SOFTWARE SPECIFICATION, DEVELOPMENT AND EVOLUTION

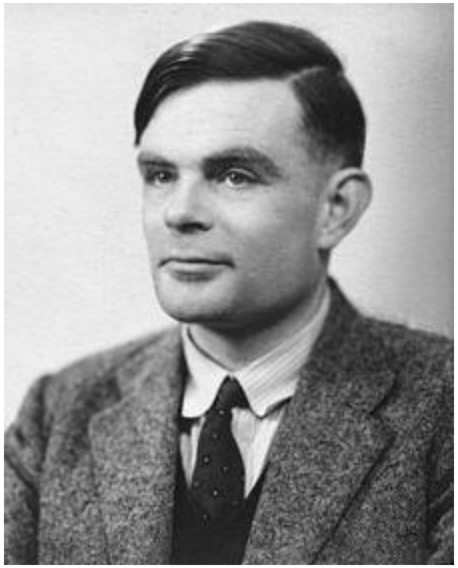
Matthieu Tixier – #1

Plan

- The roots of software engineering
  - Computer Science vs Software Engineering
  - Engineering stakes
    - Maintenable, Dependable, Efficient, Acceptable
    - Essential vs Accidental complexity [Brooks, 1987]
- The main software engineering process
  - Key steps [Sommerville, 2009]
  - Several approaches
    - Waterfall
    - Cycle en V
    - Conception itératives
    - Méthodes agiles
- A software projects typology

Historical perspective

- A dual origin of software engineering
  - **Computer Science :** The theories and models that ground the structure and process of computers and software develpment (ie, alogrithm complexity, information theory, decision problem…)

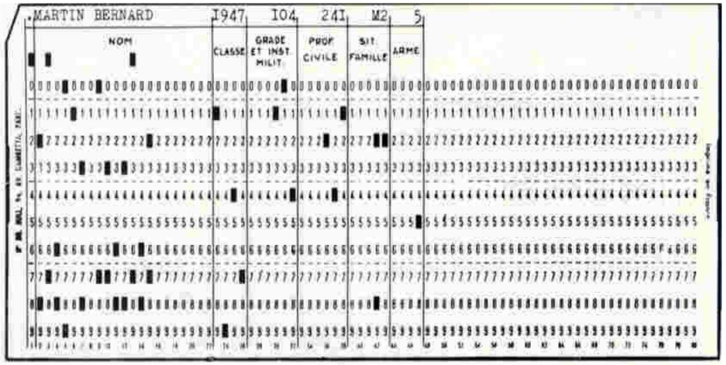


Alan Turing (1912-1954)
http://en.wikipedia.org/wiki/Alan_Turing

- Alan Turing (1912-1954)
- Mathematician, Logician, Philosopher
- First ideas of a universal computer (the Turing machine, 1936) and Artificial Intelligence

Historical perspective

- A dual origin of software engineering
  - Unit record equipment (Mécanographie) 1880’s → 1970’s
    - Early data processing needs (ie, census,vote)
      - Punch machine (Poinçonneuse)
      - Tabulating machine (Tabulatrice)
      - Collators (Interclasseuse)
      - Sorting machine (Trieuse)



A punch card (80 columns)

http://www.feb-patrimoine.com

A company mecanographic setup (1960’s)

© G. Natan – Bull Museum

Historical perspective

- First applications
  - Military: cryptography, trajectories computing, physical simulation
  - Management: census, statistics, document indexation
  - Scientific: simulation, complex comuting

    ➔ increasing productivity
- Convergence
  - IBM ➔ International Business Machine
    - With computer succeed to gather processes that were formerly distributed accross different electromechanical machines.

What is software?

- Software (Logiciel) :

A set of computer programs that aims to support general or specific functions (related to an organizational context) as well as the associated documentation that allows its use, maintenance and evolution..

