UNIVERSITÀ DEGLI STUDI DI TERAMO

COMPUTER SCIENCE THEORIES AND TECHNOLOGIES

Software Engineering

Prof. ssa Romina Eramo
Università degli Studi di Teramo
Dipartimento di Scienze della Comunicazione
reramo@unite.it

Introduction

- **Engineering** is the use of scientific principles to design and build machines, structures, and other items, including bridges, tunnels, roads, vehicles, and buildings.
- The discipline of engineering encompasses a broad range of more specialized fields of engineering, each with a more specific emphasis on particular areas of applied mathematics, applied science, and types of application.

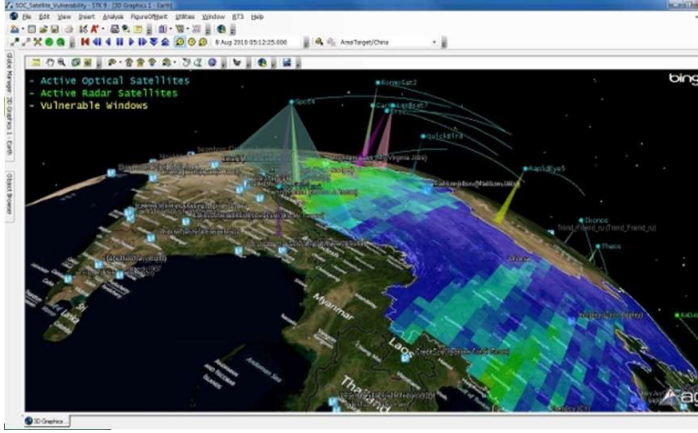
Software Engineering (working def.)

Set of automated methods

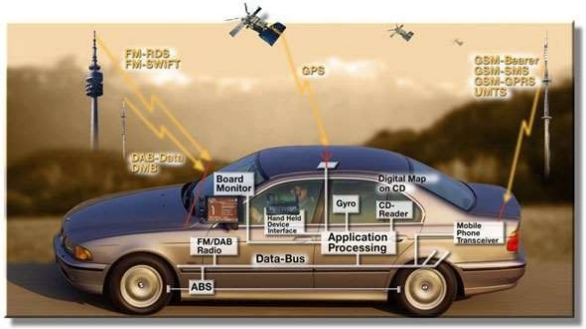
to systematically develop quality software

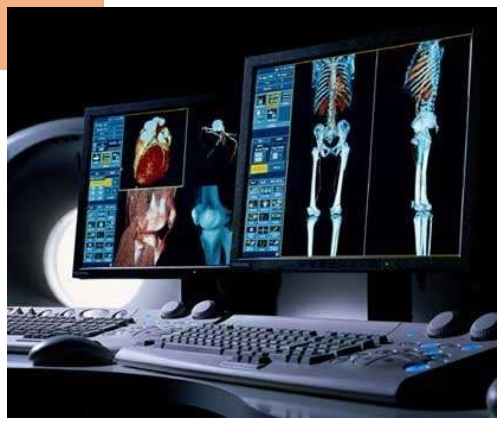
that fulfils customer needs

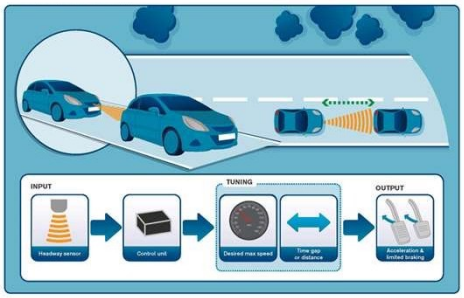
while satisfying existing constraints



Let us put the pieces together, on…







A software system is not only software, not only programming

≠

Programming is NOT enough!

It is not enough to do your best: you must
Know what to do, and THEN do your best.
-- *W. Edwards Deming*

Software Engineering (working def.)

Set of automated methods

to systematically develop quality software

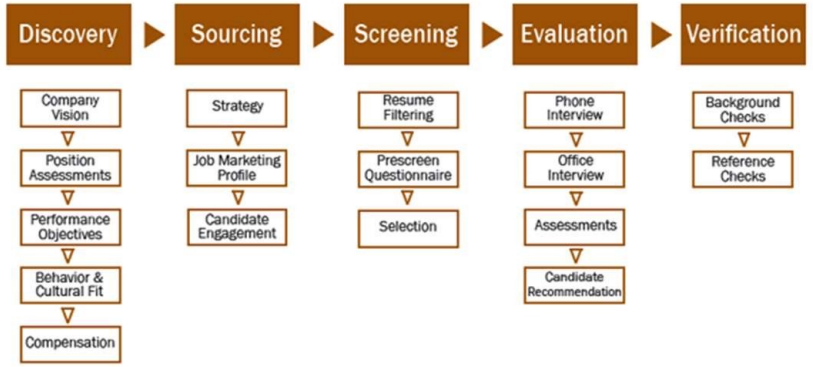
that fulfils customer needs

while satisfying existing constraints

Software Engineering: What

Automated methods

Systematic



Set of automated methods

to systematically develop quality software

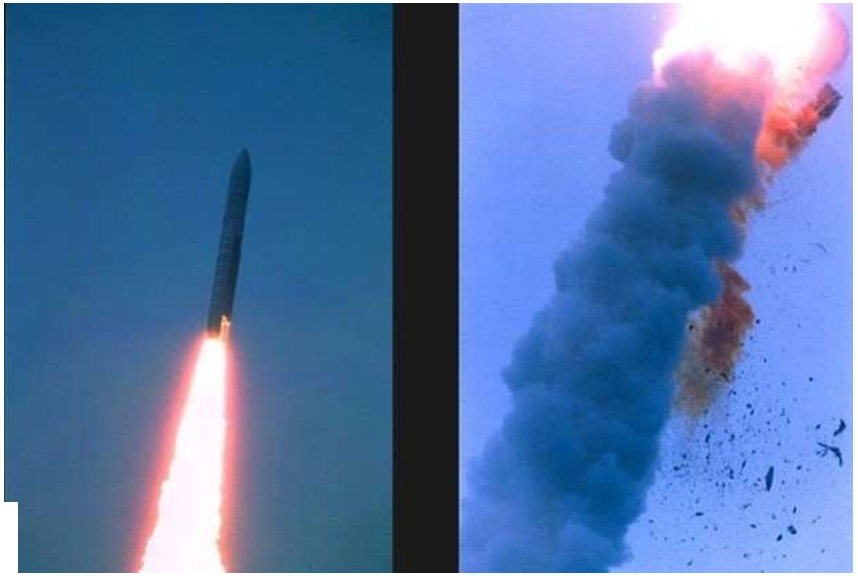
that fulfils customer needs

while satisfying existing constraints

Software Engineering: What

- Quality Software..

