Introduction to Software Engineering

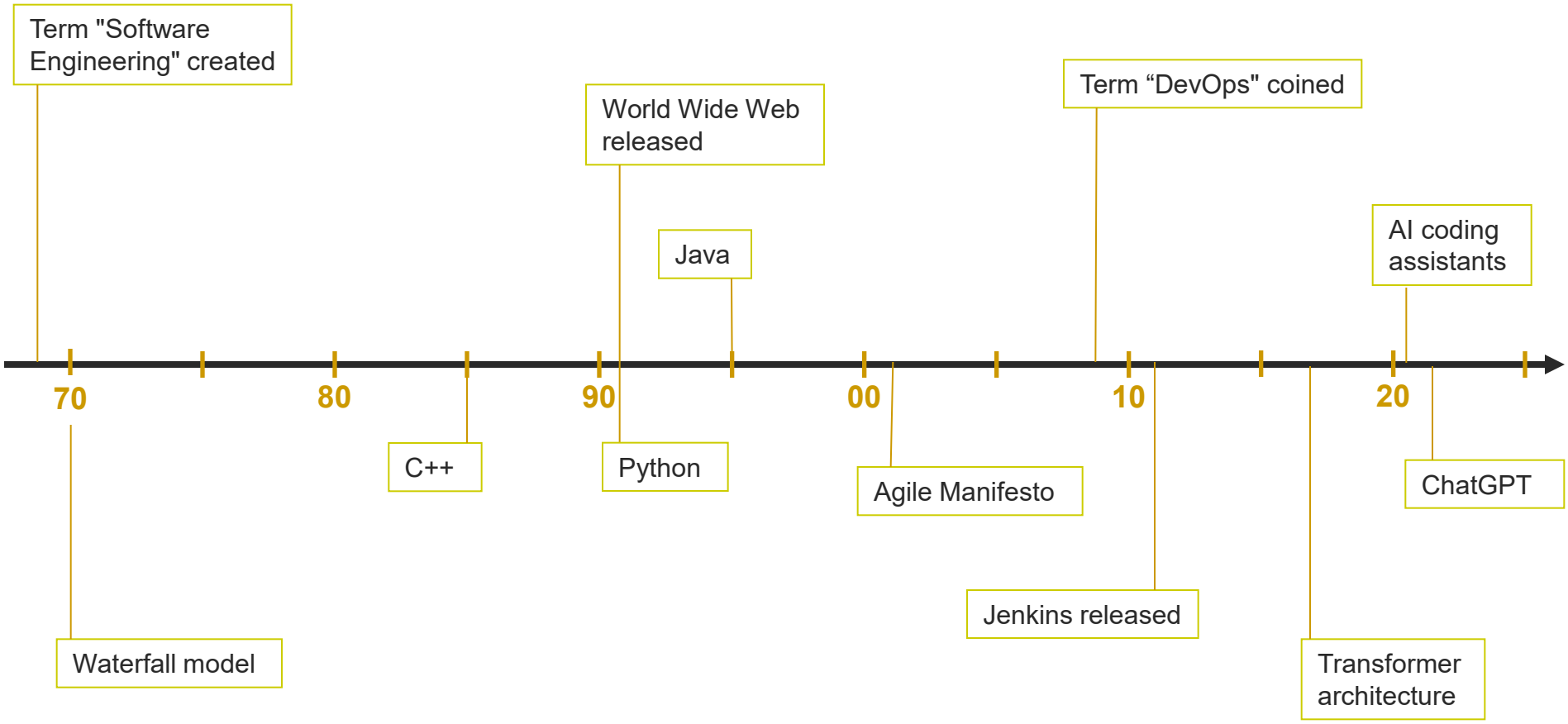
Eduardo Figueiredo

http://www.dcc.ufmg.br/~figueiredo

Software Crisis

- The term “Software Engineering" (SE) was popularized in the NATO Conference
  - In 1968, it brought together experts to address the “software crisis”
- Software projects
  - Running over budget
  - Missing deadlines
  - Failing to meet quality standards

Timeline of Main SE Events



Dominant Development

- 1960s Ad-hoc programming
- 1970s Waterfall
- 1980s Structured programming
- 1990s Object-oriented programming
- 2000s Agile Development
- 2010s DevOps and CI/CD
- 2020s AI-assisted development

What is Software Engineering?