  - Ex : Word processing software, planing management,…

- A system engineering perspective
  - A set of components

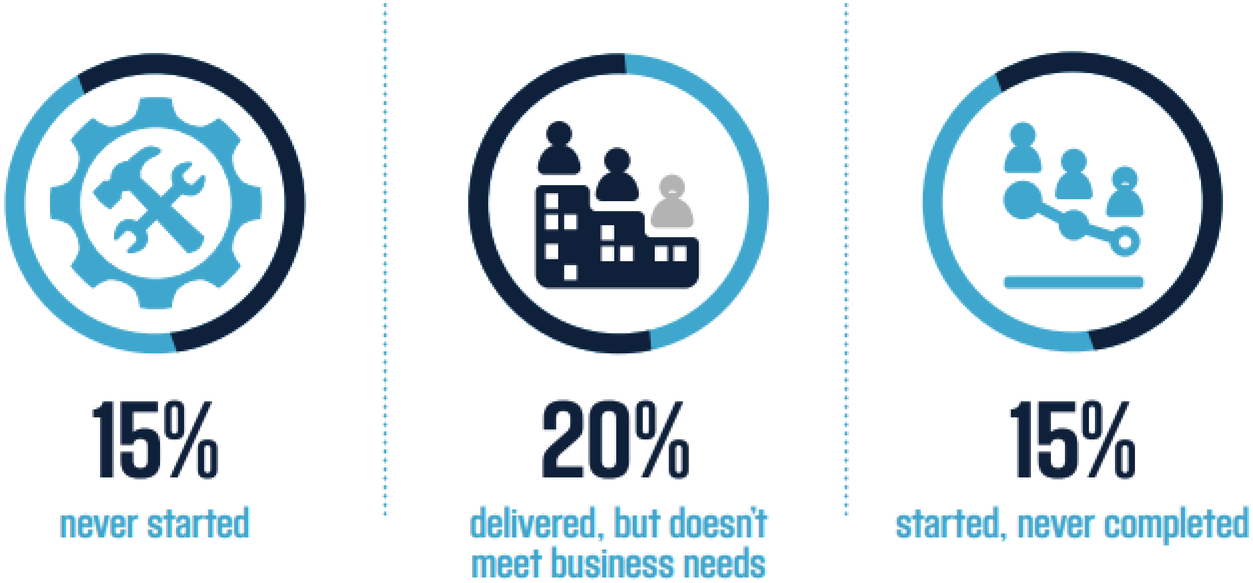
The « Software crisis »

- An increasing complexity of applications
  - Adoption in companies
  - Despite more reliable hardware
- The term 'software engineering' was suggested at conferences organized by NATO in 1968 and 1969 to discuss the 'software crisis'.
- An intrinsic complexity?
  - Essential vs Accidental complexity
    - « No Silver Bullet – Essence and Accident in SE » [Brooks, 1987]
    - Accidental complexity:

      - Can be solved by technical improvements (ie, performance, optimization issues)
    - Essential complexity:
      - Integration: 30 functions requested are 30 functions to develop
      - Heterogenity: software as a tool (or toolbox) integrated in a larger environment (ie, competitors, market, rules)
      - Work and society evolution: the societal needs, technologies adoption (ie, smartphones)

The « Software crisis »

- Does the situation improved much?
  - **50% of software project still fails in 2018**



- source : IDC/Appian

Engineering stakes

- Scales of software projects by example :
  - **Firefox**: 20 548 086 lines of code (C++)
    produced by 7 399 contributors since 1998
    (ie, 500 contributions by month)
  - **NetBean IDE**: 95 335 619 lines of code (Java)
    produced by 1077 contributors 1999
    (ie, 30 contributions by month)
  - **LibreOffice**: 9 526 750 lines of code (C++)
    produced by 1871 contributors since 2000
    - (ie, 80 contributions by month)
  - **Jquery**: 38 489 lines of code (Javascript)
    - produced by 369 contributors since 2006
      (ie, 10 contributions by month)

- source : https://www.openhub.net

Engineering stakes

- The quality of a « good » software
  - Maintenable
    - Allows error correction and evolution
  - Sûreté de fonctionnement (Dependability)
    - Disponibilité (Availability) : ability to provide the service
    - Fiabilité (Reliability) : ability to provide the expected service
    - Sûreté (Safety) : potential threats to goods and persons
    - Sécurité (Security) : system resistance to hack and intrusions
  - Efficient (Efficiency)
    - The balance between performance and resources consumption (memory, processor, network)
  - Acceptable
    - Adapted to end-users (understandable, usable and interoperable)

