2. Software Development Process Models

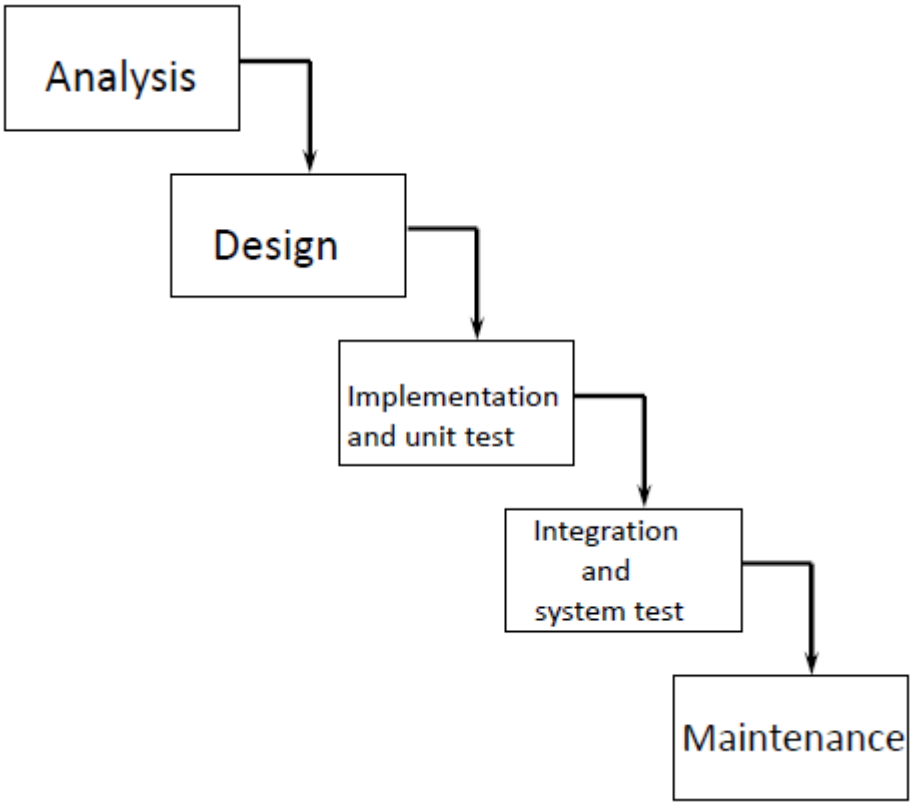
SW is not tangible; to manage, the project manager needs special methods; the **monitoring** is based on the explicit definition of activities to be performed and documents to be produced, that allow to monitor the evolution.

Models differ from each other for activities and documents produced.

The good compromise – Produce documentation useful for the project

2.1 Generic software process models

The waterfall process



- 5 phases:

1) Reqs analysis and definition

2) System and SW design

3) Implementation and unit testing

4) Integration and system testing

5) Operation and maintenance

Each phase has to be completed before jumping to the next.