- ... to mitigate failures



Set of automated methods

to systematically develop quality software

that fulfils customer needs

while satisfying existing constraints

Software Engineering : What

- **Customer needs**

Set of automated methods
to systematically develop quality software
that fulfils customer needs
while satisfying existing constraints

Software Engineering: What

- **constraints**

Set of automated methods
to systematically develop quality software
that fulfils customer needs
while satisfying existing constraints

Well Engineered Software

» It does what expected by the customer/user +
» It implements the «wanted» qualities +
» It satisfies existing constrains +
» It can be easily revised, extended, evolved …
» … and more!

Set of automated methods
to systematically develop quality software
that fulfils customer needs
while satisfying existing constraints

Software Engineering: When

» Field of computer science dealing with software systems that are:
  - large and complex =complex? large?
  - built by teams = people!, communication, ...
  - exist in many versions =version control
  - last many years =engineered to be sustainable
  - Undergo changes =evolves

Major points

1. Why it is important to design
2. Definition of Software Engineering
3. Principles that guide design
4. Collaboration and soft skills and complementarity
5. Common problems and causes of failure
6. Importance of user experience
7. Difference between engineering and SW engineering
8. The importance of the term "systematic"
9. Importance of documentation
10. Personal experience/encounters with the reality

Software development process

» What is a process?

- A process defines who does what, when, and how to achieve a given goal
- In SW engineering, the goal is to produce SW products or improve existing ones

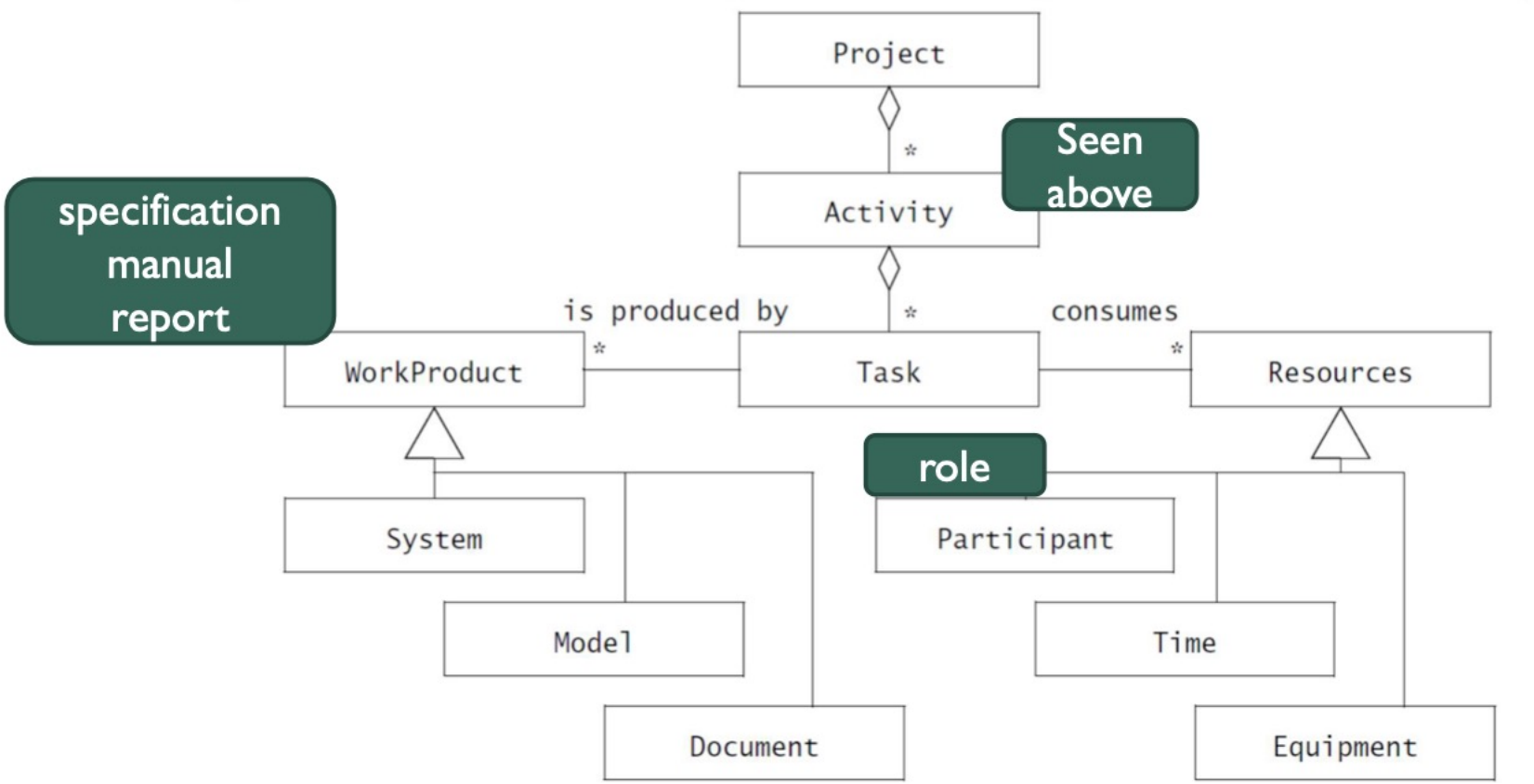


Figure 1-1 Software engineering concepts depicted as a UML class diagram [OMG, 2009].

