POLITECNICO
MILANO 1863

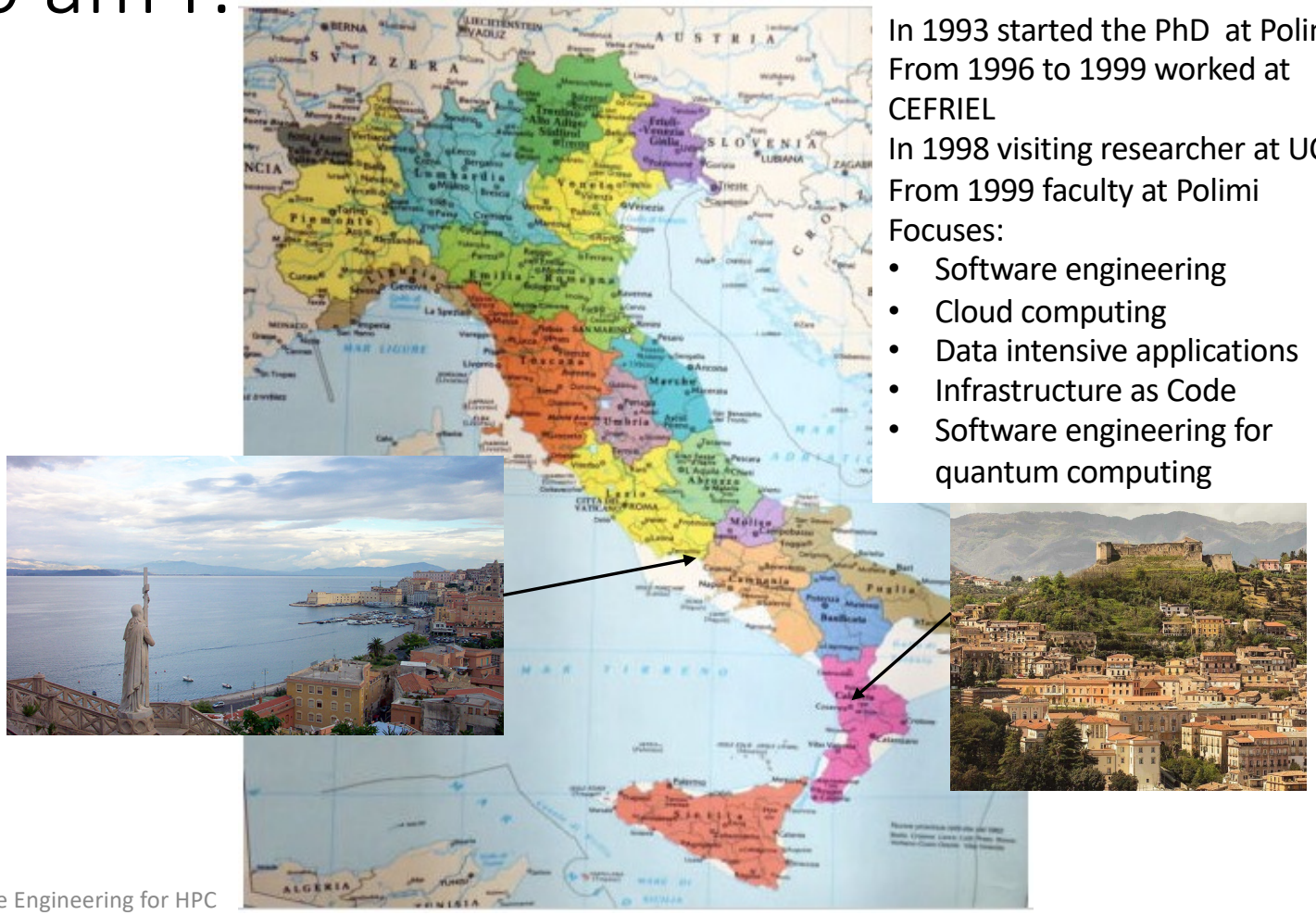
Software Engineering for High Performance Computing course (SE4HCP)

Instructors

POLITECNICO
MILANO 1863

- Elisabetta Di Nitto
  - Office address: Via Golgi, 42
  - email elisabetta.dinitto@polimi.it
  - phone: 02-2399-3663
  - http://dinitto.faculty.polimi.it
- Teaching assistant: Simone Reale
  - Office address: Via Golgi, 42
  - email simone1.reale@polimi.it

Who am I?

