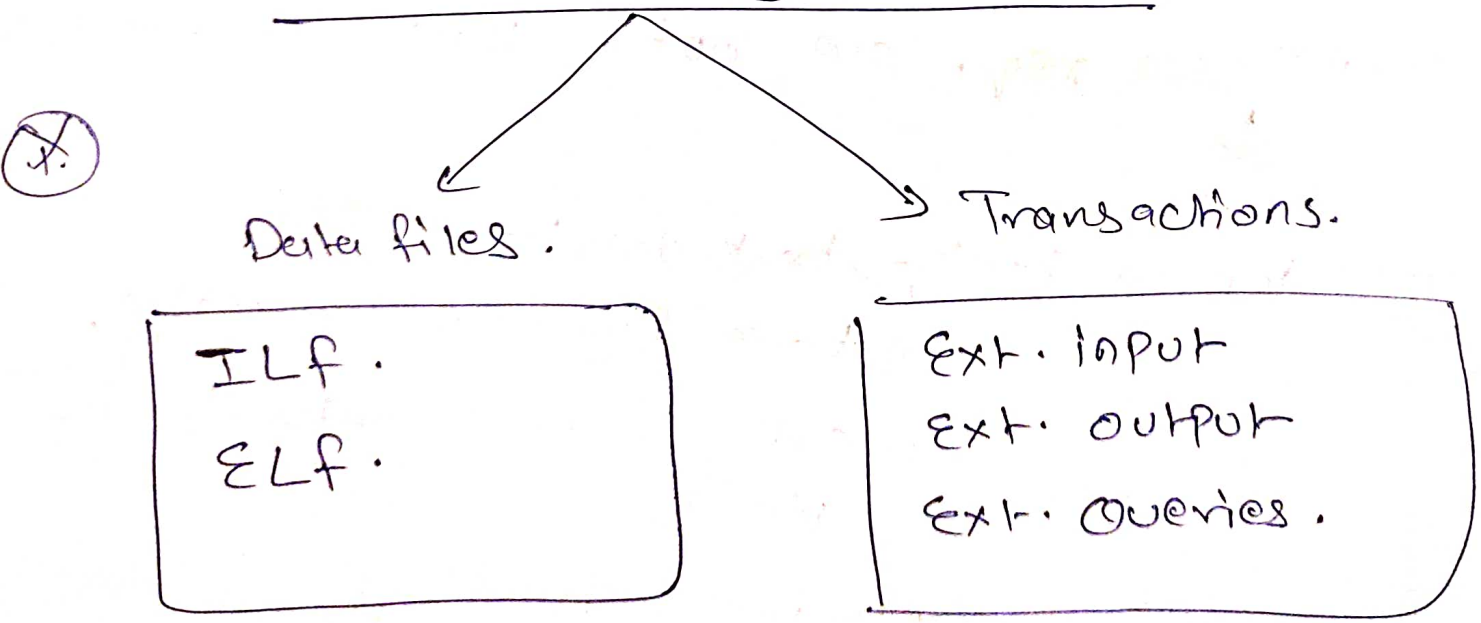
→ bank Payments Processing.

→ adding a new contanct in phone,

→ , Business Processing.



5. Imp. elementary Processes.



E.F.

① Internal logical files. (* Interface ~~not~~ logical)

- maintained by application
- logically related.

Ex. customer & entries - Purchases are 2 ILF.