The software engineering process

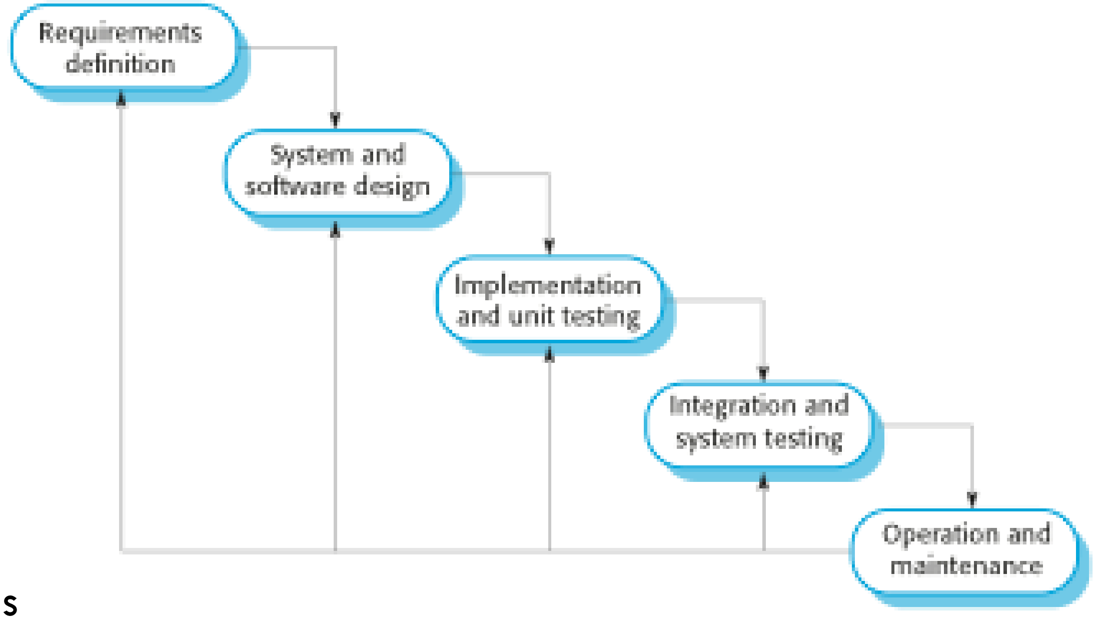
- Fundamental steps
  - Software specification
    - Joint definition of the product requirements and operational constraints between the client and software engineers
  - Software development
    - Design and development
  - Software evaluation
    - Check the conformity with specifications
  - Software evolution
    - Change to the software in order to satisfy new needs and to provide answer to the organization and market evolutions

The software engineering process

- Several models
  - Waterfall
  - V cycles
  - Incremental development
  - Agiles methods
- Benefits
  - A map, a shared awareness of the project status
- Limits - What they do not always say.
  - Roles ➔ Who is responsible for what in the project ?
  - What are the precise activities to do in order to meet the project objectives and go to the next step?

