

# Agile Processes

---

Dirk Riehle, Univ. Erlangen

**AMOS B03**

Licensed under [CC BY 4.0 International](https://creativecommons.org/licenses/by/4.0/)

# Reminder About Common Mistakes

---

1. Please sign-off your commits and declare your co-authors, if any
2. During sprint review, please show and tell, not just tell
3. Don't forget your sprint preparation meeting

# Agenda

---

1. Software development
2. Plan-driven development
3. Agile methods
4. Scrum
5. Build process review

# **1. Software Development**

---

# Products vs. Projects

---

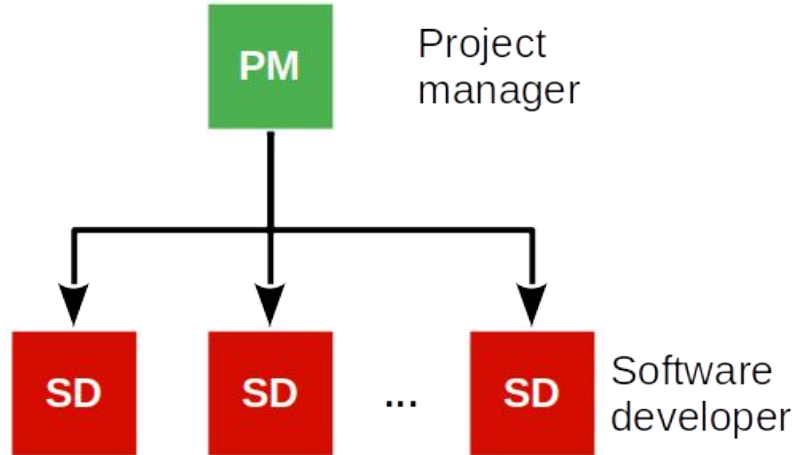
Products have a life-cycle; may live forever

- Products are developed for a market (many customers)

Projects have a defined start and end date

- Projects are developed for one client (one customer)