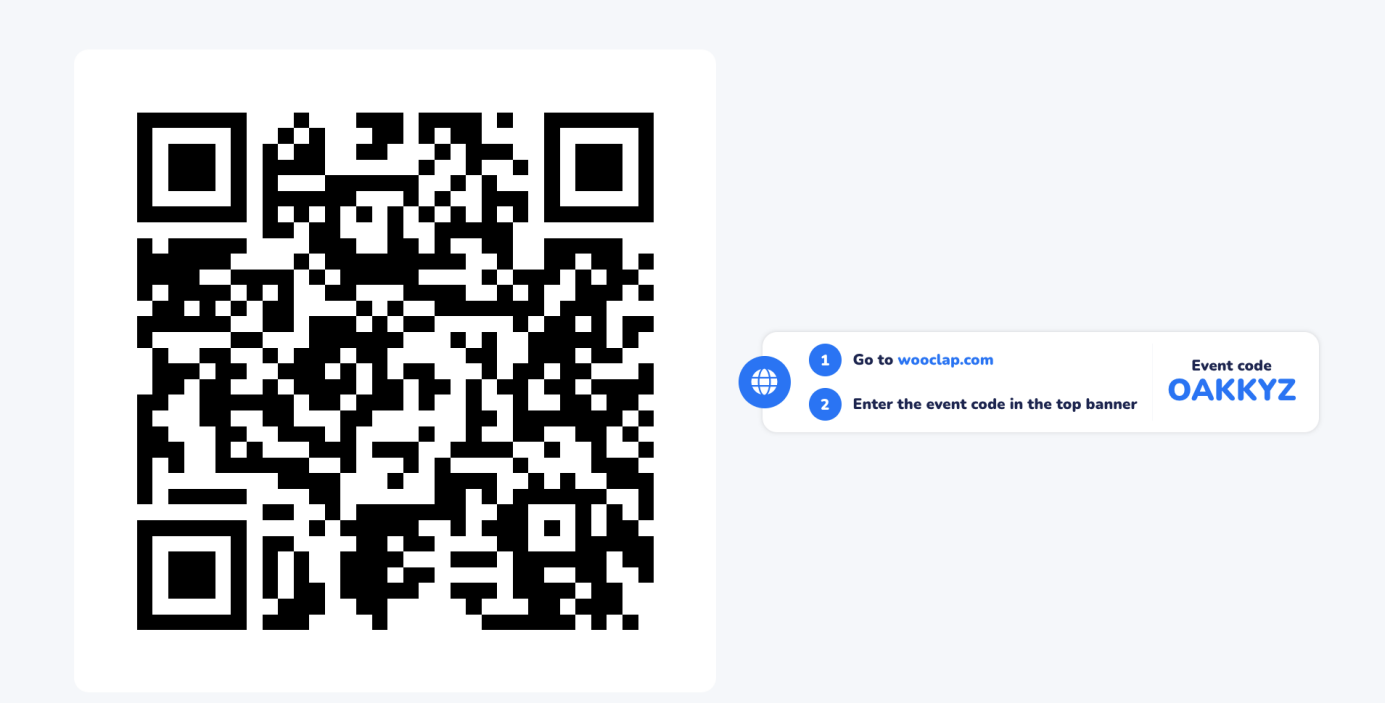


In 1993 started the PhD at Polimi
From 1996 to 1999 worked at CEFRIEL
In 1998 visiting researcher at UCI
From 1999 faculty at Polimi
Focuses:

- Software engineering
- Cloud computing
- Data intensive applications
- Infrastructure as Code
- Software engineering for quantum computing

Info about yourself…

- https://app.wooclap.com/OAKKYZ



Course logistics

POLITECNICO
MILANO 1863

- Lectures will be in person (exceptional cases will be handled on the fly, and case-by-case)
- Schedule
  - Monday 8.15-10.15, room 3.1.6
  - Wednesday 12.15-14.15, room 3.1.12
- Lectures will be recorded and made available to help you during the exam preparation
- Course website is available on webeep. It will kept updated with
  - Course material, online books, announcements, course schedule, links to lecture recordings, homework assignments, …
- Monitor it regularly!

Course objective

POLITECNICO MILANO 1863

- Present the main software engineering methodologies and techniques to organize the **development** and **operation** of HPC applications.

Topics

POLITECNICO
MILANO 1863

- Introduction to the software lifecycle
- Characteristics and qualities of HPC software
- Requirement engineering
- Software design and peculiarities of HPC software
- Software verification and peculiarities of HPC software
- Software configuration management and CI/CD pipelines
- Containment, scheduling and orchestration

Exams rules

POLITECNICO MILANO 1863

- A written exam and some practical activities (the project) developed during the course
- Written exam (up to 16 points, minimum required: 8 points)
  - Software testing and analysis exercises
  - Exercises focusing on design aspects
- Practical activities (up to 16 points, minimum required: 8 points)
  - A simple requirement analysis and software design assignment
  - Setup of a tool pipeline automating software integration, deployment, some operation steps

POLITECNICO
MILANO 1863

Software Engineering

Definitions
The process and product
Phases of the development process
Flexible lifecycles

Why software engineering is important

POLITECNICO MILANO 1863

- Software is everywhere and our society is now totally dependent on software-intensive systems
- Society could not function without software
  - Transportation systems
  - Energy Systems
  - Manufacturing Systems
- Software failures cannot be tolerated

CrowdStrike Outage on July 19th, 2024

Travelers waiting to check in at the airport in Hamburg, Germany, on Friday. Bodo Marks/DPA, via Associated Press

Chaos and Confusion: Tech Outage Causes Disruptions Worldwide

Airlines, hospitals and people's computers were affected after CrowdStrike, a cybersecurity company, sent out a flawed software update.

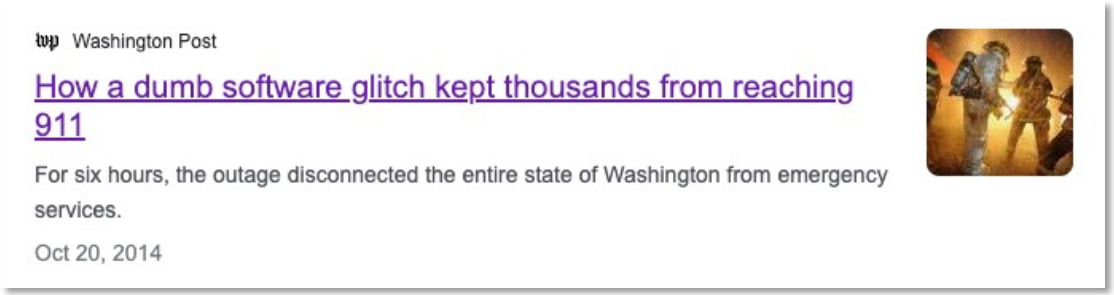
Share full article 951

Details here : CrowdStrike 2024, External Technical Root Cause Analysis — Channel File 291
https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf

911 Outage in April 2014

POLITECNICO MILANO 1863

- [April 10, 2014] Washington State had **no 911 service for six hours**
  - 911 is the phone number for emergency service in USA
  - People calling were getting the busy signal
    - A woman called more than 37 times while a stranger was breaking into her house
  - People at the central office were not aware of the problem
  - More at: https://www.theatlantic.com/technology/archive/2017/09/saving-the-world-from-code/540393/



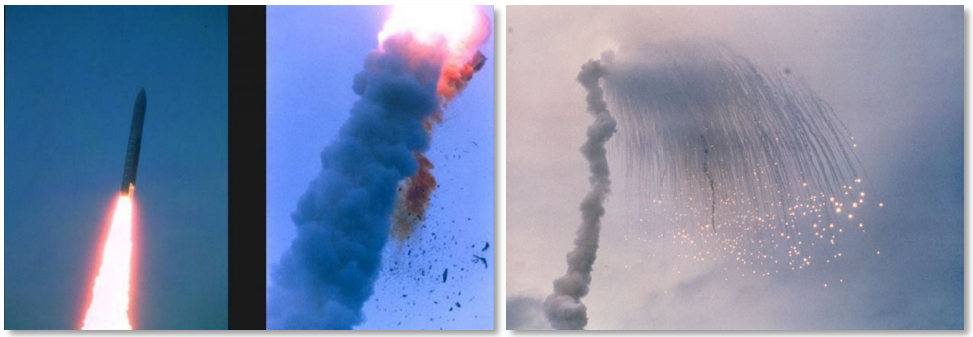
911 Outage in April 2014

POLITECNICO MILANO 1863

- Why did this happen? **Software issue!**
  - The software dispatching the calls had a counter used to assign a unique identifier to each call
  - The counter went over the threshold defined by developers…
  - All calls from that moment on were rejected

More on failures… Ariane 5, 1996

- [June 4, 1996] 40s after take off, Ariane 5 broke up and exploded
  - The total cost for developing the launcher has been of 8000M$
  - The launcher contained a cluster of satellites for a value of 500M$
  - More at
    - http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html (accident tech report)
    - https://www.youtube.com/watch?v=PK_yguLapgA (video)



More on failures… Ariane 5, 1996

POLITECNICO MILANO 1863

- The explosion has been caused by a software failure
- From the tech report:
  - “The failure […] was caused by the complete loss of guidance and altitude information […]. This loss of information was due to **specification and design errors** in the software of the inertial reference system.”
  - “The extensive reviews and tests carried out during the Ariane 5 Development Programme **did not include adequate analysis and testing** of the inertial reference system or of the complete flight control system, which could have detected the potential failure.”