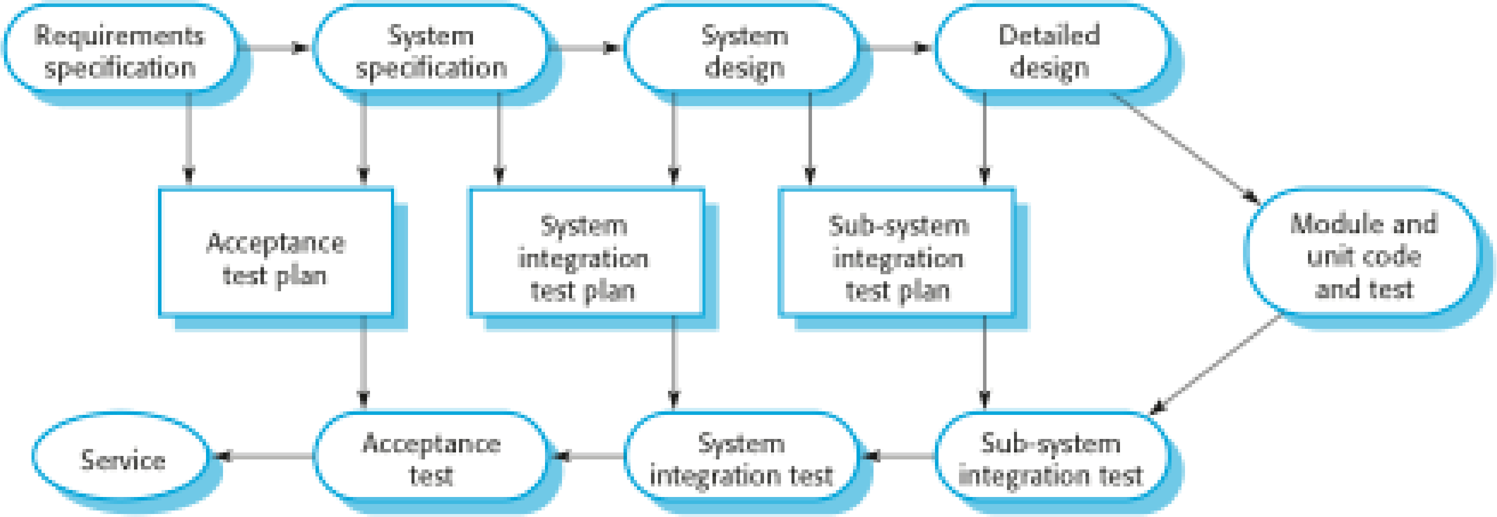
Waterfall - En cascade

- Waterfall (Royce, 1970)
  - A system engineering perspective
  - A set of steps with defined delivrables
  - Linear development
- Benefits
  - Ease project management (costs and human resources)
  - Efficient when the needs are clear
- Limits
  - Theory vs practice : overflow between the steps
  - Lack of flexibility


