SOFTWARE ENGINEERING

Lecture 1

Terminology

Engineering :

The application of a systematic, disciplined, quantifiable approach to structures, machines, products, systems or processes.

- IEEE (1990)

The application of knowledge in the form of science, mathematics and emperical evidence, to the innovation, design, construction, operation and maintainance of structures, machines, materials, software, devices, systems, processes and organizations.

→ Wikipedia

System → think about what a system ?

	Non - Engineering	Engineering
Deadline	cannot be planned	can be planned with sufficient precision

Price	determined by market value, not cost	oriented on cost, calculable
Evaluation & Comparision	Subjective	Objective, quantified criteria
Norms & Standards	Rare	Exist
Warranty	not defined, hard to enforce	clearly regulated cannot be disclaimed.
Mental PreReq	Art inspiration	Technical Know-how
Author	consider artwork a part of himself.	remains anonymous.

<u>Software</u>

Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. IEEE 1990

1> all or part of the programs, procedures, rules and associated documentation of an information processing system.

2> see 6.10.2

3> program or set of programs used to run a computer

↳ Systems and Software engineering Vocabulary 2010

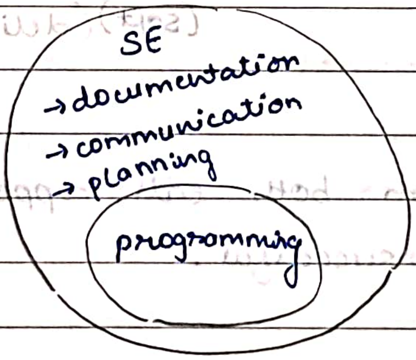
↓

NOTE : Includes firmware, documentation, data and execution control statements.

"The application of a systematic, disciplined, quantifiable approach to the development, operation and maintainence of software; that is, the application of engineering to software."

↳ Software Engineering IEEE 1990

What's the difference between software engineering & programming?



Software Engineering : Multiperson development of multiversion programs

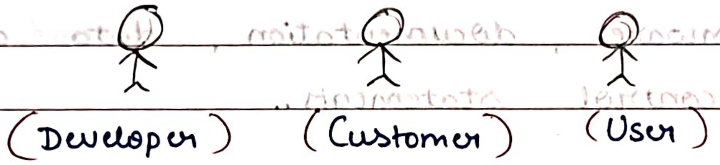
– Parnas 2011

II Succesful Software Development

1) Working definition : succes

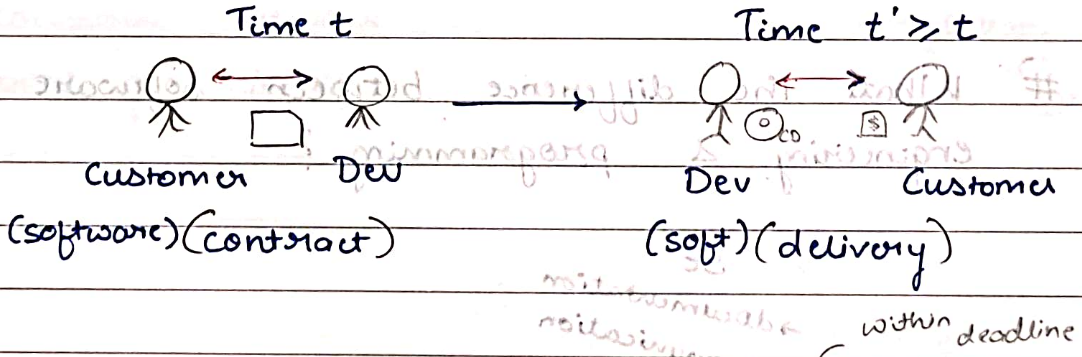
When is software development succesful ?

There are three main stakeholders.



A software development project is succesful if and only if the developer, customer and user are happy with the result at the end of the project.

Which Result ? Which Project ?



If the delivery from both ends happen → Successful

otherwise it's → Unsuccesful.

→ Emperical findings.

Famous project which failed : " Toll- Collect "

Some airport luggage software.

Causes for Unsuccessful Projects

First Approximation

1. Capturing Requirements
2. Design
3. Implementation
4. Quality Assurance
5. Project Management

Cause	Result
1	Then Software won't do what it's supposed to do
2	What's being built is not the soln / just won't work
3	Software will have bugs and have crashes
4	Requirements / Implementation testing will not be don
5	project will be delayed, undergo other issues.

III EXCURSION : Informal v/s Formal

Eg : Requirements : Airbag Controller

Req Specification : Informal : Whenever a crash is detected, the airbag has to be fired within (±300ms) (± ε)

Formal :

Fix Observables : crashdetected : Time → {0, 1}
fire airbag : Time → {0, 1}

Formalize Requirement :

$$\forall t, t' \in Time \cdot crashdetected(t) \wedge fireairbag(t') \Rightarrow$$

$$t' \in [t + 300 - \varepsilon, t + 300 + \varepsilon]$$

Benefits

→ No more misunderstandings, sometimes tools can objectively deciede : requirement satisfied yes/no.