An effective process……

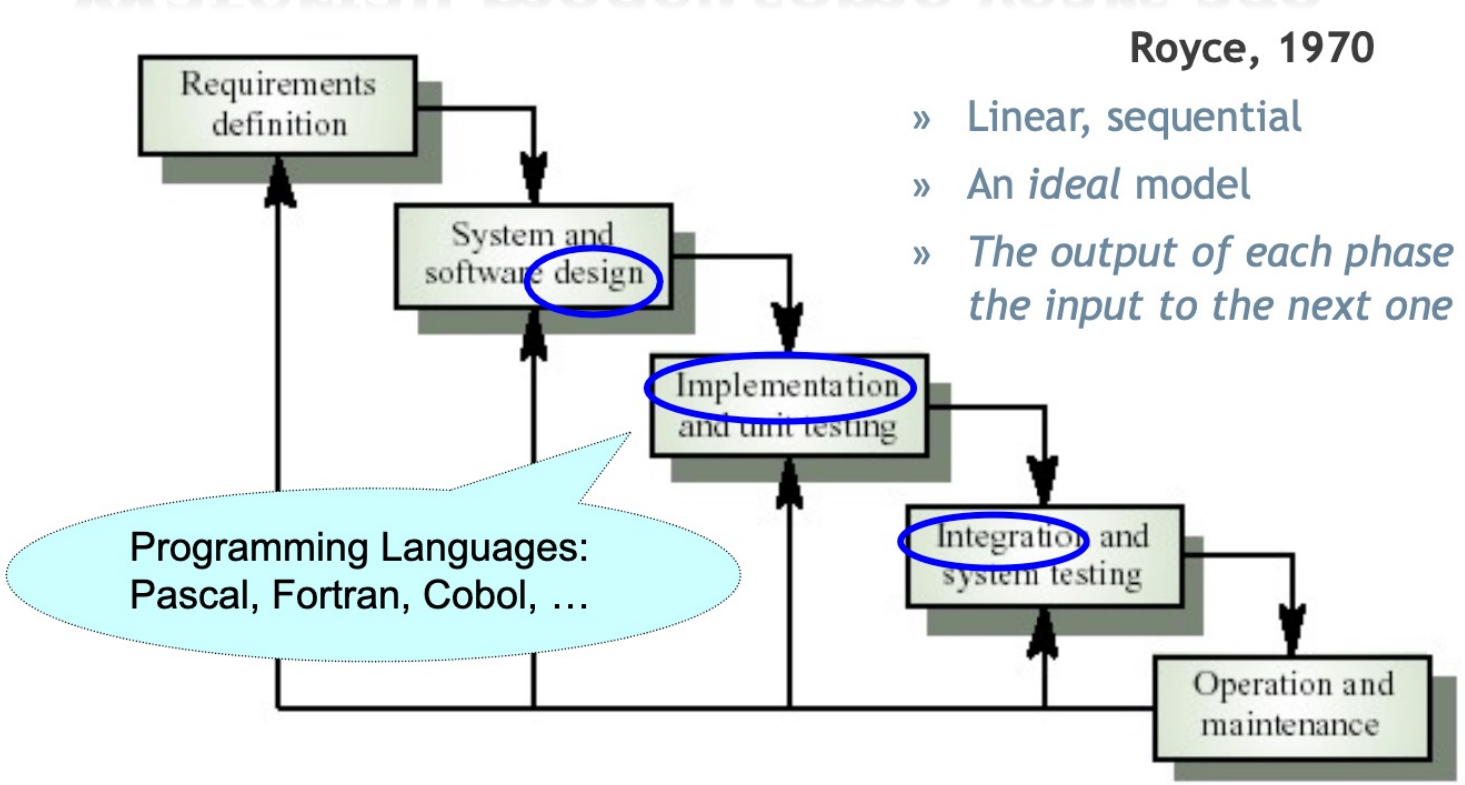
» Provides guidance on the sequence of activities within a working group

» Specifies which deliverable is to be developed and the appropriate time to do so

» Guides the tasks of individual developers and the entire working group

» Provides criteria for controlling and measuring project deliverables and activities

