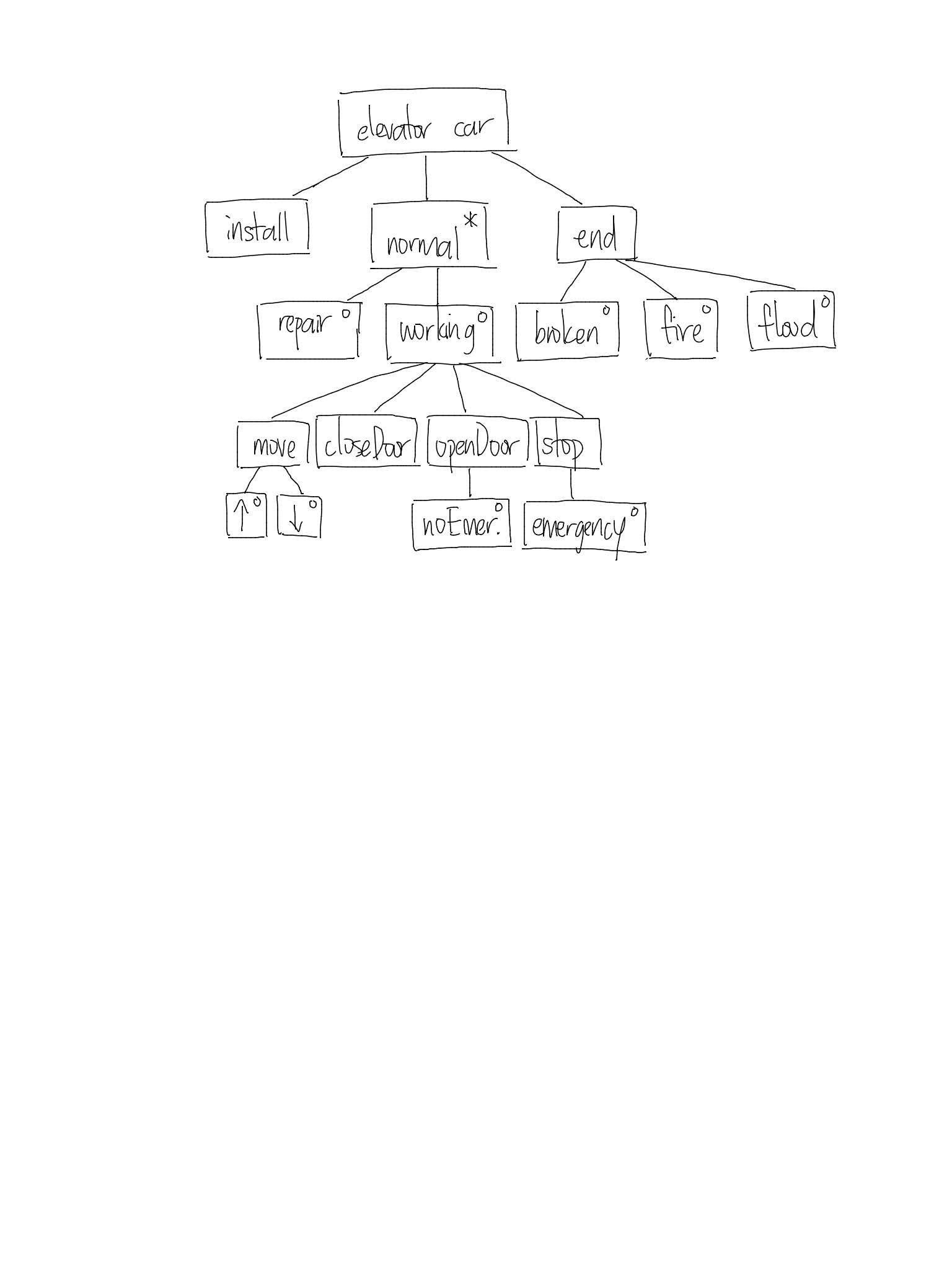
JSD Tree

* ID entity & services
* > 1 alternatives
* OOA/OOD

EX: Elevator Control

* Entity: Car
  + Constructor: Install
  + Destructor: totally broken, on fire
  + Normal Working: moving (up/down), openDoor, closeDoor, stop, emergency, repair



3 Components to Planning

1. Personnel
2. Money, pricing
   1. software cost estimation: effort to reasonably measure
   2. mergeSort, quickSort
3. Software/Hardware: Operating System, Programming Language, CASE tool
   1. computer aided software engineering
   2. IDLE: smart editor CASE
   3. Constraints: time delay
      1. Eigenanalysis O(n3) Singular Value Decomposition (SVD): factorizes linear operator A:Rn → Rm into 3 simpler linear operators (A = USVT)

How to Offer Price Tag, Estimate Software Cost

1. Time/human hrs (agile problem)
   * most popular, simple/easy, based on trust
   * pros: simple
   * cons: easy to cheat, hard to evaluate/justify
2. Task based/features
   * pros: ez to evaluate, hard to cheat
   * cons: not simple, who should decide? could be unfair
3. # of lines, size of final code
   * popular, simple, easy, based on trust
   * pros: simple, no controversy
   * cons: motivate programmers to cheat, discourage creative thinking
4. Expert opinion
   * seek help of experts who have experience in similar projects
   * pros: easy/outsource
   * cons: do you trust if they’re expert? honesty, can be expensive with more experts
   * Get second opinion
     + get more experts
     + committee vote
   * ML: ensemble learning/random forest
   * More experts == more $$$, expensive
5. Self, Prior experience
   * use your own prior experience, by self analogy
   * pros: no trust/honesty/cheating problem
   * cons: over-confident/not fit
6. Weighted sum of lines (δ/# of lines)
   * Assign diff $$ to diff lines
     + recursion/loop: $20
     + condition: $15
     + assignment: $1
     + Comments: $0.1
   * pros: more fair than # of lines (wc -1)
   * cons: encourage/inspire/motivate to cheat, honesty

Personnel

Grouping/Format

* Pure Democracy: all members created equal
  + Hard to make any decision/too many comm channels
* Dictatorship: one leader decides everything ($, hire/fire, jobs, gui)
  + Leader: manager + tech leader (double-head)
  + Team:
    - Einstein + 4 People like us
    - Your emperor/king
  + Generals v close: worry about own safety (don’t want to get Caesar'ed)
  + Ingres: UC berkeley

Group Cohesiveness

1. Close: very close like family

* pro: ego-less coding, happy, willing to help
* cons: group-think (low quality), danger to higher-ups (Ingres Top 3 DB)
  + avoid too close relations

2. Loose: treat each other like enemies

* pros: work v hard to establish
  + QS: choose the 1st element as anchor, instead you should randomize
  + you protect yourself, make sure you have no weakness in program
* cons: hostile, feels v bad, spend too much time/effort/$$$ for security

3. Right Format: professional, neither too close nor too loose, objective

* several types:
  1. professional: do the job, deliver
  2. perfectionist: unless the job is perfect, they won’t stop
  3. interaction: love socializing, talking, presentation
* in team: 3(a), 1(b), 1(c)
  + most should be (a)
* team leader: (a) or (c), never (b)

System Theory: the whole <> the sum of parts

4 people together, the performance of team

* 4x100 meter relay: positive/negative example
* world record of 4x100 < 4 x 100 dash record
* 4 best runners doesn’t mean you win
  + even top can end up fighting each other
* with practice, environment, working together, then the team produces successful result
* team player: don’t be too difficult to work with