Software engineering: definition

POLITECNICO MILANO 1863

- Field of computer science dealing with software systems
  - large and complex
  - built by teams
  - exist in many versions
  - last many years
  - undergo changes
- Multi-person construction of multi-version software

Software engineer: required skills

POLITECNICO MILANO 1863

- **Programming skills not enough!**
  - Programmer:
    - develops a complete program
    - works on known specifications
    - works individually
  - Software engineer:
    - identifies requirements and develops specifications
    - designs a component to be combined with other components, developed, maintained, used by others; component can become part of several systems
    - works in a team

Skills of software engineers

POLITECNICO MILANO 1863

- Technical
- Project management
- Cognitive
- Enterprise organization
- Interaction with different cultures
- Domain knowledge

- The quality of human resources is of primary importance

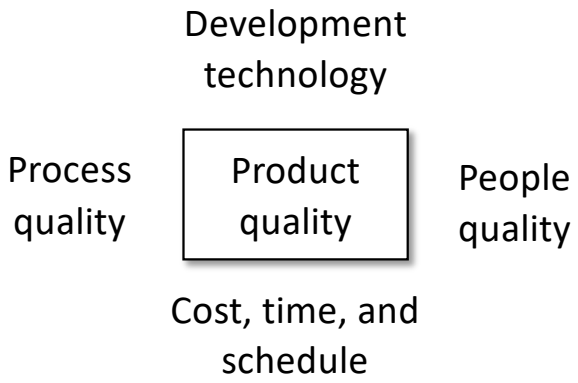
Process and product

POLITECNICO MILANO 1863

- Our goal is to develop **software products**
- The **process** is how we do it
- Both are extremely important, due to the nature of the software product
- Both have qualities
  - in addition, quality of process affects quality of product
  - …even though, other aspects such as the quality of the development team are important as well

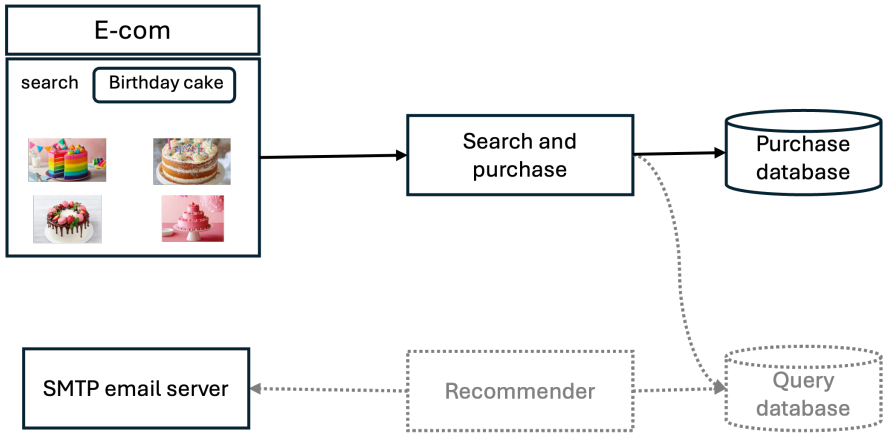
The software product

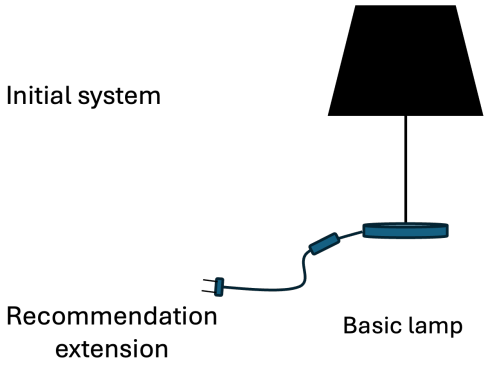
- Different from traditional types of products
  - intangible
    - difficult to describe and evaluate
  - malleable
  - human intensive
    - does not involve any trivial manufacturing process
- Aspects affecting product quality

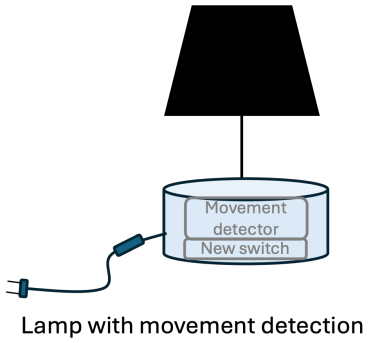


What does it mean for software to be malleable?

- Modifying an e-commerce system and a table lamp…

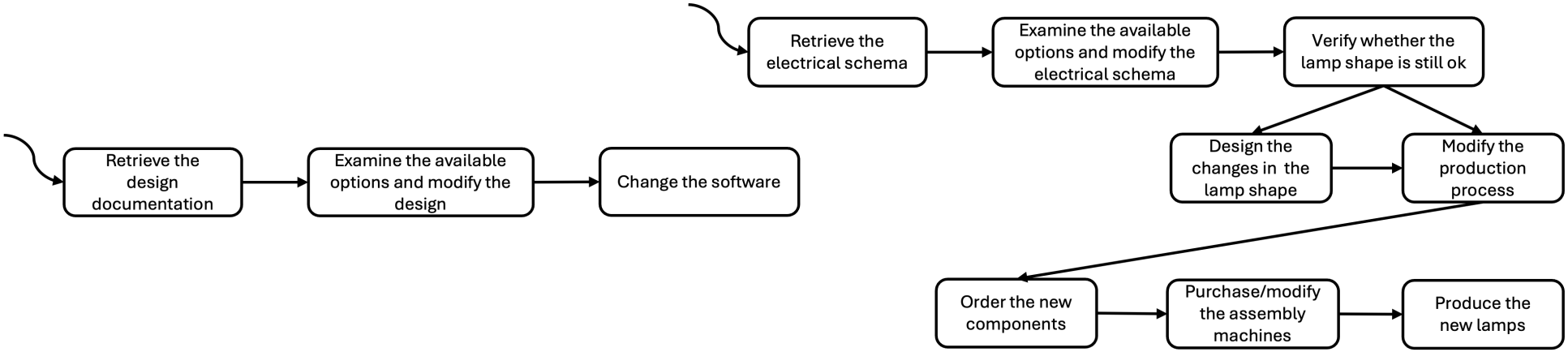






What does it mean for software to be malleable?