Waterfall model: some years ago…



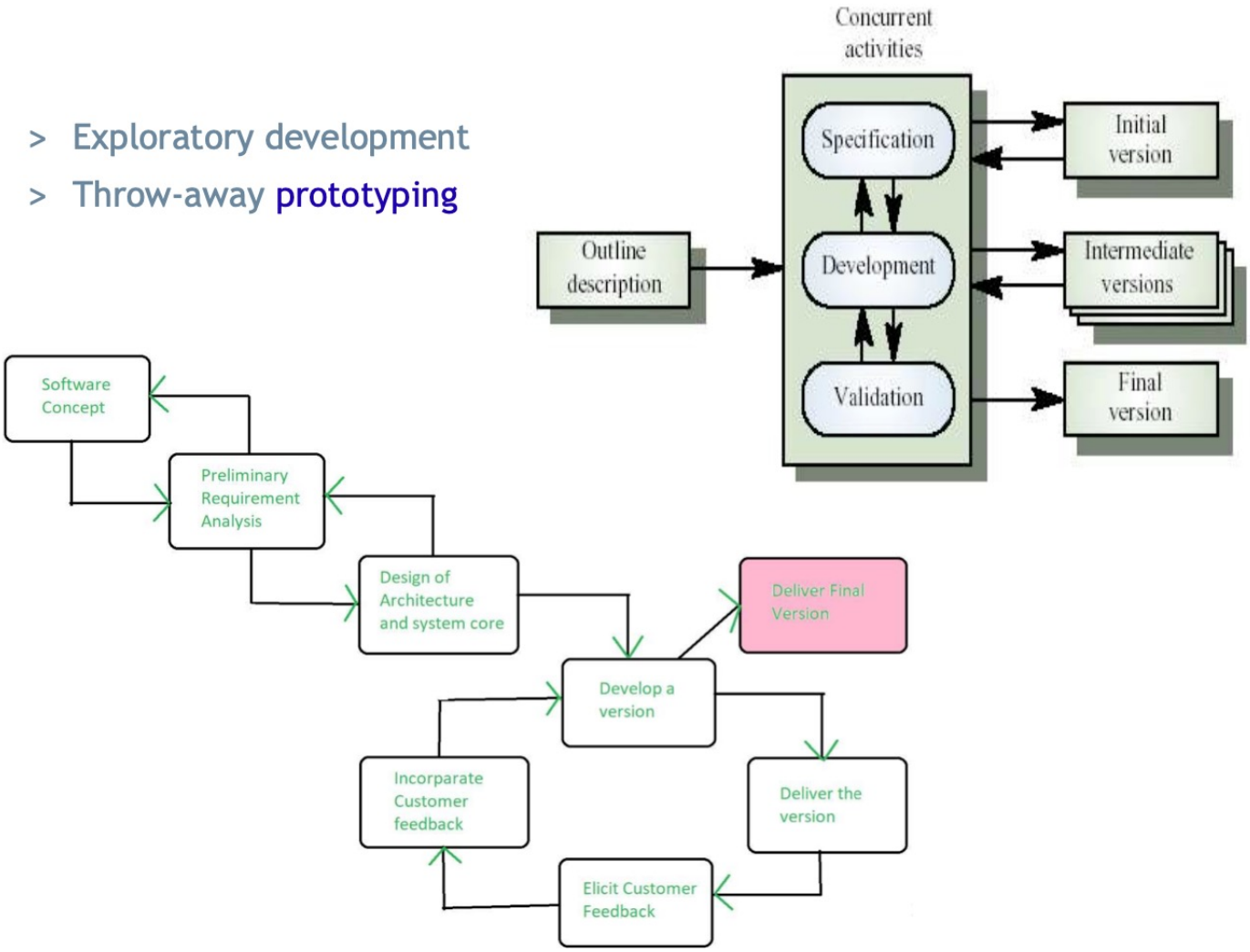
Royce, 1970

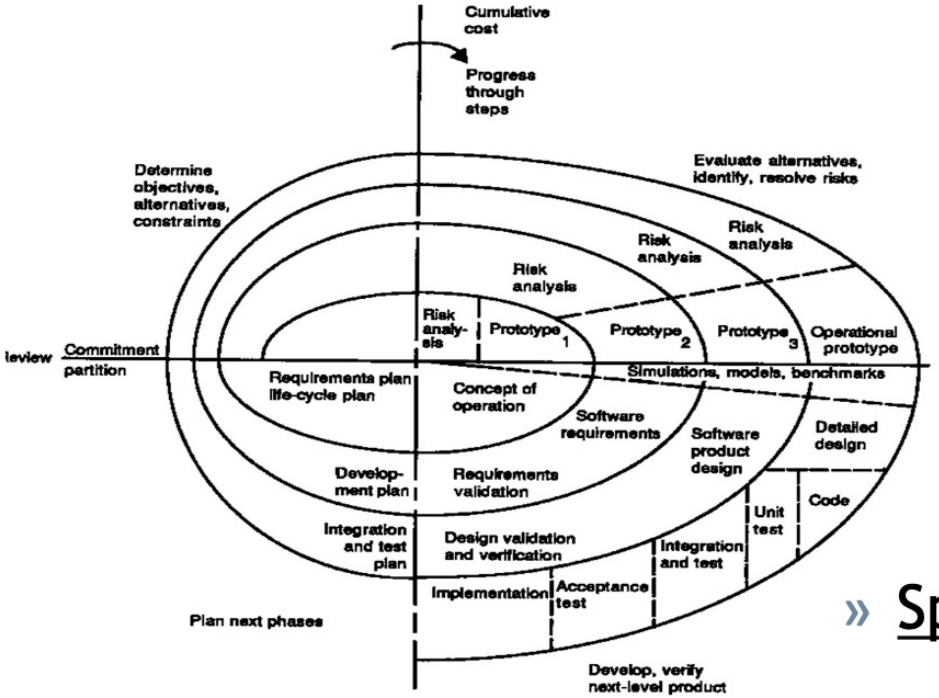
- Linear, sequential
- An *ideal* model
- *The output of each phase is the input to the next one*

[Figure from Ian Sommerville]

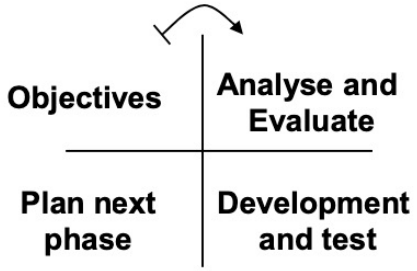
Evoutionary processes

- Exploratory development
- Throw-away prototyping

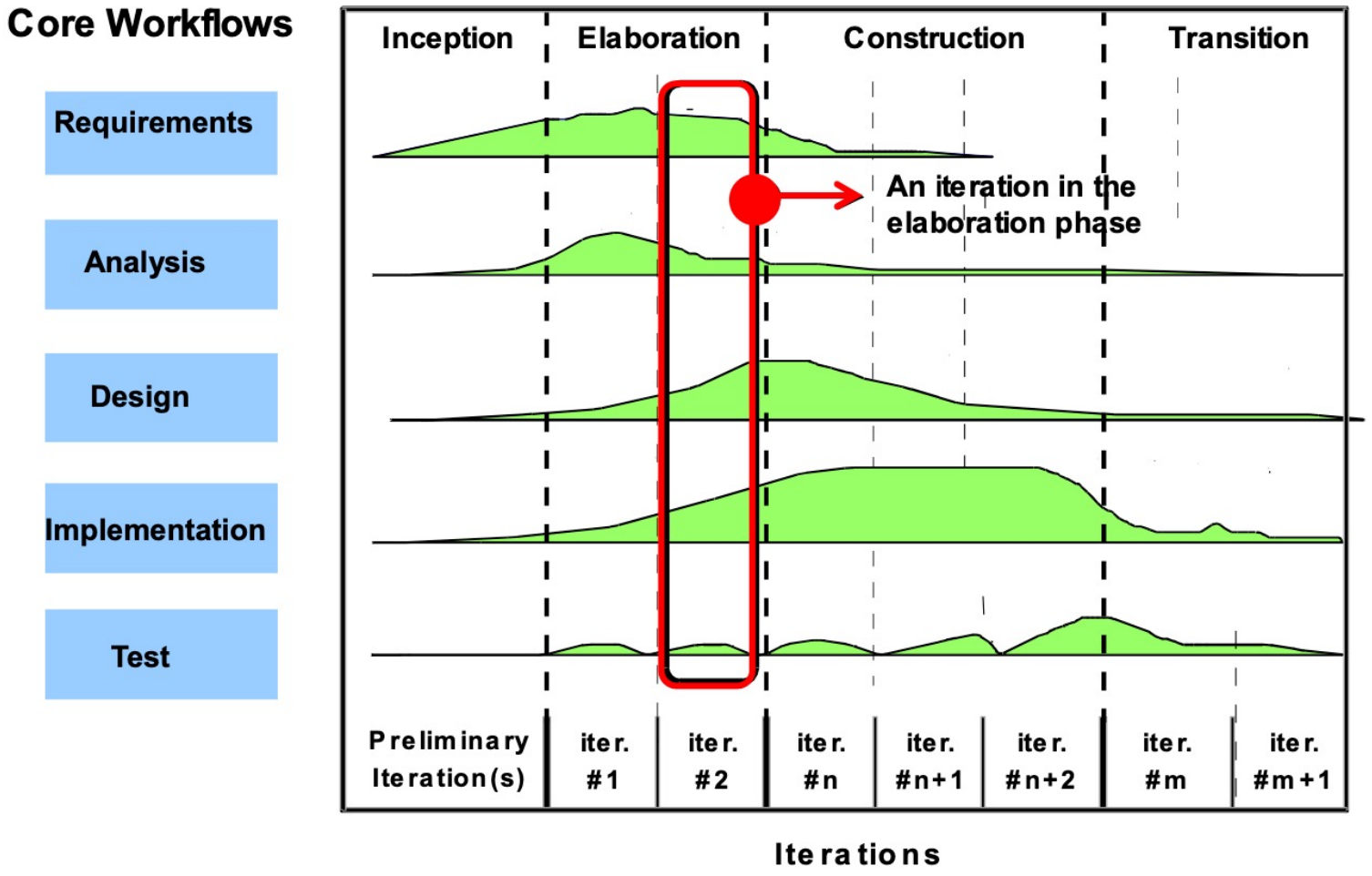




» Spiral, Boehm, 1985



Rational Unified Process (RUP)