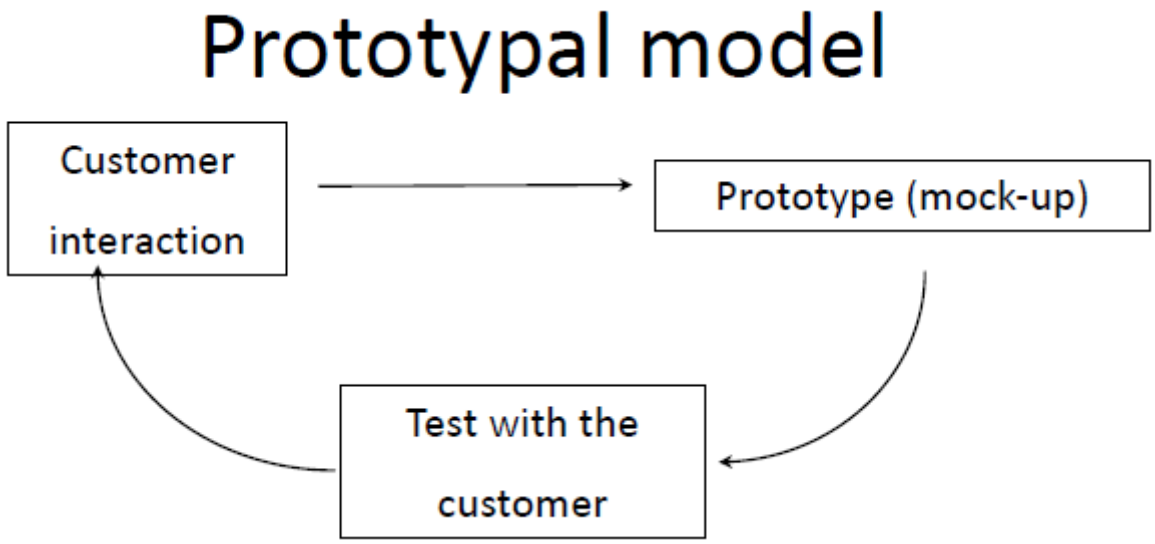
P**roblems**: difficulty of accommodating changes after the process is underway; initial uncertainty: the end user has not a clear view of what the product is(has to wait till a working version, programmers have to wait the end of the analysis for starting to work)

Reqs always evolve, so process iteration where earlier stages are reworked is always part of the process

two approaches:

2.1.1 Incremental delivery

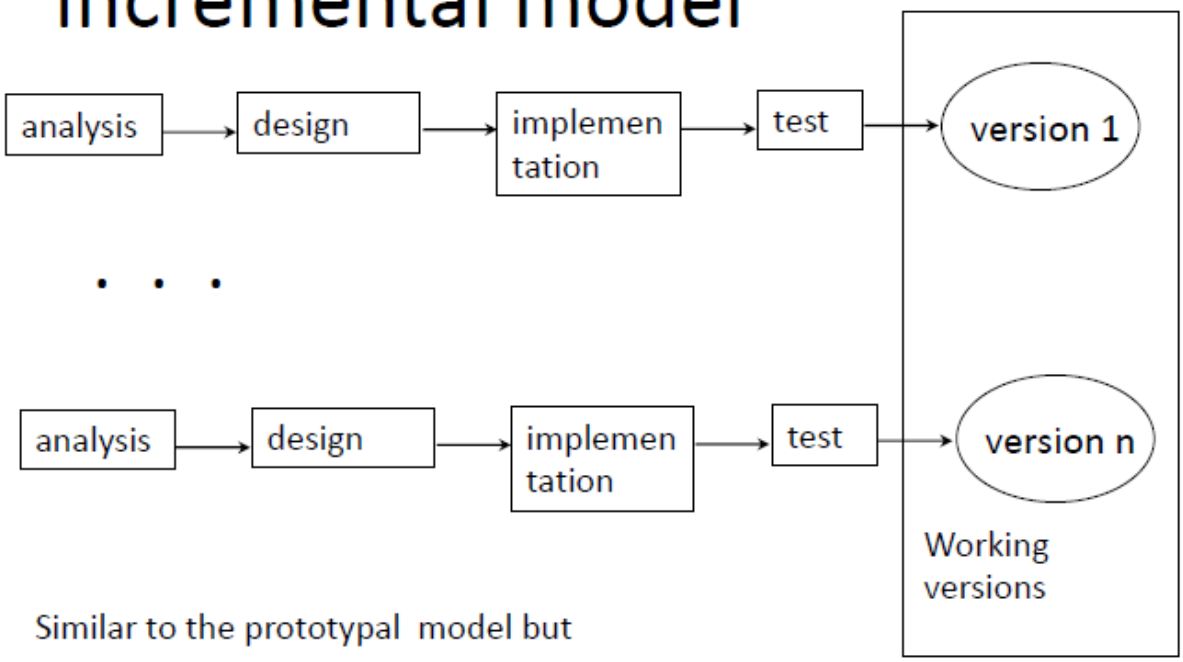
- Prototypal model



the prototype implements only some basic functions (UI), obj → collecting and disambiguating reqs involving the customer