- E-commerce system vs table lamp: impact on the development process



Software product qualities – ISO/IEC 25010:2024 Software Quality Model

POLITECNICO MILANO 1863

SOFTWARE PRODUCT QUALITY								
FUNCTIONAL SUITABILITY	**PERFORMANCE EFFICIENCY**	**COMPATIBILITY**	**INTERACTION CAPABILITY**	**RELIABILITY**	**SECURITY**	**MAINTAINABILITY**	**FLEXIBILITY**	**SAFETY**
FUNCTIONAL COMPLETENESS FUNCTIONAL CORRECTNESS FUNCTIONAL APPROPRIATENESS iso25000.com	TIME BEHAVIOUR RESOURCE UTILIZATION CAPACITY	CO-EXISTENCE INTEROPERABILITY	APPROPRIATENESS RECOGNIZABILITY LEARNABILITY OPERABILITY USER ERROR PROTECTION USER ENGAGEMENT INCLUSIVITY USER ASSISTANCE SELF-DESCRIPTIVENESS	FAULTLESSNESS AVAILABILITY FAULT TOLERANCE RECOVERABILITY	CONFIDENTIALITY INTEGRITY NON-REPUDIATION ACCOUNTABILITY AUTHENTICITY RESISTANCE	MODULARITY REUSABILITY ANALYSABILITY MODIFIABILITY TESTABILITY	ADAPTABILITY SCALABILITY INSTALLABILITY REPLACEABILITY	OPERATIONAL CONSTRAINT RISK IDENTIFICATION FAIL SAFE HAZARD WARNING SAFE INTEGRATION

https://iso25000.com/index.php/en/iso-25000-standards/iso-25010

Process qualities: productivity

POLITECNICO MILANO 1863

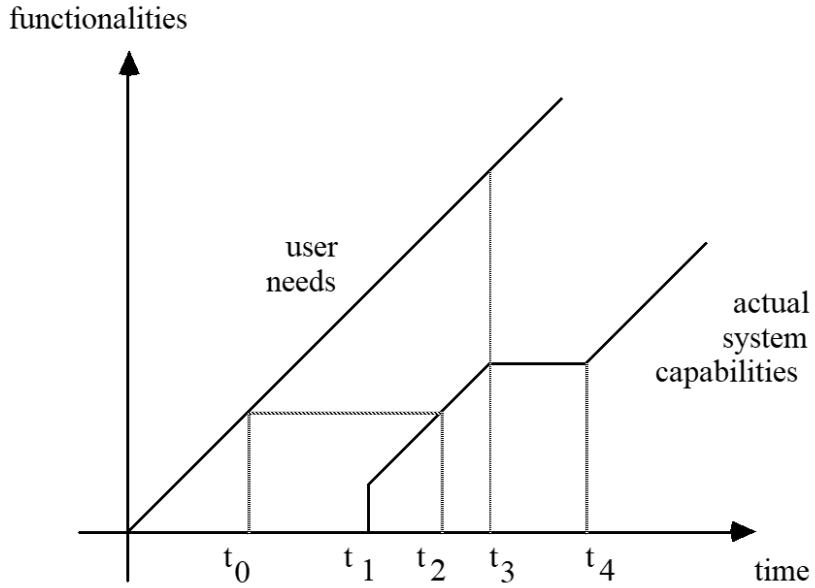
- Ability to produce a “good” amount of product
- How can we measure it? → Delivered items by unit of effort