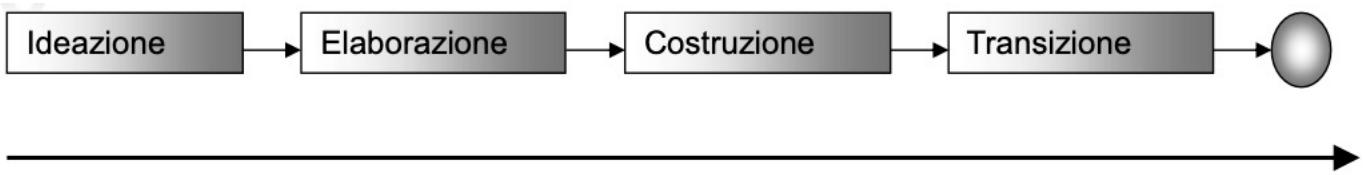
Process structure

- First Dimension (Horizontal Axis)
  - Dynamic process structure
  - Represents time and shows the aspects of the process inherent in its life cycle deployment
  - Dynamic aspect of the system in terms of cycles, phases, iterations and milestones
- Second dimension (Vertical axis)
  - Static structure of the process
  - Represents the main workflows (activities) of the process, which group activities that are logically related by their nature
  - Static aspect of the process in terms of process components, activities, workflows and roles

Dynamic process structure

- **Cascading process** (Requirements, Design, Implementation, and Integration) is a reasonable process as long as.
  - Requirements do not change (it is difficult for requirements to remain the same over time)
  - A complete project can be developed before proceeding does NOT work when projects have a high degree of innovation,uncertainty and risk SOLUTION: ITERATE the cascading process.

Dynamic process structure (2)



- Ideation (Inception):
  - Defines the purpose of the project
- Elaboration (Elaboration):
  - Project planning, feature specification and architectural basis
- Construction (Construction):
  - Builds the product
- Transition (Transition):
  - Transfers the product to users

Static process structure

» Process describes who does what, how and when

» RUP represented using four main modeling elements

- Who: roles
- How: activities
- What: processed → When: workflow

Software Engineering (in practice)

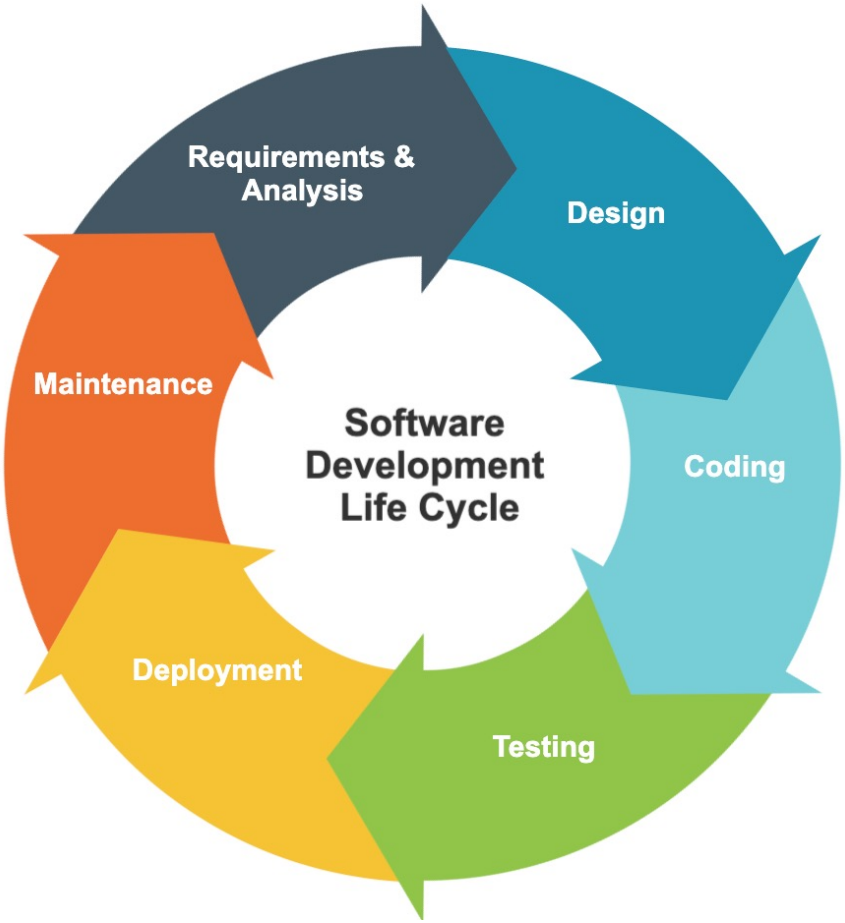
» Software engineering is a systematic engineering approach to software development
  - Focus on design, development, maintenance, testing, and evaluation of software
  - Use of engineering principles and methods to create reliable, efficient, and high-quality software

» Software engineers use programming languages to write and test code, and tools and frameworks to manage the software development process
  - Wide range of software projects, including applications for desktop and mobile devices, operating systems, and embedded systems

Software Development Life Cycle

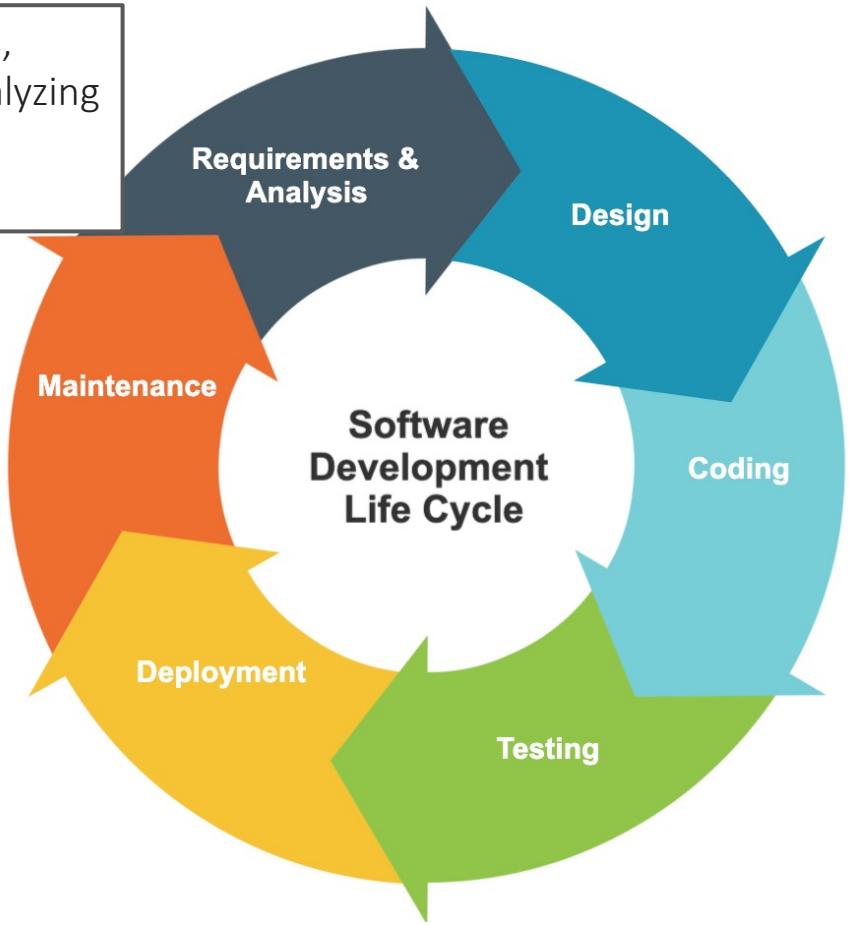
- The ***systems development*** life cycle is a process for planning, creating, testing, and deploying an information system