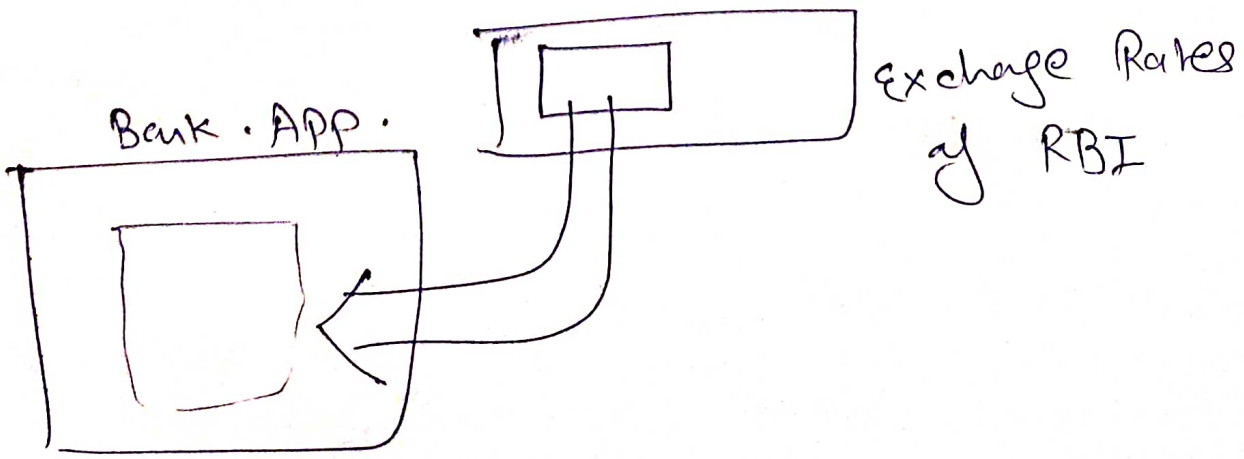
- contacts in a contact application.

② Ext. logical files. EIF.

↑ interface.

→ Refereenced by applicatn. (Read only)

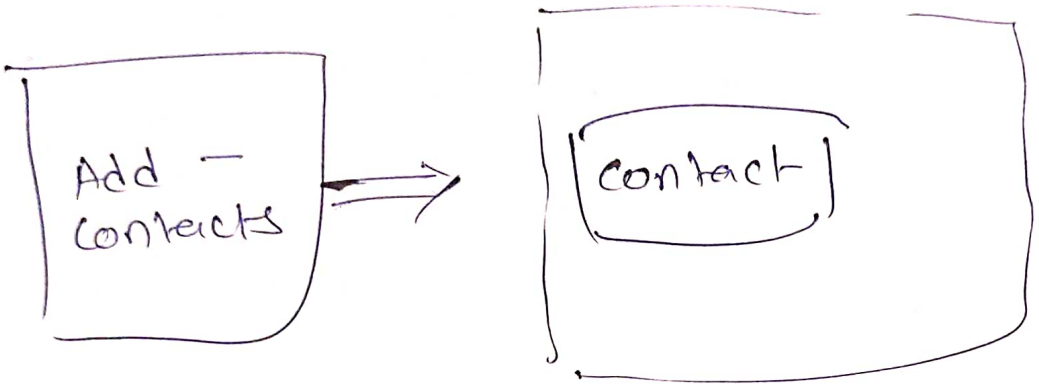
→ owned by another application.



(3) Ext. Input EI

→ Brings data into application.

→ Maintains ELF.