lines of code (and variations)
function points

person month
WARNING: persons and months cannot be interchanged

Process qualities: timeliness

- Ability to respond to change requests in a timely fashion

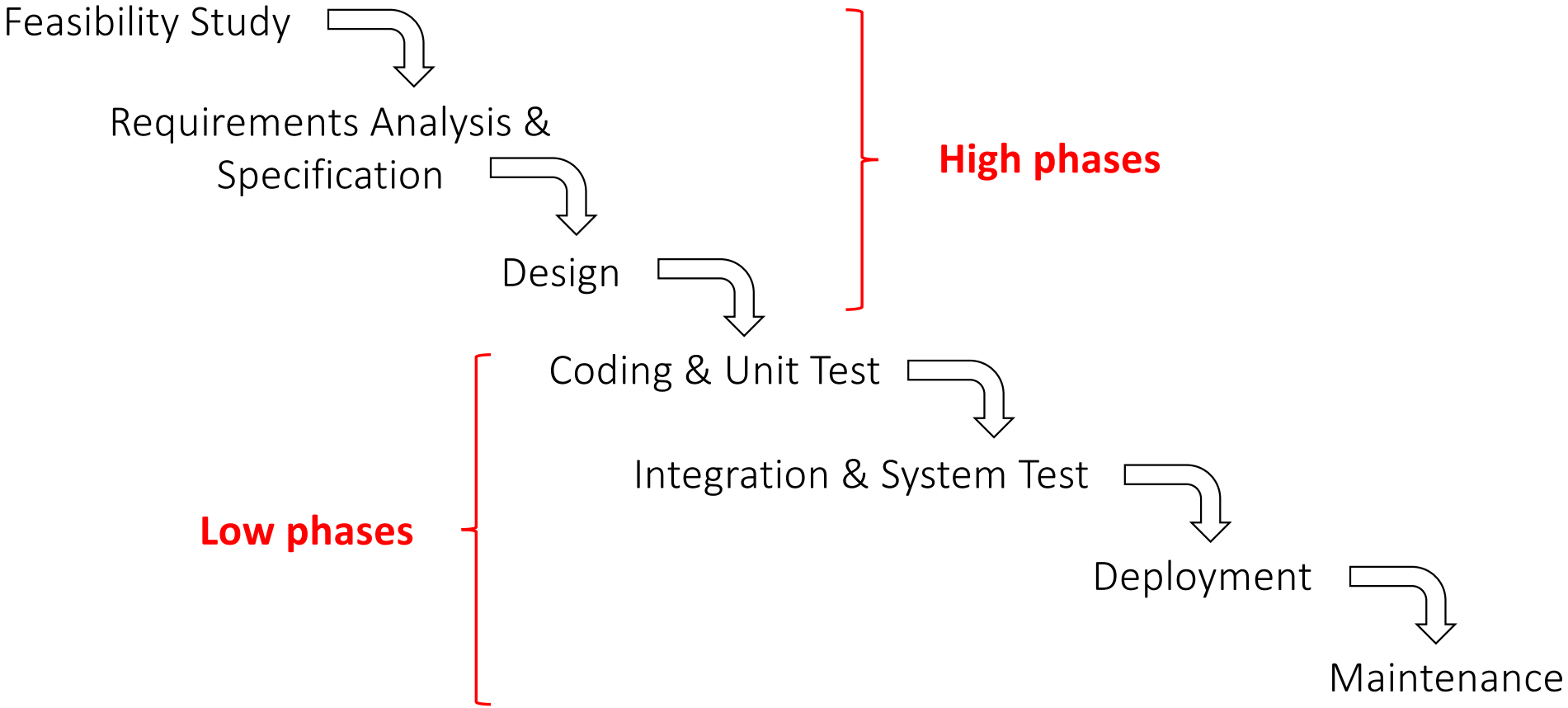


Software lifecycles

POLITECNICO MILANO 1863

- Initially, no reference model: Just code & fix
- As a reaction to the many problems: traditional “waterfall” model
  - identify phases and activities
  - force linear progression from a phase to the next
  - no returns (they are harmful)
    - better planning and control
  - standardize outputs (artifacts) from each phase
  - Software like manufacturing
- Then, flexible processes: iterative models, agile methods, DevOps

A waterfall organization



Feasibility study & project estimation

POLITECNICO MILANO 1863

- Cost/benefit analysis
- Determines whether the project should be started (e.g., buy vs make), possible alternatives, needed resources
- Outcome
  - **Feasibility Study Document**
    - Preliminary problem description
    - Scenarios describing possible solutions
    - Costs and schedule for the different alternatives

Requirement analysis and specification

POLITECNICO MILANO 1863

- Analyze the domain in which the application takes place
- Identify requirements
- Derive specifications for the software
  - Requires an (continuous) interaction with the user
  - Requires an understanding of the properties of the domain
- Outcome
  - **Requirements Analysis and Specification Document (RASD)**

Design

POLITECNICO MILANO 1863

- Defines the software architecture
  - Components (modules)
  - Relations among components
  - Interactions among components
- Goal
  - Support concurrent development, separate responsibilities
- Outcome
  - **Design Document**

Coding&Unit level quality assurance

POLITECNICO MILANO 1863

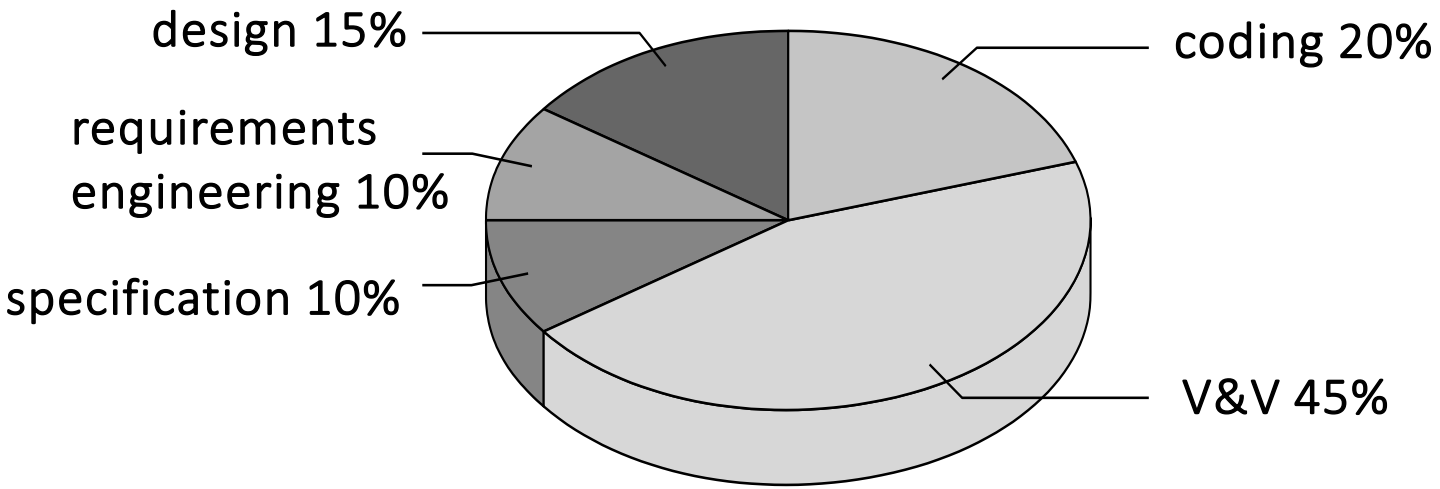
- Each module is implemented using the chosen programming language
- Each module is tested in isolation by the module developer
- Inspection can be used as an additional quality assurance approach
- Programs include their documentation

Integration&System test

POLITECNICO MILANO 1863

- Modules are integrated into (sub)systems
- Integrated (sub)systems are tested
- Follows an incremental implementation scheme
- Complete system test needed to verify overall properties
- Sometimes we have **alpha test** and **beta test**

Effort distribution (van Vliet 2008)



Maintenance

POLITECNICO MILANO 1863

- Corrective maintenance: deals with the repair of faults or defects found ≈ 20%
- Evolution
  - **adaptive** maintenance: consists of adapting software to changes in the environment (the hardware or the operating system, business rules, government policies…) ≈ 20%
  - **perfective** maintenance: mainly deals with accommodating to new or changed user requirements ≈ 50%
  - **preventive** maintenance: concerns activities aimed at increasing the system's maintainability ≈ 10%

Correction vs evolution

POLITECNICO MILANO 1863

- Distinction can be unclear, because specifications are often incomplete and ambiguous
- This causes problems because specs are often part of a contract between developer and customer
  - early frozen specs can be problematic, because they are more likely to be wrong

Problems of software evolution

POLITECNICO MILANO 1863

- It is (almost) never anticipated and planned; this causes disasters
- Software is very easy to change
  - often, under emergency, changes are applied directly to code
  - inconsistent state of project documents