- Incremental model

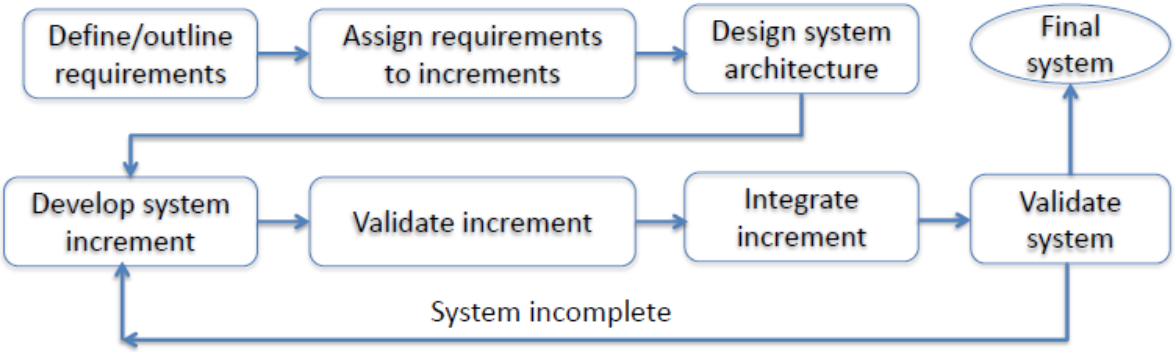
Incremental model



Similar to the prototypal model but

- Intermediate versions are full working
- It allows for a more accurate design

Incremental development



Deliver part of the required functs. ; user reqs are prioritised, once the dev of an incr is started, the reqs are frozen for later.

- advantages:
  - the sw can be delivered after each increment so new functionalities are available earlier.
  - early increments work as a prototype

- lower risk of overall project failure (cause we test every new functionality, not the overall)
- the highest priority, the higher testing period

2.1.1.1 Formal methods

Use of algebric formalisms for reqs specification, dev and test

(for every, &&, ||, etc.)

2.1.1.2 Extreme programming

Part of the agile model family

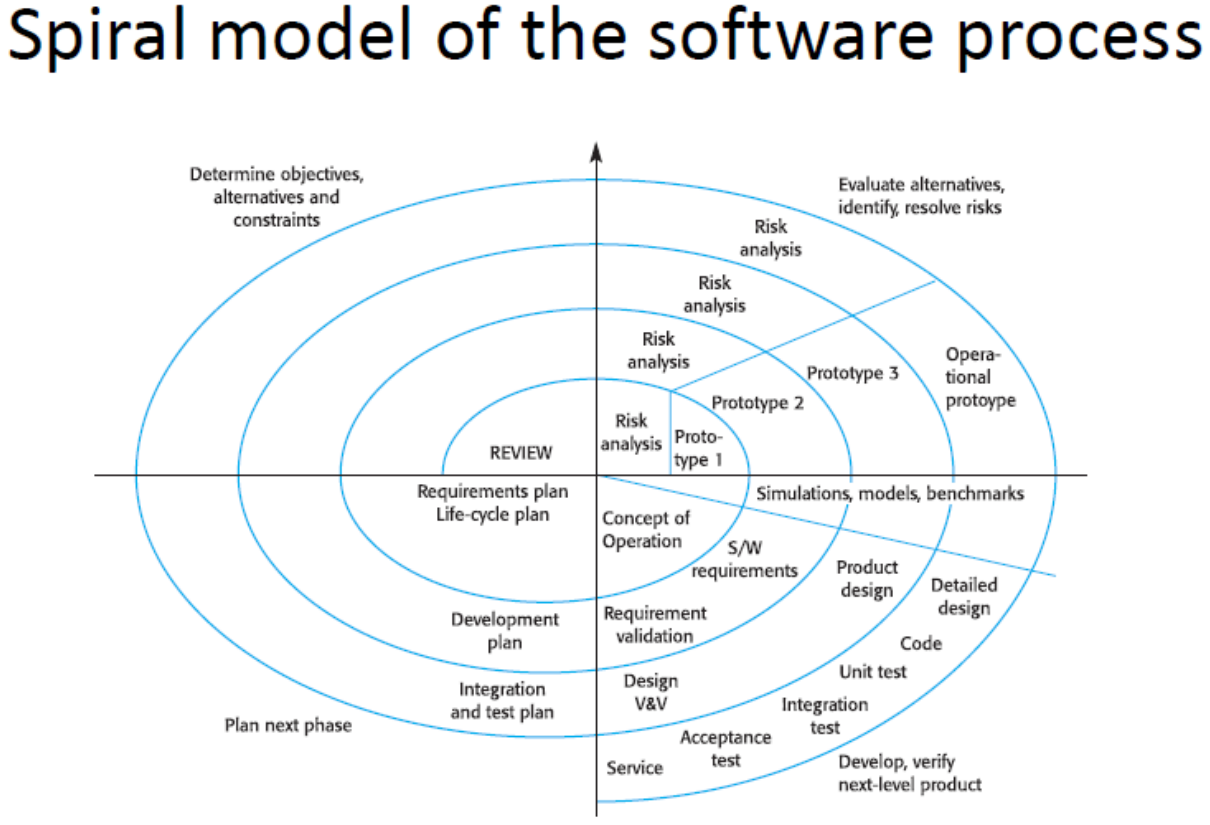
approach based on very small increments, constant code improvement and user involvement