Software Development Life Cycle



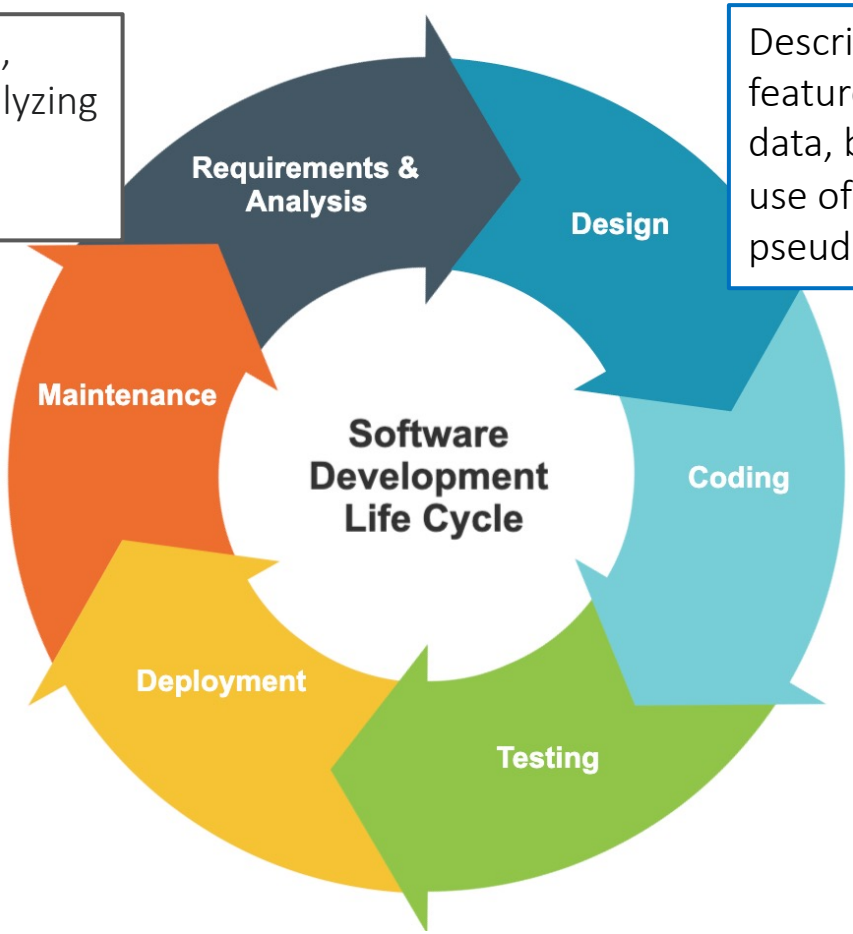
Software Development Life Cycle

Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements



Software Development Life Cycle

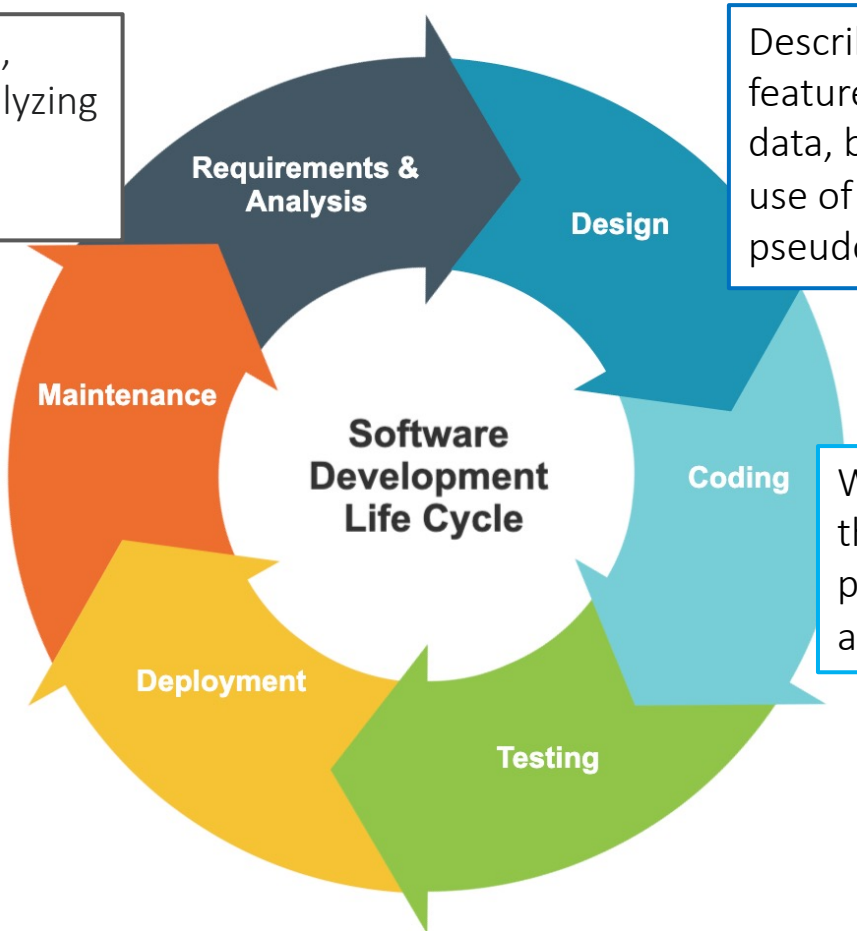
Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements



Describing system components, features, operations, functions, data, behavioural rules with the use of models, diagrams and pseudo-code