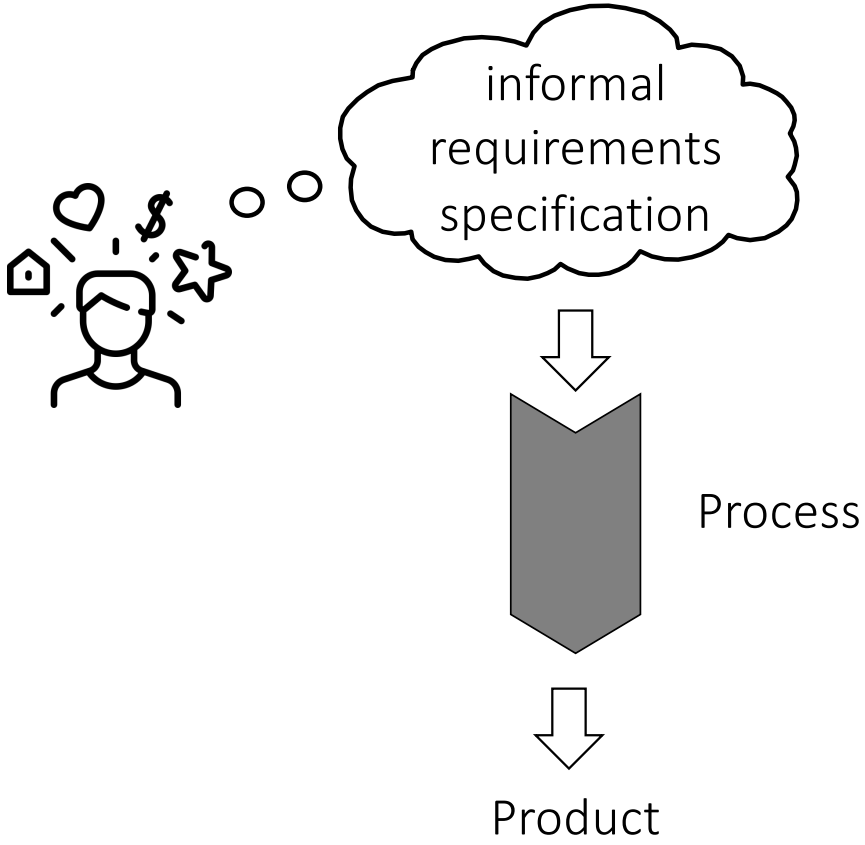
How to face evolution

POLITECNICO MILANO 1863

- Good engineering practice:
  - first modify design, then change implementation
  - apply changes consistently in all documents
- Likely changes must be anticipated
- Software must be designed to accommodate changes in a cost-effective way

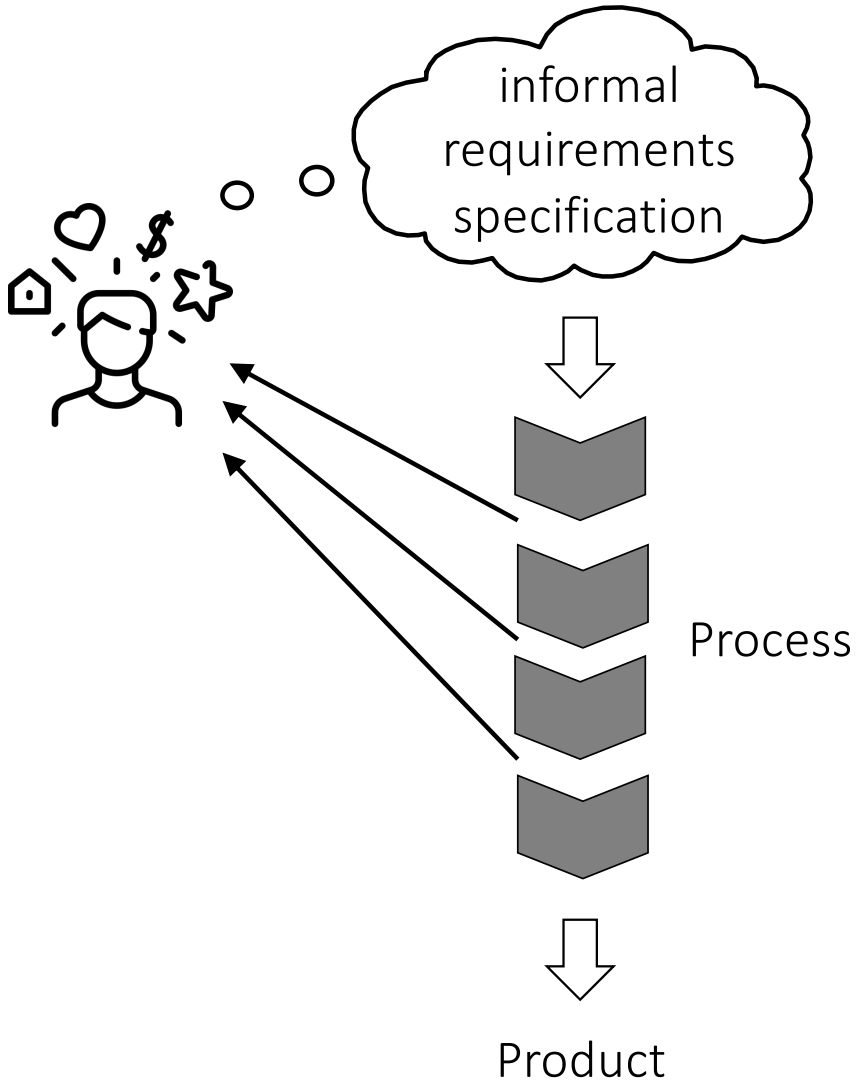
- **This is one of the main goals of software engineering**

Waterfall is “black box”

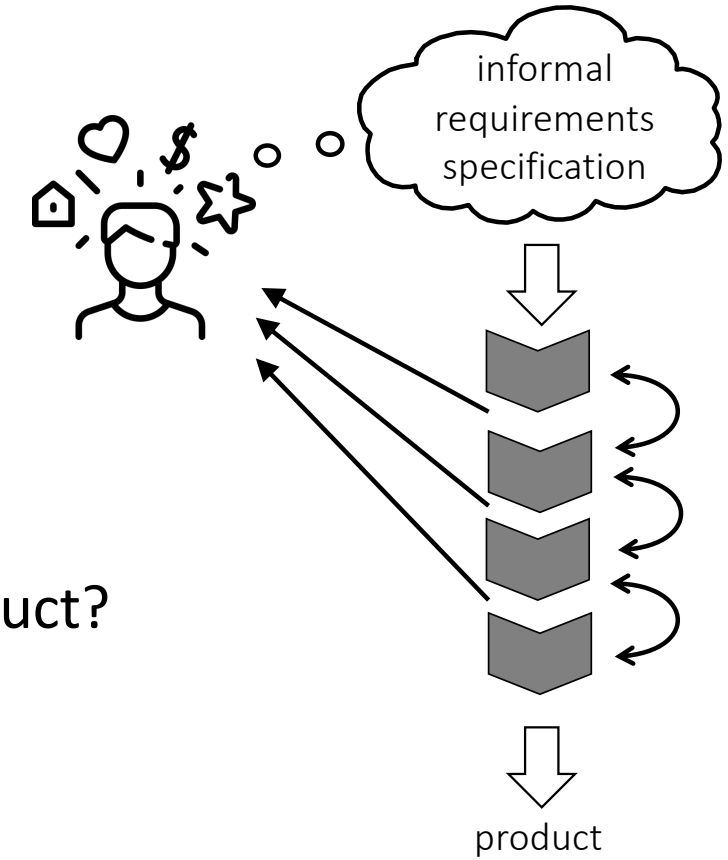


Need for transparency

- Transparency allows early check and change via feedback
- It supports flexibility