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

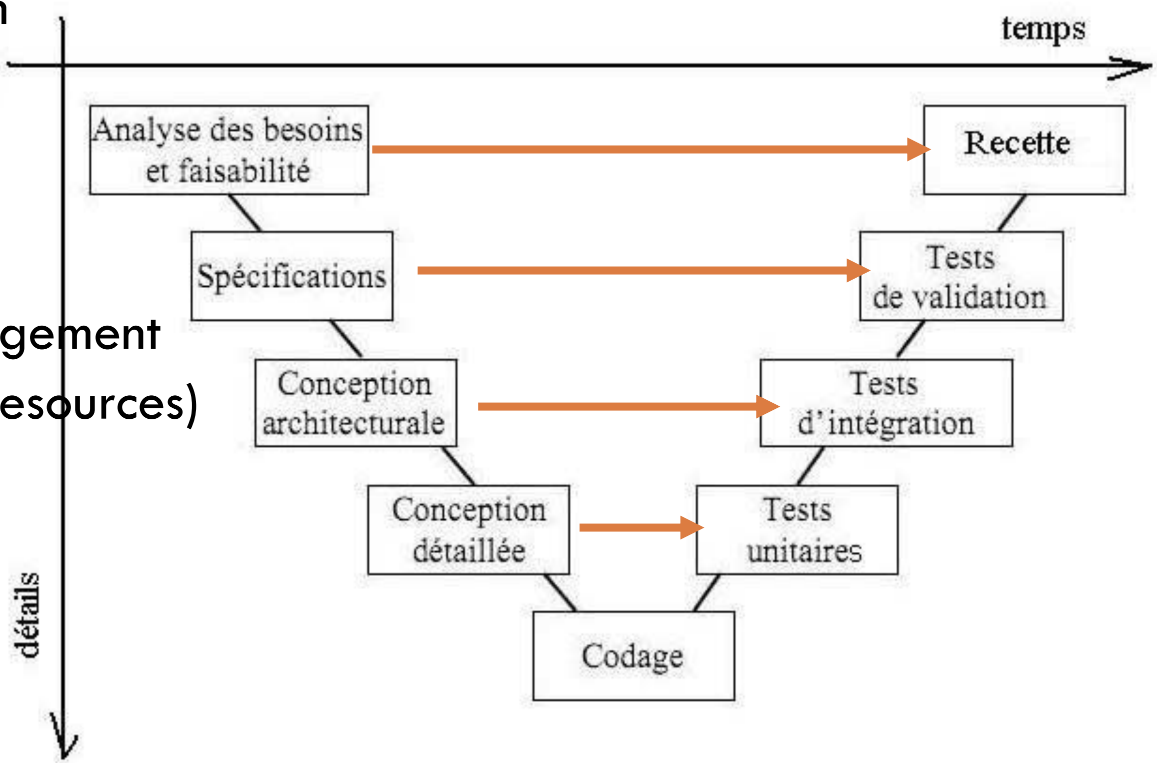
V cycles - Cycle en V

- □ V cycles


Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and test
Service
Acceptance test
System integration test
Sub-system integration test

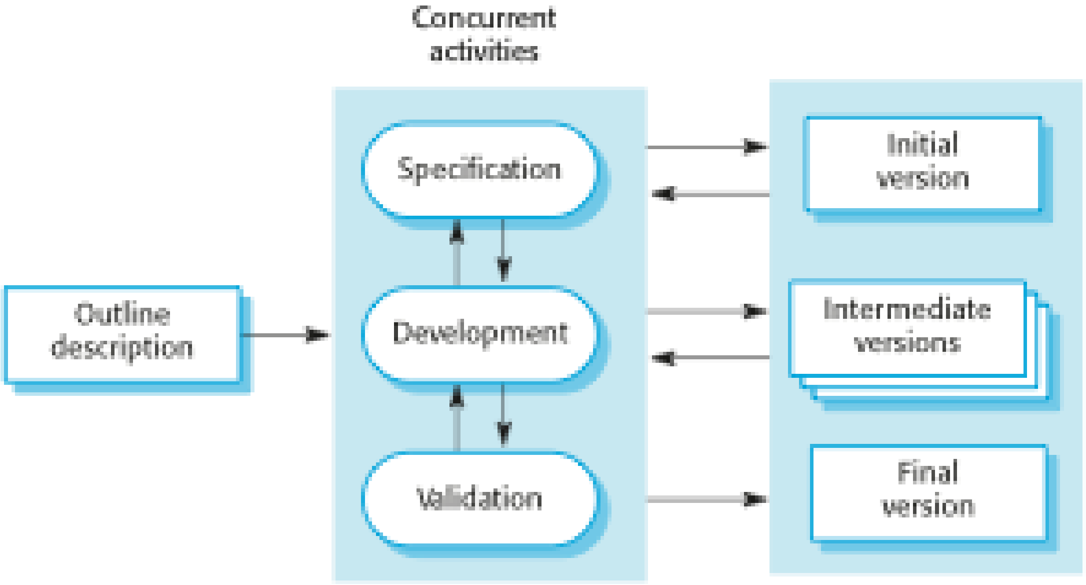
V cycles - Cycle en V

- V cycles
  - Structured evaluation
  - Good for evolution
- Benefits
  - Emphasize on software quality (several test plan)
  - Ease project management
  - (costs and human resources)
- Limits
  - Iteration duration is long
  - Lack of flexibility