# Traditional Software Project Organization (Consulting Firm)



# Job Descriptions in Software Consulting Projects

---

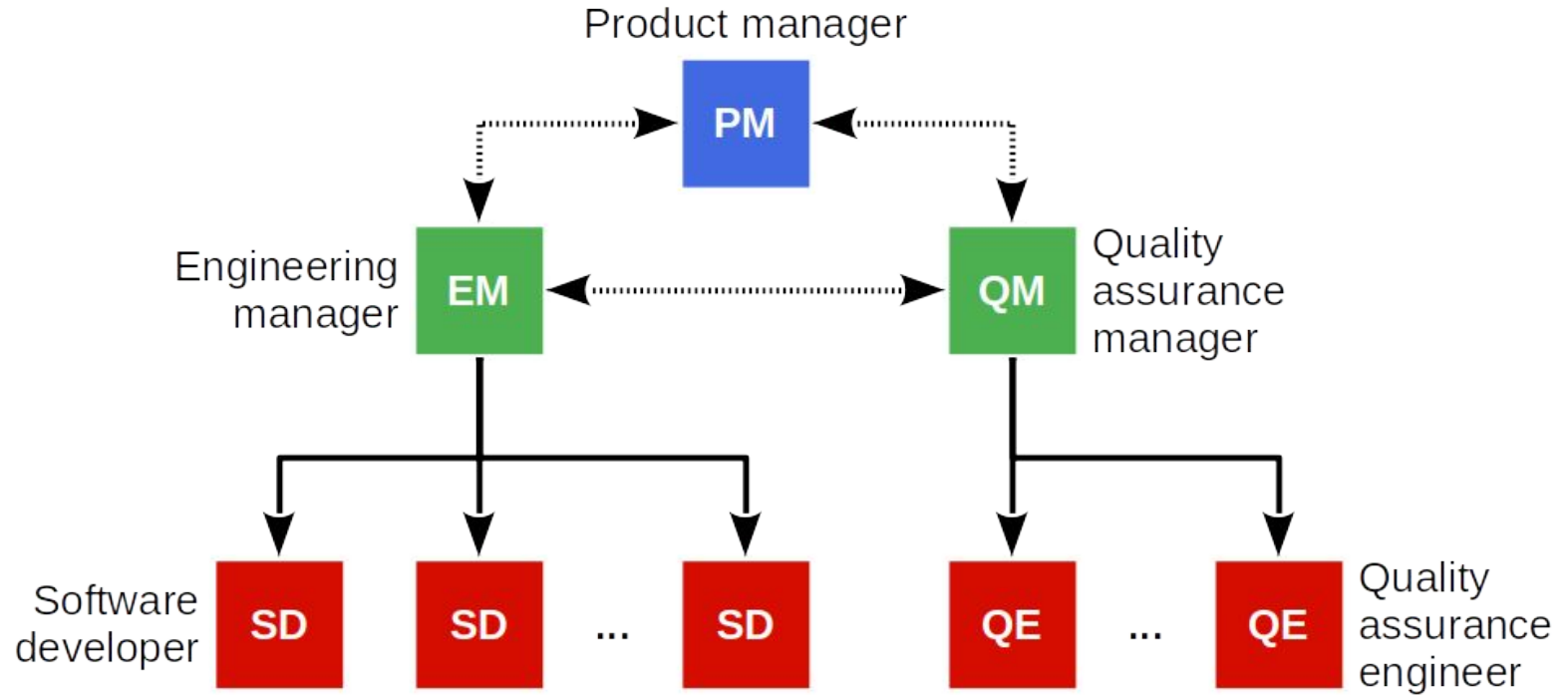
A project manager is responsible for

- Planning, managing, and delivering the project results to clients

A software developer is responsible for

- Implementing the functionality as requested by the project manager

# Traditional Software Product Organization (Vendor)





# Job Descriptions in Software Product Development

---

A product manager is responsible for

- What needs doing

An engineering manager is responsible for

- Who gets to do it and when

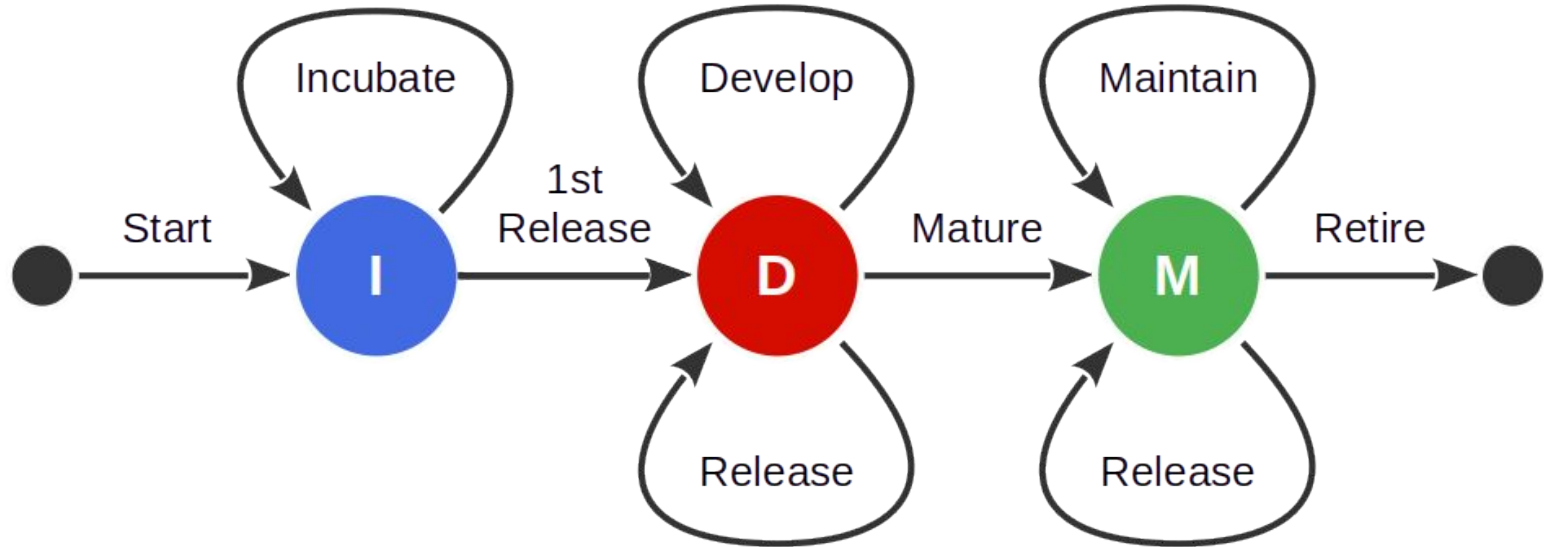
A software developer is responsible for

- How it gets done and how long it will take

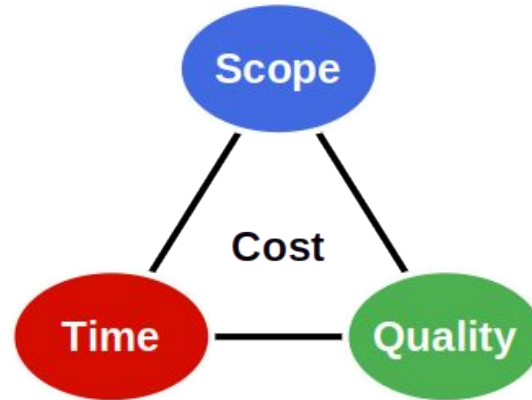
Quality assurance is responsible for

- Ensuring that the product meets the expectations

# Basic Software Product Life-Cycle



# The Magic Triangle (“Pick Two”)



## **2. Plan-Driven Development**

---

# Basic Plan-Driven Development



# Video From “The Pentagon Wars” [1]



[1] See <https://youtu.be/r0op8e0LuoU>

# Video Lessons

---

## Stakeholders problems

- Multiple stakeholders with conflicting interests
- Meddling stakeholders intervening into the process

