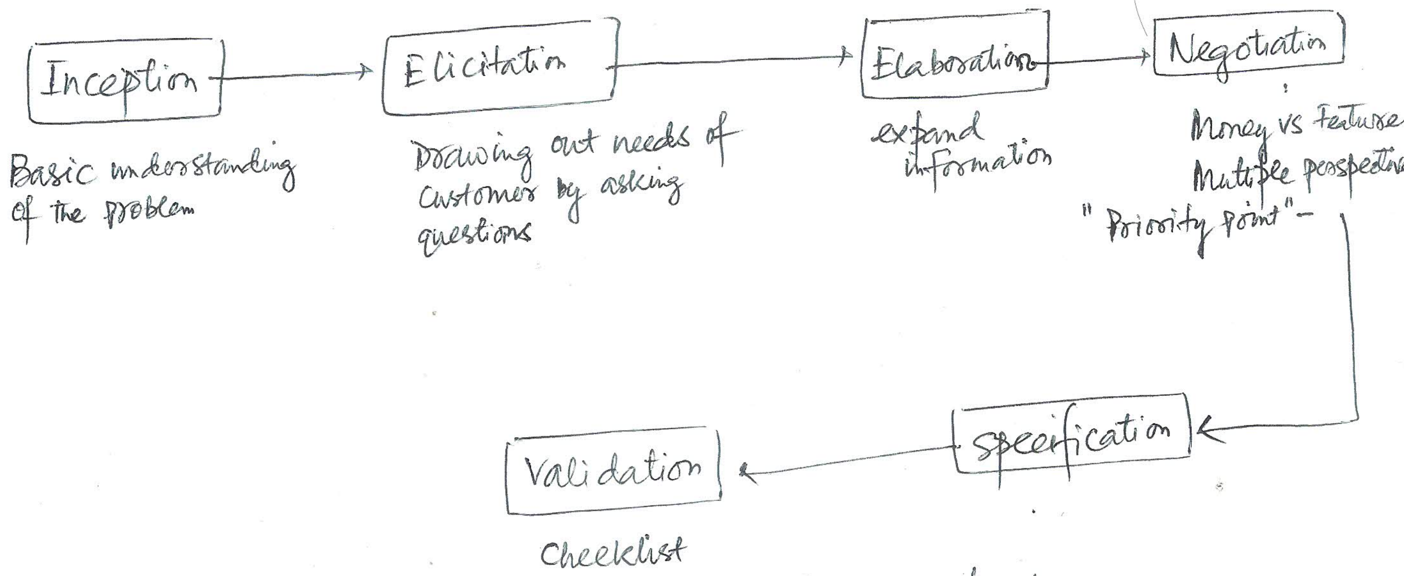
# Requirement engineering

Meeting, Interview, Observation, Survey, Past data . . . . .



Inception → Elicitation → Elaboration → Negotiation → specification → Validation

Inception: Basic understanding of the problem

Elicitation: Drawing out needs of customers by asking questions

Elaboration: expand information

Negotiation: Money vs Feature
Multiple perspectiv
"Priority point" –

Validation: Checklist

Customer/End user may be located in different city/country.
may have vague idea of what is required.
conflicting openion
limited time to interact

CHALLENGES

Stakeholders: person with direct interest or benefits from the system
(Product managers, Marketing people, Internal-external customers, engineers etc.)

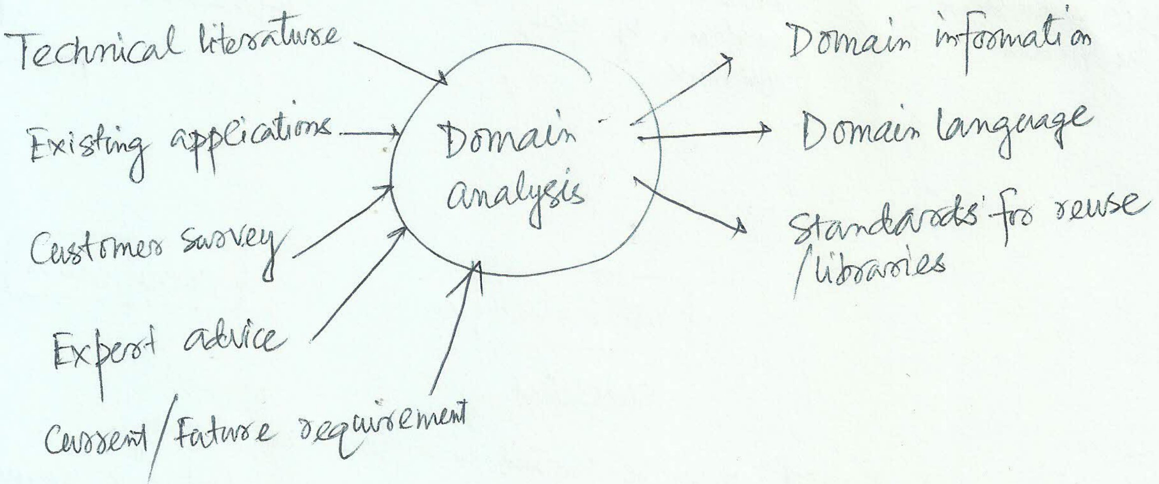
Recognizing multiple viewpoints
Collaborate among stakeholders —— priority point based voting

Meeting ~~with~~ by stakeholders —— after intial meeting one-or-two page "product request" is written & distributed to attendees.