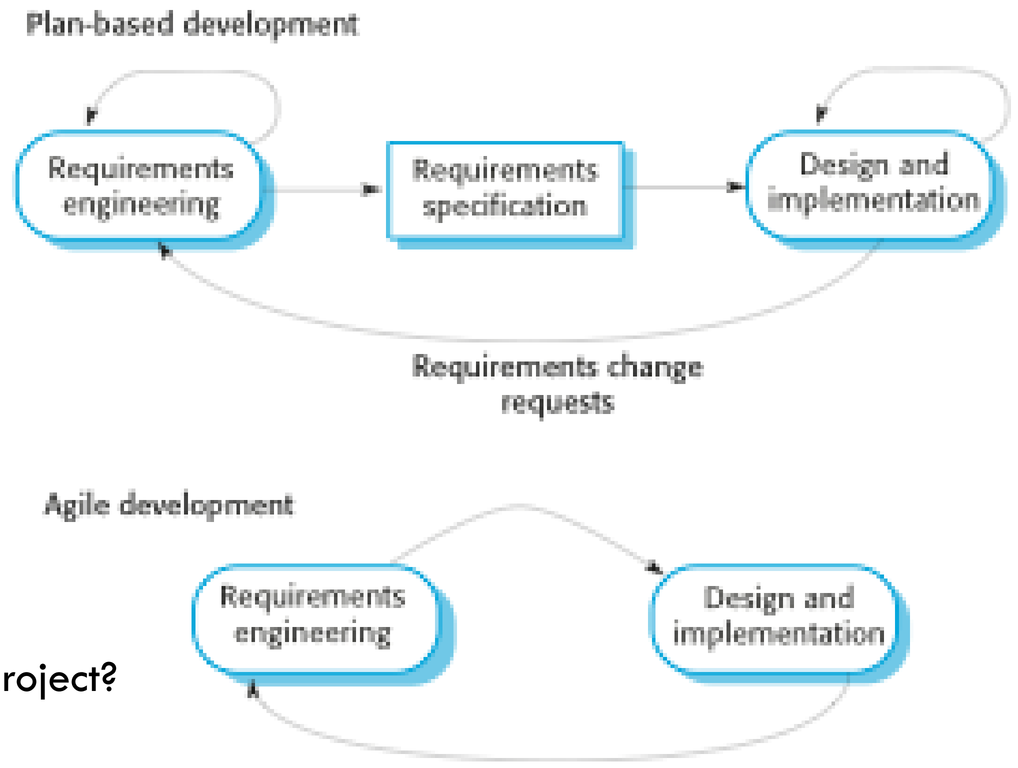
Incremental development

- Iterative cycles
  - Address the interdependancies between activities
  - Mid-term delivery
  - Concurrency
- Benefits
  - Follow the evolution of client needs
- Limits
  - Increased complexity of project and costs management



Agile ?

- Agile Manifesto (2001)
  - Users needs and requirement evolution as essential complexity
  - Involving the client in the project mangement (priority definition, deliverable evaluation)
  - Incremental delivery of working features
  - « People not process »
  - « Keep it simple »
- Benefits
  - Tailored to client needs
  - Pragmatism
- Limits
  - Ability to scale with important project?
  - Client and team involvment


Plan-based development
Requirements engineering
Requirements specification
Design and implementation
Requirements change requests
Agile development
Requirements engineering
Design and implementation

To sum up

- Specification/Development/Evaluation/Evolution
  - Different ways to organize the software project activities
- Plan oriented process
  - Waterfall
  - V cycles
- Need oriented process
  - Incremental development
  - Agile methods

➔ Mastery vs Adaptation

A software project typology (1/2)

- A basic typology [Sommerville, 2009] :
  - **« Stand-alone application » (client lourd) :** Application systems that run on a personal computer or apps that run on a mobile device. They include all necessary functionality and may not need to be connected to a network. (ex. CAO, suite bureautique)
  - **« Interactive transaction based application » (client léger) :** Software executed on a remote computer and that are accessed by users from their own terminal. (ex. mainframe et web)
  - **Embedded control system (Systèmes de contrôle embarqués) :** Software control systems that control and manage hardware devices. Hardware constraints are often critical. (ex : anti-lock braking in a car and software in a microwave oven to control the cooking process)
  - **Batch system (Système de traitement par lots) :** Systems that are designed to process data in large batches. They process large numbers of individual inputs to create corresponding outputs. (ex : pay roll system, periodic billing system)

A software project typology (2/2)

- A basic typology [Sommerville, 2009] :
  - **Entertainment systems:** Systems for personal use that are intended to entertain the users. The quality of the user interaction offered is the most important distinguishing characteristic of entertainment systems.

  - **Système de modélisation et simulation:** These are systems that are developed by scientists and engineers to model physical. These are often computationally intensive and require high-performance parallel systems for execution.