Software Development Life Cycle

Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements

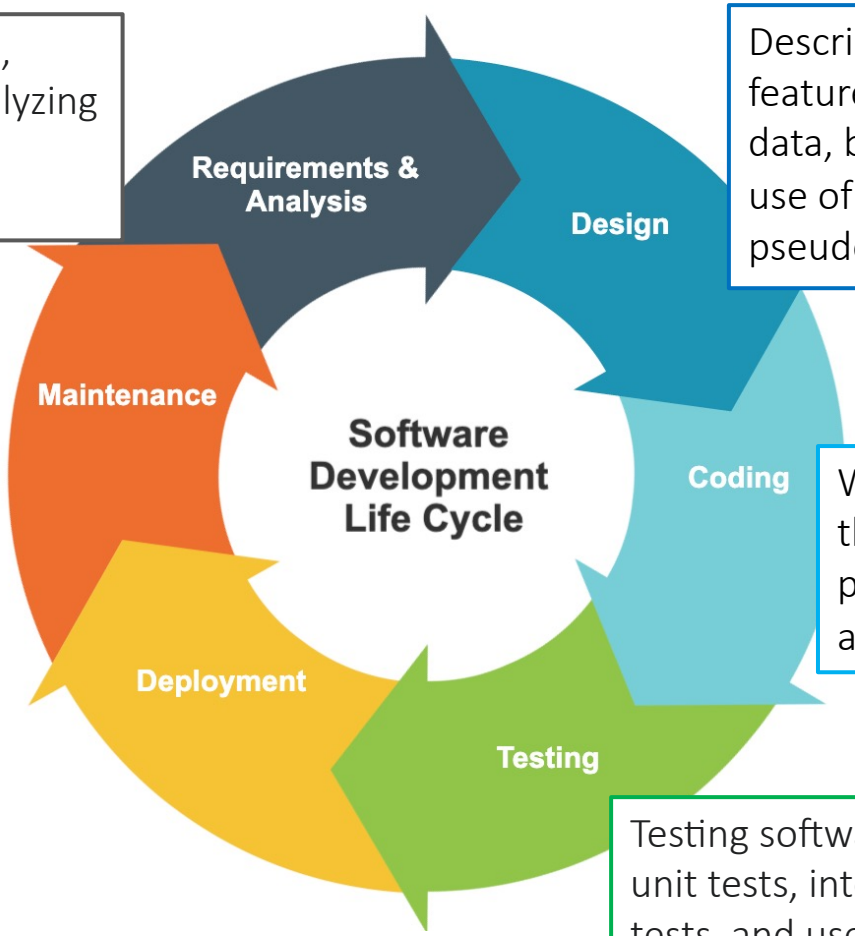


Describing system components, features, operations, functions, data, behavioural rules with the use of models, diagrams and pseudo-code

Writing and maintaining the source code using programming languages and frameworks

Software Development Life Cycle

Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements



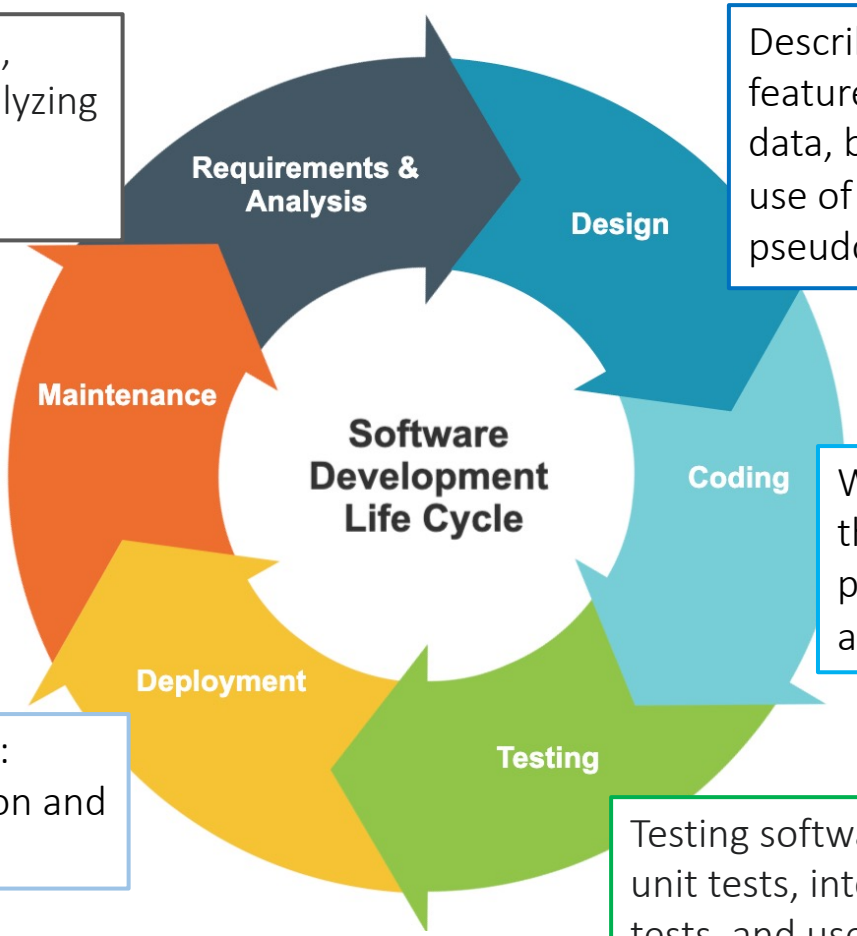
Describing system components, features, operations, functions, data, behavioural rules with the use of models, diagrams and pseudo-code

Writing and maintaining the source code using programming languages and frameworks

Testing software at various levels: unit tests, integration tests, system tests, and user acceptance tests

Software Development Life Cycle

Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements



Describing system components, features, operations, functions, data, behavioural rules with the use of models, diagrams and pseudo-code