Software engineering is an engineering discipline concerned with all aspects of software production.

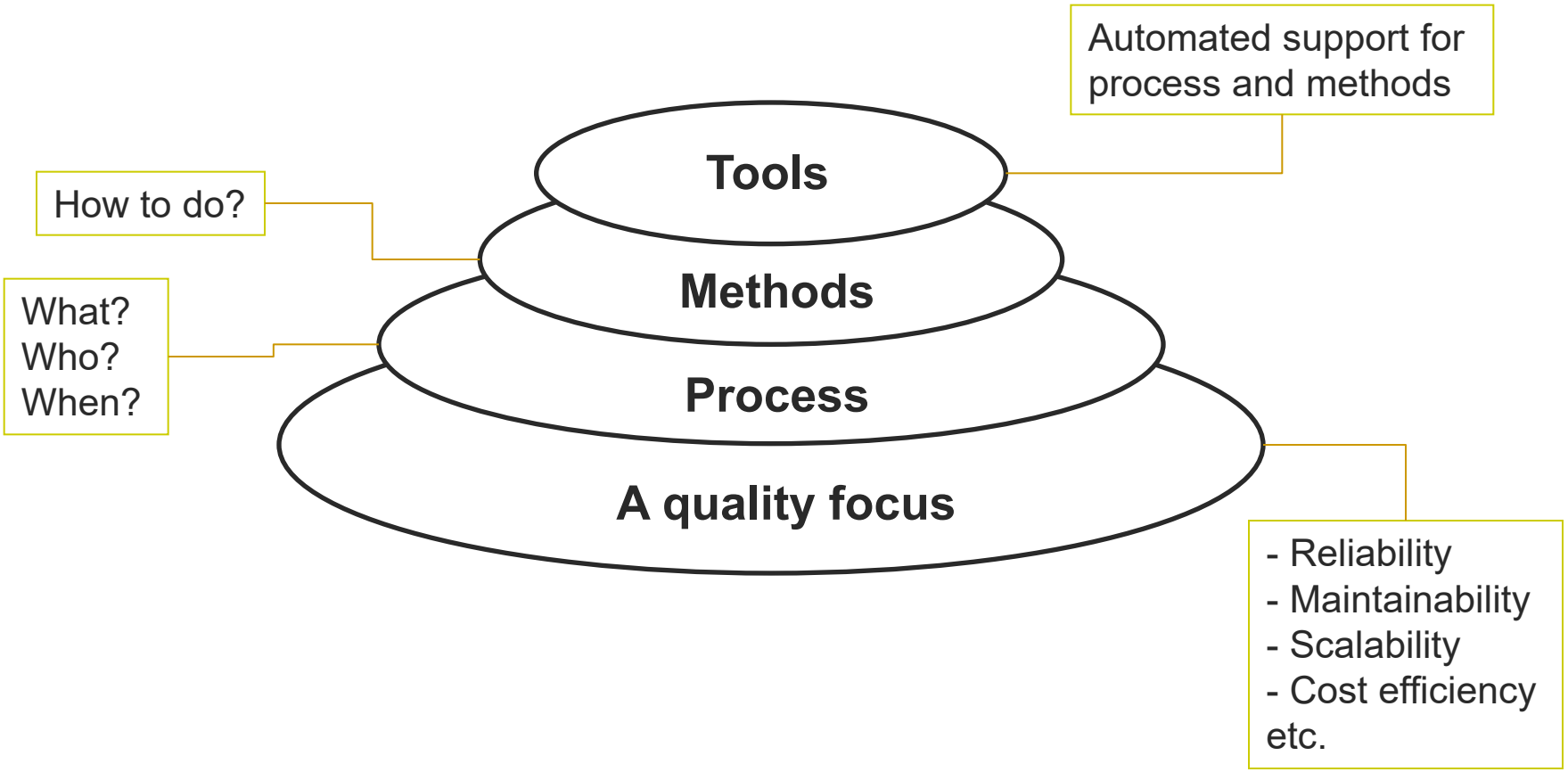
Ian Sommerville

Software engineering is the systematic, disciplined, and quantifiable approach to developing, operating, and maintaining software systems.