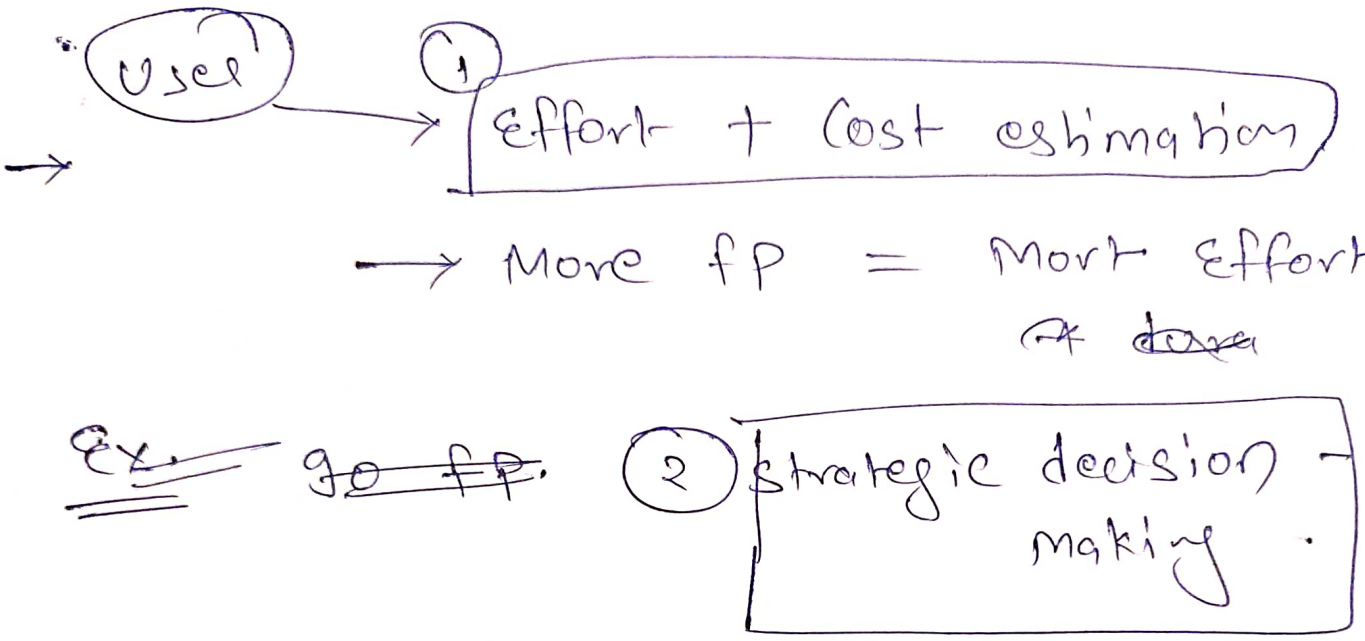
(4) Ext. O/P. → o/p given to user

(5) Ext Queries.

→ Presents data to user.

→ only data retrieval + Presentation.

function Points.



→ More fp = More efforts.

~~Ex. 90 fp.~~

Ex. 90 fp. ⇒ 120 fp

→ whether to build an application or to buy it from ext. sources.

function point.

→ To measure the s/w worth, funct^{n} point developed.

→ Measures functionality from the user point of view as what user receives from s/w & what it request.

→ what functionality f.p. delivers.

→ fp :— (1) Estimate the cost & effort to design, code, test software.

(2) Predict no. of error s/w encounter during testing.

(3) forecast the no. of components & source lines in an implemen. system.

calculate fp.

(1) ILf - Internal logic files.
Control info./logic related data within sys.

(2) ELf - Control data referenced by sys. that present in another system.

(3) EI External input.

(4) EO -11- output.

(5) EQ External Enquired.
- combo of i/o - o/p.

$$f.p. = ufp \times CAf.$$

ufp → unadjusted f.p.

CAf. → Complexity Adj. factor.

(2) $CAf = 0.65 + (0.01 \times \sum f_i)$

$f_i \rightarrow$ VAf (value Adj. factor) based on responses to following 14 questions.

1. Data comm.
2. Distributed data Proc.
3. Performance.
4. Heavily used config.
5. Transactn Rate.
6. online data Entry.
7. End user η.
8. online update.
9. Complex. Processing.
10. Reusability.
11. opeatn Ease.
12. Translatn Ease.
13. multiple site.
14. facility chnge.

on scale of 0-5

0 - No Imp
1 - Incidental
2 - Moderate
3 - Average.
4 - Significant
5 - Essential

Q. Moderate complex Project.

$$\Sigma f_i = 14 \times 2 = 28.$$

$$CAf = 0.65 + 0.01 \times 28$$

$$= 0.93.$$

Q: Given following values, calculate f.p. when CAf — significantly complex & weighting factors are high.

UI = 55
UO = 35
U. Enquiry = 40
user files = 8.
EI = 5.