Verification and validation



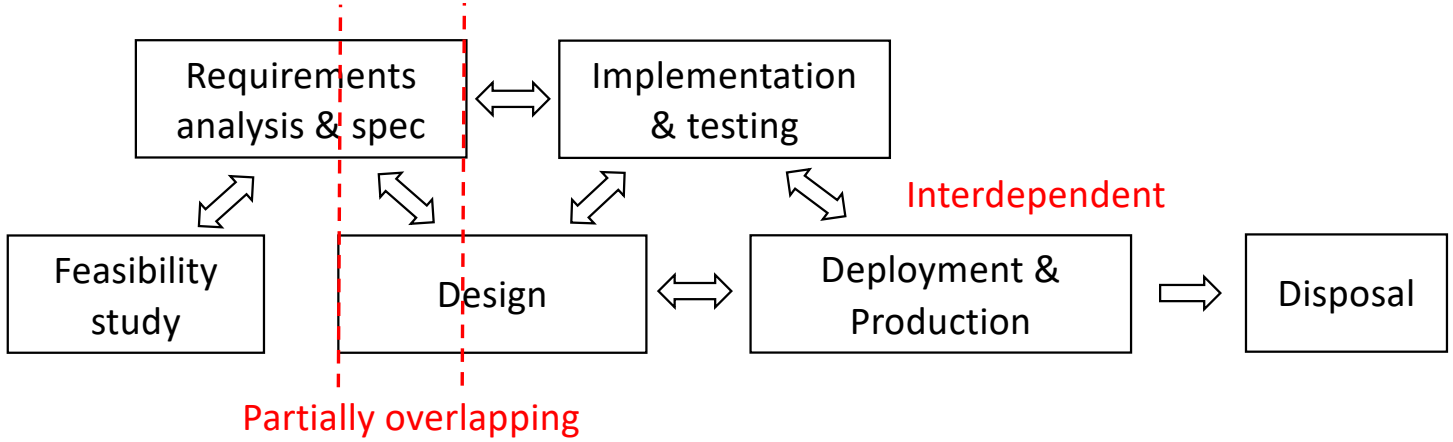
Validation:
are we making
the ”right” product?

Verification:
are we doing
the product right?

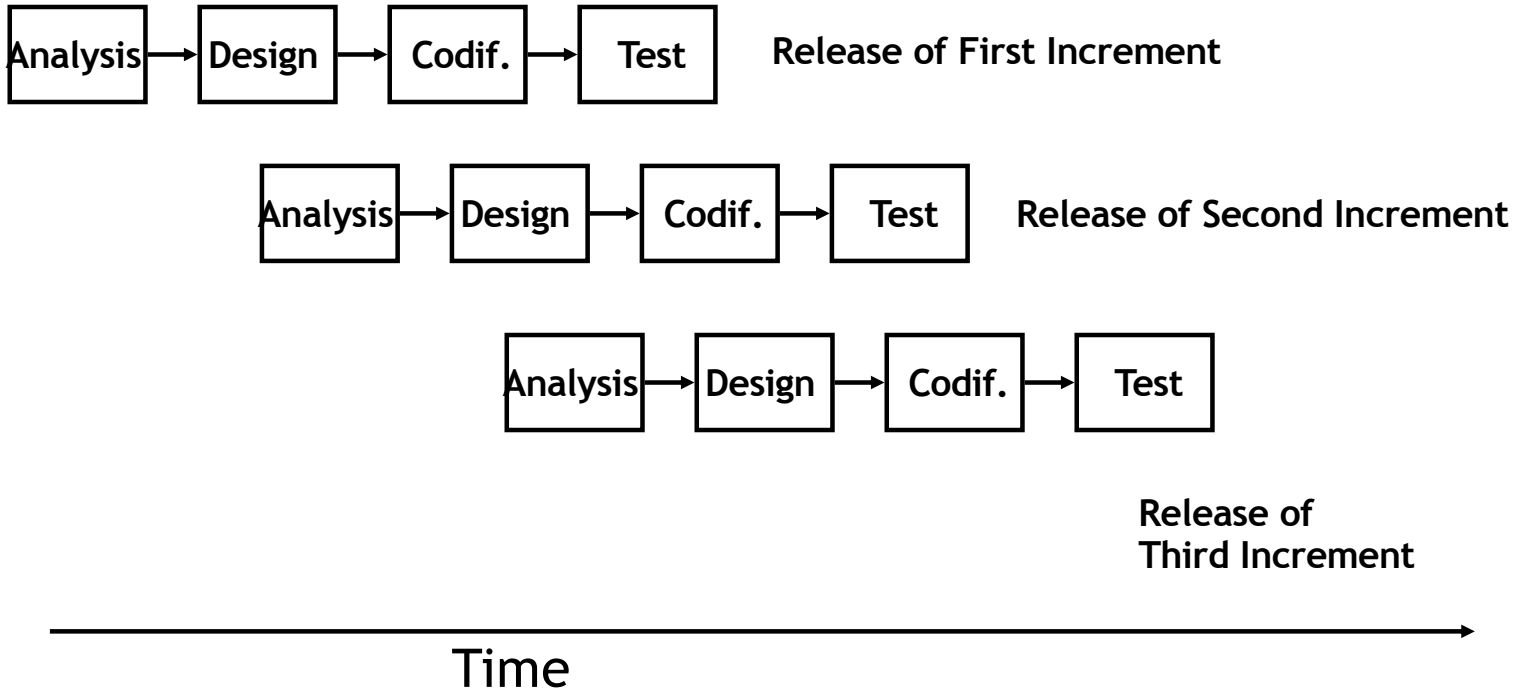
Flexible processes

- Main goal: adapt to changes (especially in requirements and specs)
- The very idea:
  - stages are not necessarily sequential
  - processes become iterative and incremental