IEEE

- Key Software Engineering goals
  - Software Quality
  - Cost Efficiency

Layers of Software Engineering



Software Quality

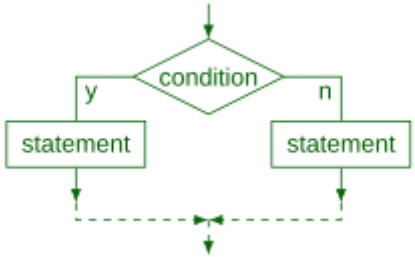
- In general, quality means that a product should meet its specification
- This definition is not easy for software
  - Some quality requirements are difficult to specify in an unambiguous way
  - Software specifications are usually incomplete and often inconsistent
- The focus is on ‘fitness for purpose’ rather than specification conformance

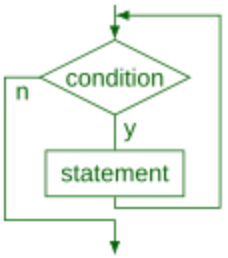
Software Quality Attributes

Safety	Understandability	Portability
Security	Testability	Usability
Reliability	Adaptability	Reusability
Resilience	Modularity	Efficiency
Robustness	Complexity	Learnability

Structured Programming

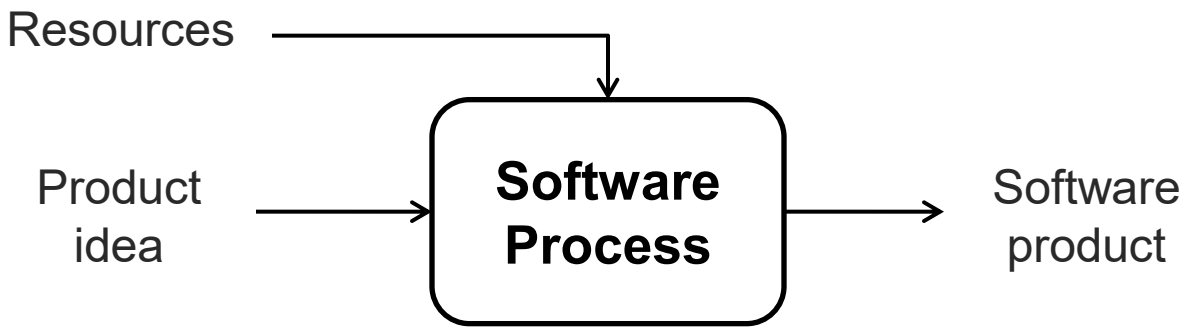
- Structured program theorem (1966)
  - Sequence
  - Selection (Condition)
  - Iteration





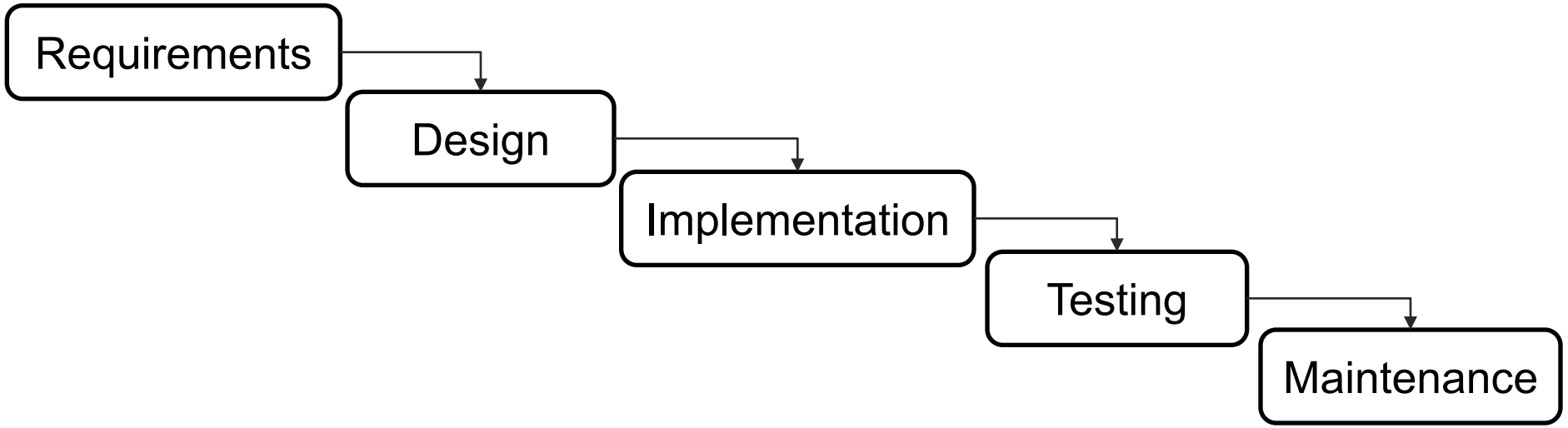
- Principles:
  - Avoid GOTO
  - Use loops and functions
  - Improve code readability