		Count		low Avg	(High)		
Avg.	EI	55	x		6	=	330
	EO	35	x		7	=	245
	EQ.	40	x		6	=	
	ILf	8	x		15	=	240
	ELf	5	x		10	=	120
							50
						+	
					Total	=	985

① Calculate ufp. —

Functn unit	weight factor * w.f.		
	low	Avg.	High.
EI	3	4	6
EO	4	5	7
EQ.	3	4	6.
ILf	7	10	15
ELf.	5	7	10

⇒ ufp is sum of all complexities of EI EO EQ ILf, ELf.

	count.	w.f: low	Avg	High	
EI	□	3	4	6.	□
EO	□	4	5	7	+
EQ	□	3	4	6	□
Ilf	□	7	10	15	+
ELf.	□	5	7	10	□
					+
					□
					+
					□

ufp = Total count = □

$$CAf = 0.65 + (0.01 \times \Sigma fi^{0})$$

$$\Sigma fi^{0} = 14 \times 4 = 56.$$

$$\Sigma Af = 1.21.$$

$$fp = ufp \times CAf$$

$$= 985 \times 1.21 = 1191.85$$

Q. Given following value, compute f.p. when all CAf & weight factors are Average.

UI = 50

UO = 40

UE = 35

IL files = 6

ELf = 4.

45 4 10 7

(1) $ufp = 50 \times 4 + 40 \times 5 + 35 \times 4 + 6 \times 10 + 4 \times 7.$

$= 200 + 200 + 140 + 60 + 28$

$= 628.$

(2) $CAf = 0.65 + 0.01 \left[14 \times 3 \right]$ ↑ Average

$= 0.65 + 0.42 = 1.07$

(3) $fp = 628 + 1.07 = 629.07$

Functn Point Approach

(U)

(1) size of s/w delivered is measured independent of language & tools.

(2) fp is directly estimated from requirement, before design & coding.

(3) useful for those users without technical expertise.

Project Estimation

1. Project size
2. Efforts needed to complete Project.
3. Project duration.
4. cost.

These are the parameters which need to be estimated.

As its imp. for resource planning + scheduling.

3 types of Estimation Techniques :-

1. Empirical E.T.
2. Heuristic E.T
3. Analytical E.T.

1. Empirical E.T. → Educated guess of Project parameters.

→ Based on prior experience of developement of similar Project.

✱

(2) ~~Expe~~ Heuristic E.T.

→ Relationship that exist among the diff. Project Parameters can be modeled using suitable Mathematical Expression.

→ once independant parameters are known, dependent Parameter can be easily determined by substituting the value of indep. Paramter in Expression.

→ Can be classfied as single variable & Multivariable models.

(1) single variable model.

$$\text{Estimated Parameter} = c_1 e^{d_1}$$

e → independant variable characteristic. [Already been estimated].

c_1 & d_1 are constant, determined using data collected from Past.

Ex: COCOMO Basic Model.

COCOMO

Basic Model.

1. Effort $a\ (KLOC)^b$ Person-month

2. Development time. $c \cdot (Effort)^d$ month.

3. Avg. staff size $\rightarrow \left(\frac{Effort}{dev.\ time}\right)$ Persons

4. Productivity $\rightarrow \left(\frac{KLOC}{Effort}\right)$

Q. Project is estimated to be 400 KLOC. Calculate effort & time for each of 3 modes of development.

Mode.	a	b	c	d.
organic	2.4	1.05	2.5	0.38
semi detached	3.0	1.12	2.5	0.35
Embedded.	3.6	1.20	2.5	0.32

1. organic.

$$\text{effort} = 2.4\,(400)^{1.05} = 1295$$

$$\text{dev. time} = 2.5\,(1295)^{0.38}$$

$$= 38 \text{ Months}.$$

2. Semi detached.

$$\text{effort} = 3\,(400)^{1.12} \cong 2462.$$

$$\text{dev. time} = 2.5\,(2462)^{0.35}$$

$$\cong 338.4.$$

3. Embedded.

$$\text{effort} = 3.6 \times (400)^{1.2}$$

$$\cong 4772.$$

$$\text{dev. time} = 2.5\,(4772)^{0.32}$$

$$=$$

Q:

Intermediate COCOMO

Basic COCOMO

→ Static model to estimate S/W development effort quickly & roughly.