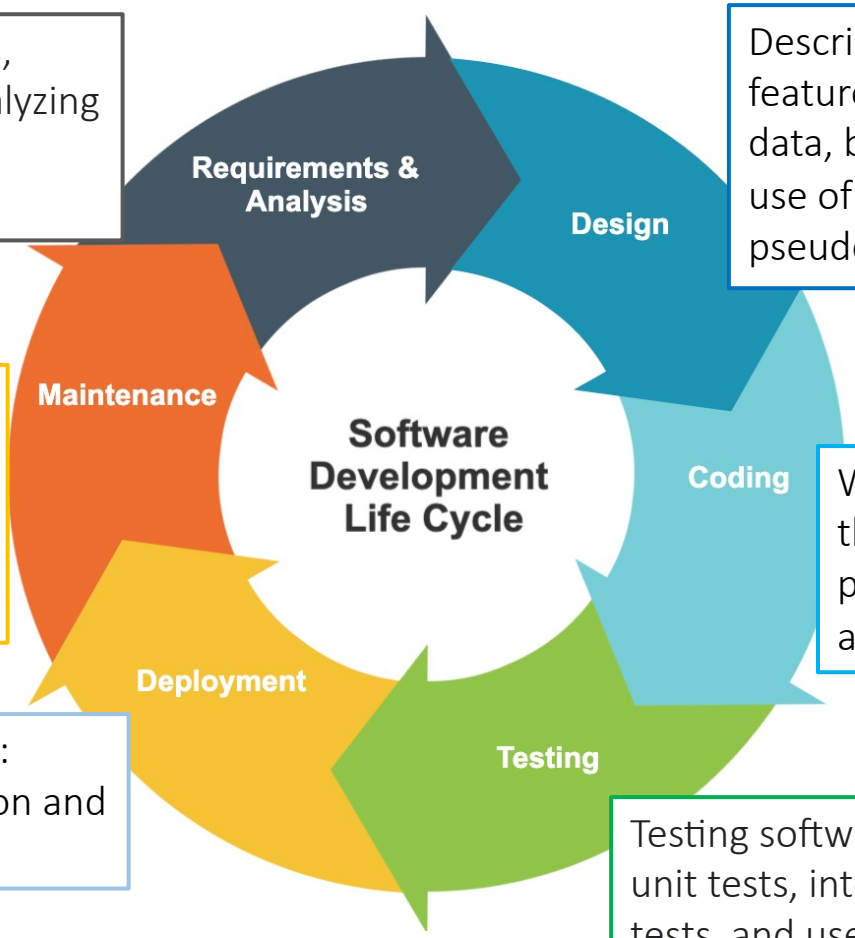
Writing and maintaining the source code using programming languages and frameworks

Testing software at various levels: unit tests, integration tests, system tests, and user acceptance tests

Including the activities of: relese, delivery, installation and activation, update

Software Development Life Cycle

Identifying costs, benefits, resource, user needs, analyzing project goals to define requirements



Describing system components, features, operations, functions, data, behavioural rules with the use of models, diagrams and pseudo-code

Modifying the software product after delivery to correct faults, to improve performance or other attributes

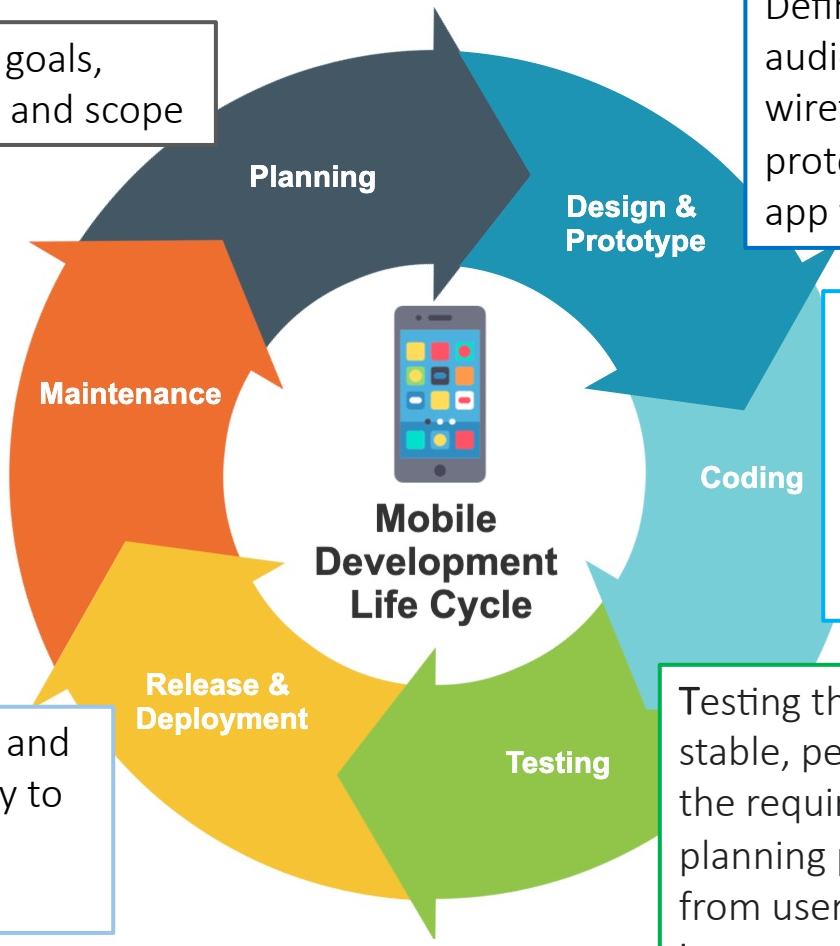
Writing and maintaining the source code using programming languages and frameworks

Including the activities of: relese, delivery, installation and activation, update

Testing software at various levels: unit tests, integration tests, system tests, and user acceptance tests

Mobile Development Life Cycle

Defining project goals, target audience, and scope



Defining project goals, target audience, and scope. Creating wireframes, mockups, and prototypes to visualize how the app will look and function

Writing the code for the app, using programming languages and frameworks, depending on the platform you are targeting (e.g. iOS, Android)

Testing the app to ensure it is stable, performs well, and meets the requirements defined in the planning phase. Also feedback from users can identify issues or improvements.

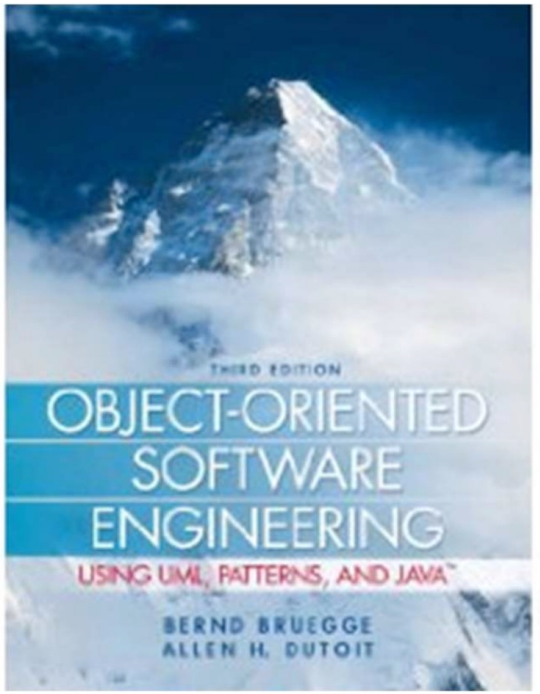
Once the app is complete and has been tested, it is ready to be published and made available to users

Even after the app has been deployed, it needs to be continuously updated to fix bugs, add new features, and keep it running smoothly

MOBILE APP Development

References

Bernd Bruegge, Allen H. Dutoit, Object-Oriented Software Engineering: Using UML, Patterns and Java, 3rd Edition, Publisher: Prentice Hall, Upper Saddle River, NJ, 2009; ISBN-10: 0136061257 ISBN-13: 978-0136061250 (Preface)



Acknowledge

Prof. Henry Muccini (University of L'Aquila)