2.1.2 Spiral Development

Each loop of the spiral represents a phase

no fixed phases such as specification or design

risks are assessed and resolved during the process



- sectors:

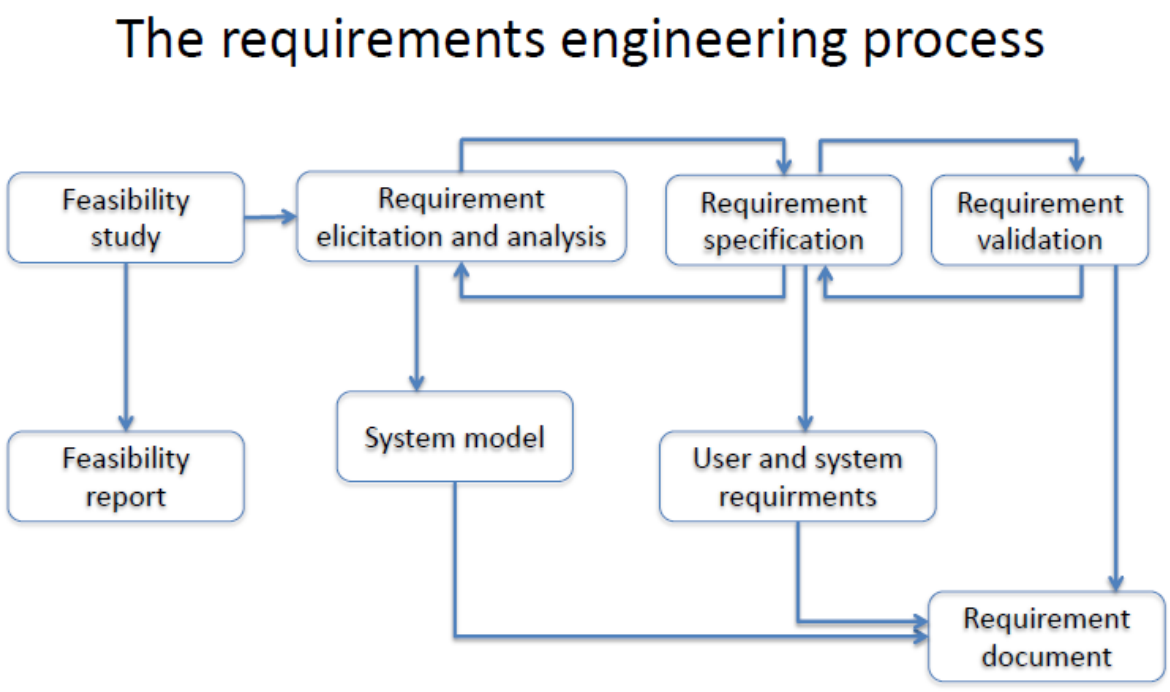
- objective setting
- risk assessment and reduction
- dev and validation
- planning next phase