→ Deals with no. of lines of code.

→ Estimation Accuracy is less as we don't consider all parameter of Project.

Intermediate COCOMO

→ Estimates software development effort in terms of size of Program & other related cost drivers (Project parameter, H/W parameter, resource parameter).

$$E = a(KLOC)^b * EAF$$

$$Time = c(effort)^d$$

$$Person\ required = Effort/time$$

①

Parameter	organic	Semi Detached.	Embedded.
A	3.2	3.0	2.8
B	1.05	1.12	1.20
C	2.5	2.5	2.5
D	0.38	0.35	0.32

② Effort Computation.

- Cost drivers. very low, low, Nominal, High. very high, Extra High.
- Reliability.
- DB size.
- Product Complexity

(1)

Q. for a given Project was a estimate with size of 300 KLOC. Calculate effort, scheduled time for development by considering High applicatn Expeience & very low exp. in Programing.

Ans. Develope highly applicatn experience.
= 0.82 [As per table].

- Developer very low exp. in Programing
= 1.14 [As per table]

(1) EAf = 0.82 × 1.14 = 0.93

(2) Effort = a (KLOC)*b EAf

= 3.0 (300)^1.12 0.93

= 1668 MM.

(3) scheduled time = C (E)^d

= 2.5 (1668)^0.35

= 33.5 Months.

(1)

Parameter	organic	Semi Detached	Embedded
A	3.2	3.0	2.8
B	1.05	1.12	1.20
C	2.5	2.5	2.5
D	0.38	0.35	0.32

(2) Effort computation.

- Cost drivers. very low, low, Nominal, High, very high, Extra High.
- Reliability.
- DB size.
- Product Complexity

a broken

Testing.

1. white box
2. Black box.
3. Acceptance.
4. Regression.
5. unit Testing
6. Integration testing.
7. system Testing.

Acceptance Testing

(1) Test during pre release phase.

(2) Identify Problems before customer receives product.

(3) Effective way to take a feedback. at. early stage

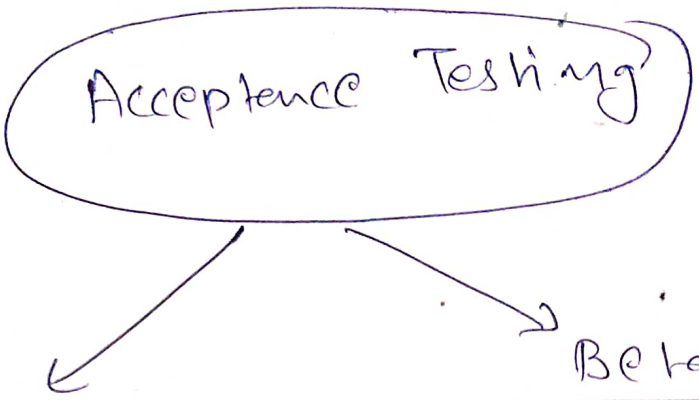
(4) Involves stakeholders, developers, subject matter experts.

Ex. Paytm wants to make applicatn for Payment. Tcs is given contract.

Tcs completes Application but before delivering to paytm it tests whether paytm will accept it.

Why → (1) Requirement may be chnged during developement, not effectively communicated to team.

→ Developers can understand the actual requirement of customer.



Alpha

Test on their own (Manual / automate)

Before releasing to public.

Beta.

→ Product team delivers a near-finished product to group of target users to evaluate performance in real world.

Regression Testing

→ Ex. - Tester reports a broken login button.

- once developers fix bug, login button is tested for expected results.

- Testing to confirm that a recent program/code change has not adversely affected existing features.

- Methods

→ Retest All. !- Expensive

→ Reg. Test selection -

- only few selected test cases.

→ Prioritize the Test Cases -

- depending on business Impact, critical functionality.