Example



Incremental delivery



The agile manifesto - principles

https://agilemanifesto.org/

POLITECNICO MILANO 1863

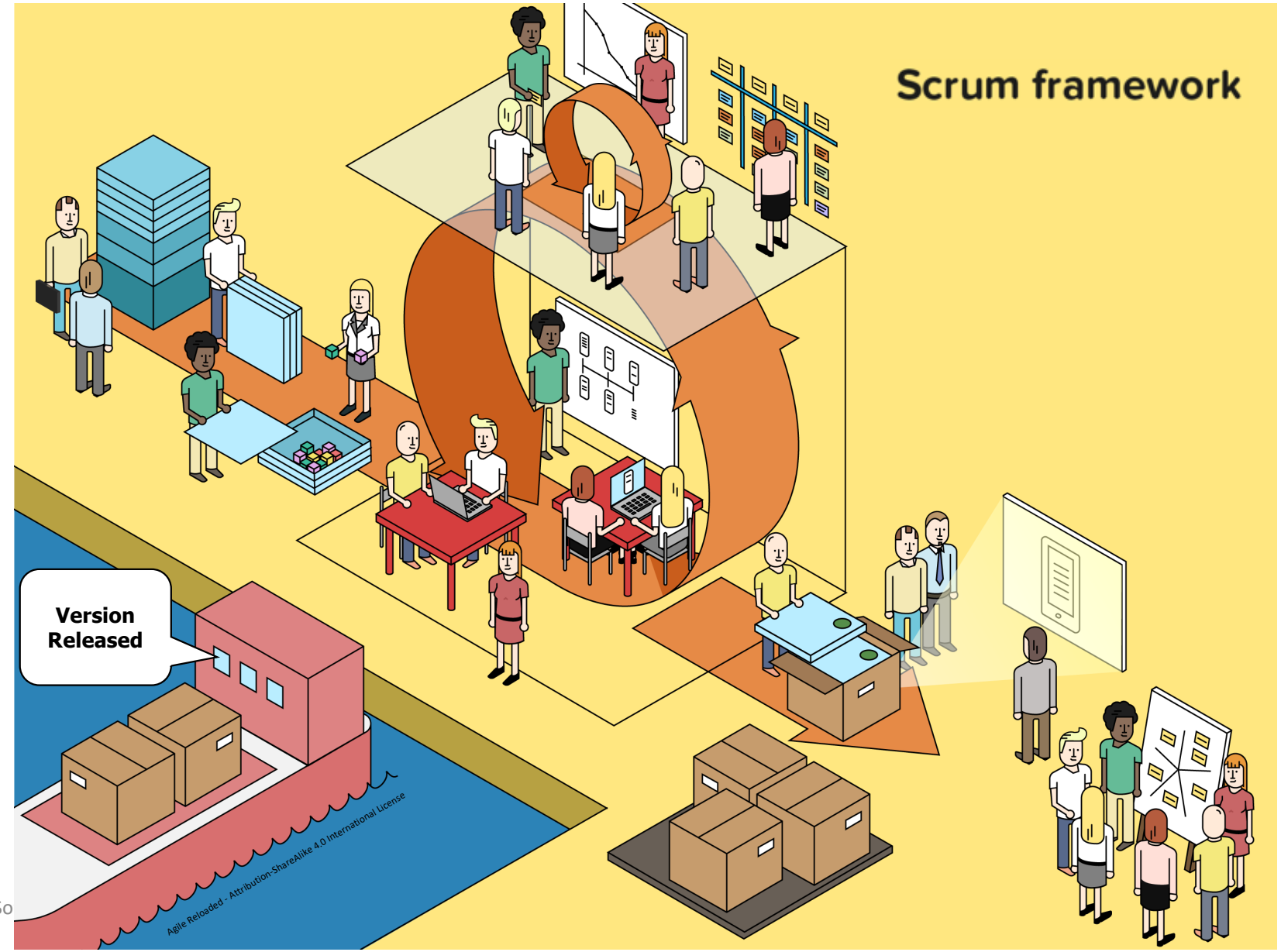
- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.

The agile manifesto - principles

https://agilemanifesto.org/

POLITECNICO MILANO 1863

- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
- Continuous attention to technical excellence and good design enhances agility.
- Simplicity--the art of maximizing the amount of work not done--is essential.
- The best architectures, requirements, and designs emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.


Scrum framework
Version Released
Agile Reloaded - Attribution-ShareAlike 4.0 International License

So

DevOps (Development and Operation)

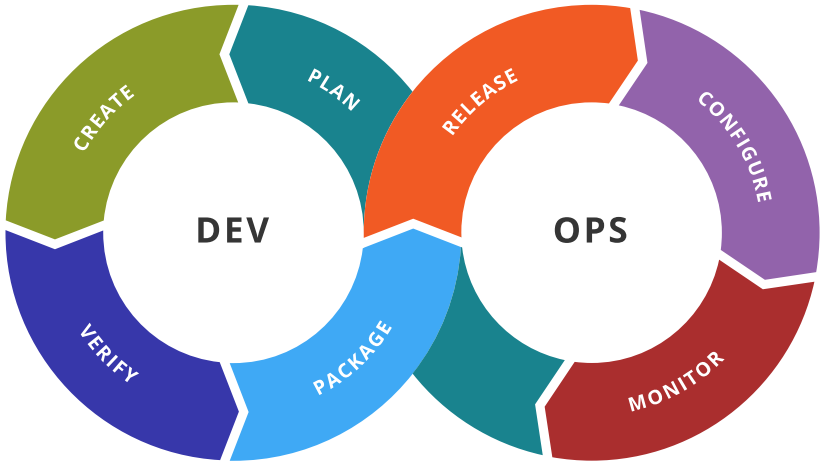


Image by Kharnagy (Own work) [CC BY-SA 4.0 (http://creativecommons.org/licenses/by-sa/4.0)], via Wikimedia Commons

Waterfall vs. Agile vs. DevOps

POLITECNICO MILANO 1863

Concerns	Waterfall	Agile	DevOps
Main Focus	Design and development	Design and development	**Whole application life-cycle**
Attention to Operation	Minimal or none	Minimal or none	**Main concern**
Role of Maintenance	Maintenance seen as a redevelopment phase	Maintenance seen as a redevelopment phase	**Replaces a running system with a running system**
Process Outcome	A packaged system ready to be deployed	*Various intermediate demo versions of the system*	**A continuously updated running system**
Quality Assurance	Testing typically performed toward the end of the process	*Test-driven development*	**Test-driven development and monitoring-driven operation with feedback to Dev**

“Assignment” for next lecture (Wednesday)

- Prepare your own answers to the following questions
  - What is High Performance Computing?
  - What are the qualities a HPC software should offer?
  - What are the most critical issues we should pay attention to during the lifecycle of HPC software?