2.2 Core process activities

2.2.1 SW Specification

Establish what services are required: functional reqs, non-functional reqs

- reqs engineering process
  - feasibility study
  - reqs elicitation and analysis
  - reqs specification
  - reqs validation



2.2.2 SW Design and implementation

Converting the system specification into an executable system

- design a sw structure that realises the spec:
  - architecture design
  - interface
  - component
  - data structure
  - algorithm

  design is usually documented as a set of models (UML)
- Implement an executable program from the design, there is no generic programming process, it is a personal activity

  test to discover faults, then debug:
  - locate errors
  - design error repair
  - repair
  - re-test

these 2 are related and may be inter-leaved

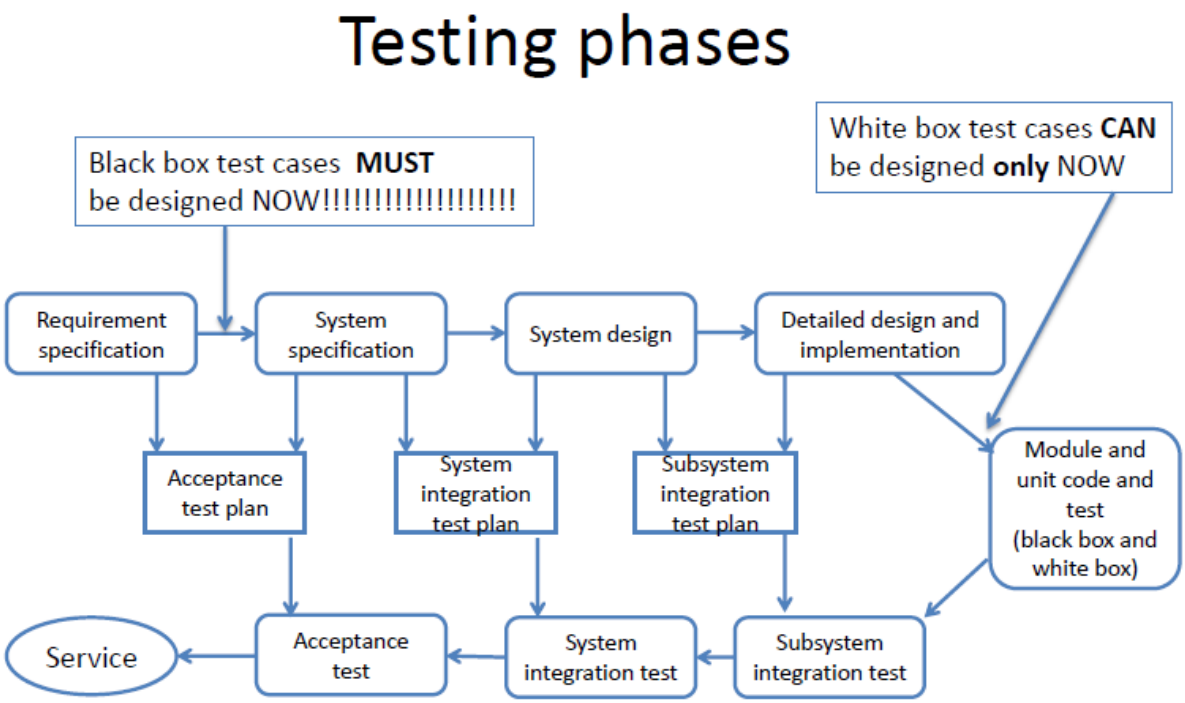
2.2.3 SW Validation

V & V is intended to show that a system: conforms to its specification (verification) and meet reqs (validation)

involve checking & review, and testing(involves executing the system with test cases)

Testing:

- Component testing (every component tested indipendently)
- System testing (test of the system as a whole)
- Acceptance testing (test with customer data to check that meets reqs)



2.2.4 SW Evolution

SW is flexible and can change,

as reqs, sw also evolves to support businnes needs.