  - **Data collection and analysis systems :** Systems that collect data from their environment and send that data to other systems for processing. The software may have to interact with sensors and often is installed in a hostile environment such as inside an engine or in a remote location. ‘Big data’ analysis may involve cloud-based systems carrying out statistical analysis and looking for relationships in the collected data.

  - **System of systems:** These are systems, used in enterprises and other large organizations thatare composed of a number of other software systems. Some of these may be generic software products, such as an ERP system. (ex : Information system)

Ouverture

- Software as a complex object
  - Internet
  - Web services
  - Cloud computing
  - Smartphones
  - Internet of things
  - […]

« Nuggets »

- What is software?
- The quality of « good » software?
- The main activities in a software project?

References - Bibliographie

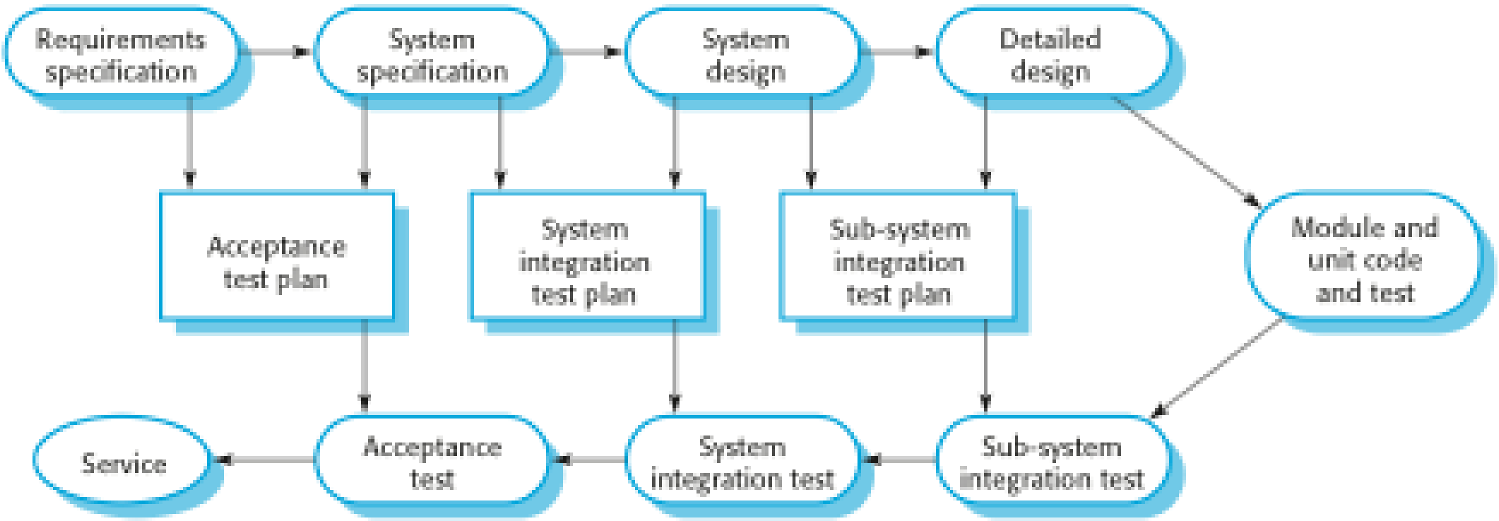
- Thanks for your attention
  - Question(s) ?

- I. Sommerville, *Software Engineering*. Pearson Education, 2009.
- F. Brooks, *No Silver Bullet — Essence and Accident in Software Engineering*, Proceedings of the IFIP Tenth World Computing Conference, 1986.
- http://agilemanifesto.org/
- http://www.computerhistory.org
- http://www.histoireinform.com/
- *https://www.developpez.com/actu/228268/Etude-50-pourcent-des-projets-de-developpement-d-applications-se-soldent-par-un-echec-cela-est-il-du-a-la-lenteur-des-codeurs-et-la-dette-technique/*

Annexes

Cycle en V

- Présentation alternative


Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and test
Service
Acceptance test
System integration test
Sub-system integration test