Customer voice table: Raw data gathered for requirement is translated into table of requirements which are reviewed by stakeholders.

## Domain analysis

Dis cover ~~related~~ information related to similar application.



## Data modelling

Data object → Data members → Data attributes

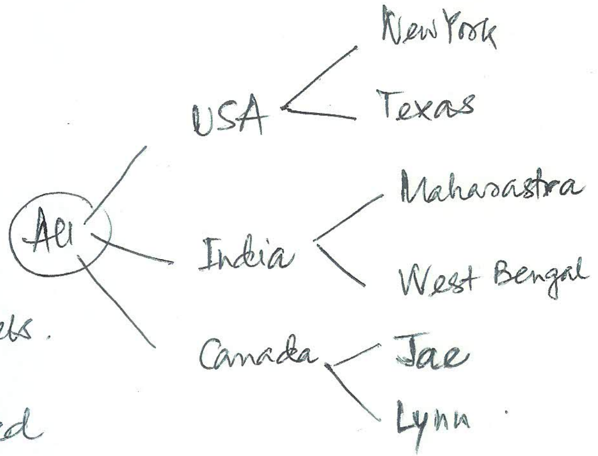
Class diagram

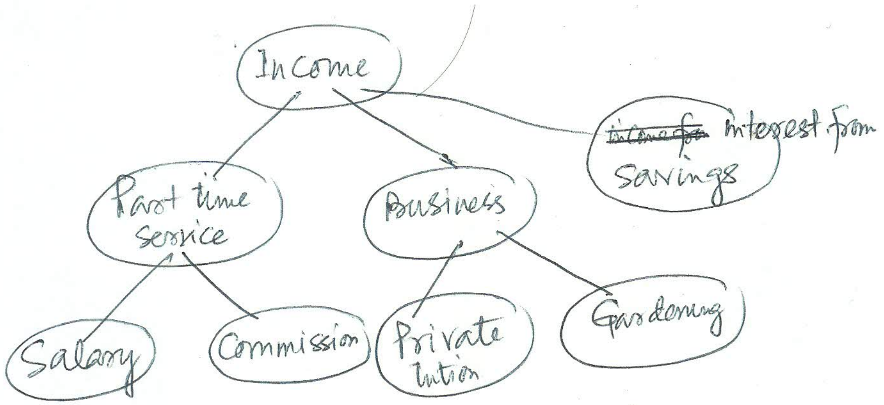
ER diagram.

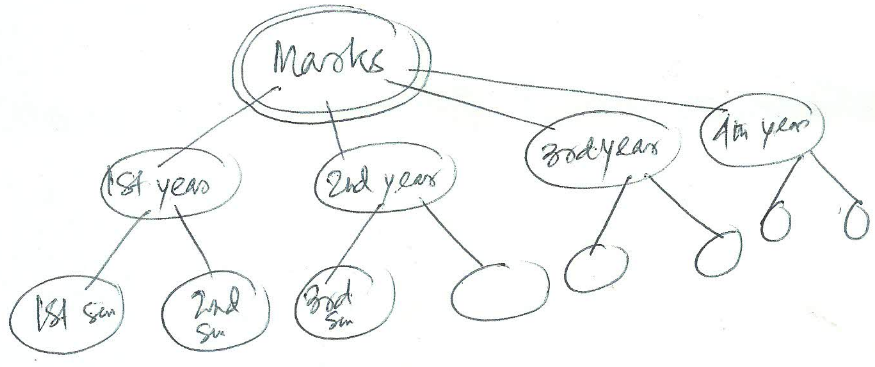
## Data tree

Displays hiearchial structure of data — used for visualizing data broken down into multiple levels.

Data —— expanded or collapsed at any level







# Functional & Non-functional requirements

1.

| Requirement ID | Requirement Statement | Priority | Comment |
|---|---|---|---|
| FR001 | | | |
| FR002 | | | |
| FR003 | Website has page that lists purpose of organization | High | |
| ⋮ | | | |
| NFR 0010 | Scalability | Normal | |
| NRF011 | | | |

{High, Normal, Low}

2.

| Req. ID | Functional requirement statement | Priority | Comment |
|---|---|---|---|
| FR001 | | | |
| FR002 | | | |
| | ⋮ | | |

| Req ID | Non-functional requirement statement | Priority | Comment |
|---|---|---|---|
| NF001 | | | |
| NF002 | | | |
| ⋮ | | | |

# A good SRS document

1. Concise, unambiguous, consistent ~~and~~, complete, correct, clear

~~2.~~

2. Well-structured

3. **Black box view:** Should specify external behaviors of the system — not discussion of any implementation issue — Right level of abstraction according to the purpose of SRS — independent of design issues.