Software Process



- The complexity of software process means that it is hard to be optimized
  - Companies need to continuous try to improve their software processes

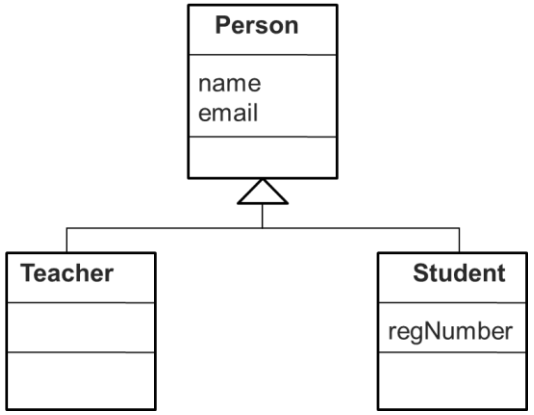
Process: Waterfall Model (1970)



- Phases in sequential order
  - It gives emphasis on documentation
  - Clients may not know the exact requirements before they see working software

Object-Oriented Programming

- Software is modeled as objects with state (attributes) and behavior (methods)
  - Key concepts: class, object, inheritance
- Expected benefits
  - Higher-level of abstraction
  - Modularity
  - Reusability
  - Maintainability

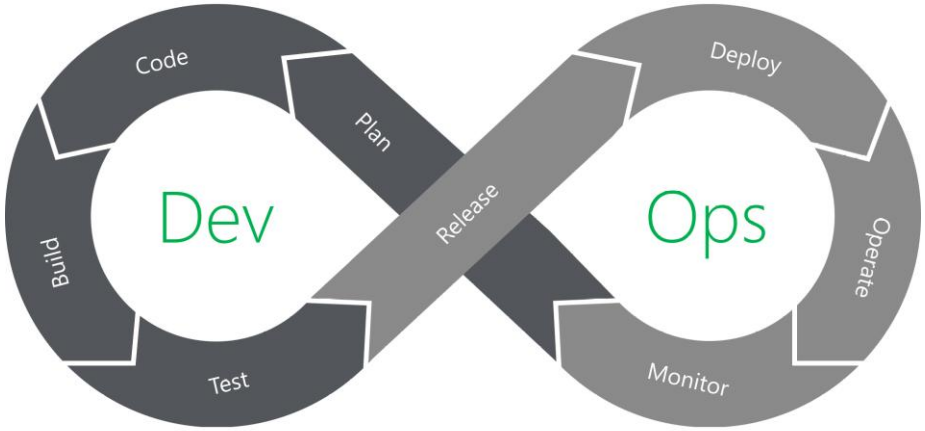


Agile Manifesto (2001)

- We value:
  1. **Individuals and interactions** over processes and tools
  2. **Working software** over comprehensive documentation
  3. **Customer collaboration** over contract negotiation
  4. **Responding to change** over following a plan

http://agilemanifesto.org/

DevOps and CI/CD



- It aims to bridge the gap between development and operations teams
  - 2010s: Rise of CI/CD and tools like Docker, Kubernetes, and Jenkins

AI for Software Engineering

- The main expected benefit is automation
  - By applying AI across the entire lifecycle in partnership with human developers
- AI Tools in Software Development
  - *Claude Code*: terminal-based tool built for agent-style, multi-step workflows
  - *GitHub Copilot*: assistant for real-time code suggestions and chat-based help

Bibliography

- Ian Sommerville. **Software Engineering**, 10th Edition. Pearson Education, 2016.
- Roger S. Pressman, Bruce Maxim. **Software Engineering: A Practitioner's Approach**, 8th Ed. McGraw Hill, 2014.
- C. Wohlin et al. **Experimentation in Software Engineering**, Springer. 2012.