4. Reader can understand easily / Understandable

5. Requirements should be verifiable.

6. Testability: Easy to generate test cases and testing plan.

7. Modifiability: Easy to modify

8. Ranking: Requirements are stated with priority.

---

- Avoid overspecification
- Avoid referring element discussed much later in SRS.

# Representation by proposition logic

A proposition is a declarative statement which is either true(T) or false(F).

## Representation format

Propositional symbol/Atom/variables $P, Q, R \ldots$

Logical constants $T, F$

Logical operators $\wedge, \vee, \sim, \rightarrow, \leftrightarrow$

"If I look into the sky and I am alert then I ~~see~~ will see a dim star or if I am not alert then I will not see a dim star"

$P \rightarrow Q = \sim P \vee Q$

$P \leftrightarrow Q = (\sim P \vee Q) \wedge (\sim Q \vee P)$

P: I look into the sky

Q: I am alert

R: I will see a dim star

$\alpha : (P \wedge Q \rightarrow R) \vee (\sim Q \rightarrow \sim R)$

To check if $\alpha$ is valid or not

| P | Q | R | $\alpha$ |
|---|---|---|---|
| T | T | T | |
| T | F | T | |
| T | T | F | |
| T | F | F | |
| F | T | T | |
| F | F | T | |
| F | T | F | |
| F | F | F | |

John or Mary or both will go to cinema. If John goes then Jenny will go. Mike will go if Mary goes. Mike does not go to cinema. Conclude that Jenny will not go to cinema.

P : John goes to cinema.

Q : Mary " " "

R : Jenny " " "

S : Mike " " "

$P \vee Q \vee (P \wedge Q)$

$P \rightarrow R$

$Q \rightarrow S$

$\sim S$

$$((P \vee Q \vee (P \wedge Q)) \wedge (P \rightarrow R) \wedge (Q \rightarrow S) \wedge \sim S) \rightarrow \sim R$$

# First Order predicate logic

**A predicate** is a relation that maps $n$ ~~terms~~ values into true (T) or false (F).

$$P(t_1, t_2, t_3 \cdots t_n) \in \{T, F\}$$

x is greater than y  $\quad$ $GREATER(x, y) = T$ if $x > y$
$\qquad\qquad\qquad\qquad\qquad\qquad\;\; = F$ otherwise

x likes y $\quad$ $LIKE(x, y) \in \{T, F\}$

x is a natural number $\quad$ $NATURAL(x)$

Predicate symbols are written in uppercase letters.

**A function** is a mapping that maps $n$ ~~terms~~ values to a ~~term~~ value.

$$\Gamma^n \rightarrow \Gamma$$

$$f(t_1, t_2 \cdots t_n) = \Gamma$$

father (Suresh)

plus (2,3)

**Quantifiers** $\quad$ $\forall$ (for all) $\quad$ $\exists$ (There exists)

**Logical operators** $\quad$ $\sim, \wedge, \vee, \rightarrow, \leftrightarrow$

Suresh likes Anupam

LIKE(Suresh, Anupam)

Suresh likes somebody

$\exists x$ LIKE(Suresh, $x$)

Suresh likes everyone

$\forall x$ LIKE(suresh, $x$)

Every person is liked by somebody

$\forall x \exists y$ LIKE($x$,$y$)

Representing Grandfather

✓ PARENT($z$, $y$)

$z$ is parent of $y$

✓ FATHER($x$,$z$)

$x$ is father of $z$

✓ MOTHR($x$,$z$)

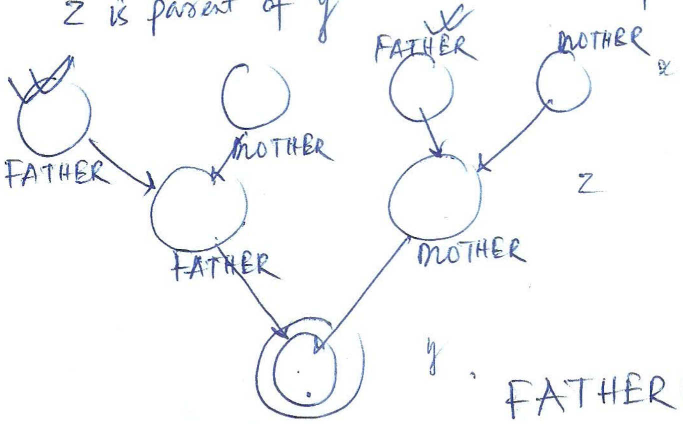
$x$ is mother of $z$.

✓ GRANDFATHER($z$, $x$)

$z$ is grandfather of $x$

PARENT($x$,$y$)

$x$ is parent of $y$



FATHER($z$,$y$) $\vee$ MOTHER($z$,$y$) $\rightarrow$ PARENT($z$,$y$)

PARENT($z$,$y$) $\wedge$ FATHER($x$,$z$) $\rightarrow$ GRANDFATHER($x$,$y$)

~~$\forall x \exists y$ FATHER($x$,$y$)~~

every person is father

$\boxed{\forall x \exists y \text{ FATHER}(x,y)}$

$\boxed{\forall x \exists y \text{ FATHER}(y,x)}$

every person has father