## Requirement problems

- Inconsistent requirements (poor quality assurance)
- Changing requirements (wandering focus, long project)
- Feature creep (from troop carrier to tank)

## Product problems

- Cost explosion due to lack of focus, rework
- Unclear market and wandering purpose

# The Waterfall Model [1]

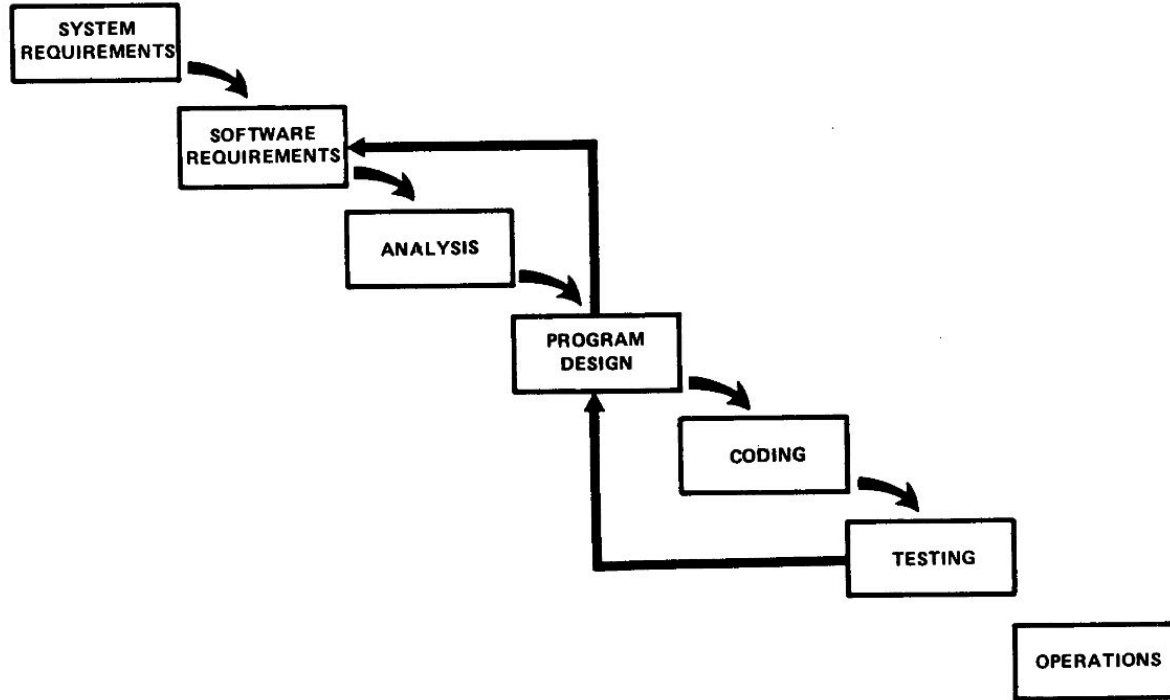


Figure 4. Unfortunately, for the process illustrated, the design iterations are never confined to the successive steps.

[1] Royce, W. W. (1970). Managing the development of large software systems. Proceedings of IEEE WESCON. Los Angeles, 328-388.



**Phases  $\neq$  Activities**

# **3. Agile Methods**

---

# Agile Methods

Agile methods are a category of software development methodologies

- Defined in opposition to plan-driven development
- Driven by consultants as a significant business opportunity

The key idea of agile methods is to have a fast feedback loop

- Steer, don't plan and blindly execute
- Codified as the agile manifesto

Examples agile methodologies

- Scrum, XP, the Crystal Methods, Feature Driven Development

# Principles of the Agile Manifesto [1]

---

## Individuals and interactions

- Over processes and tools

## Working software

- Over comprehensive documentation

## Customer collaboration

- Over contract negotiation

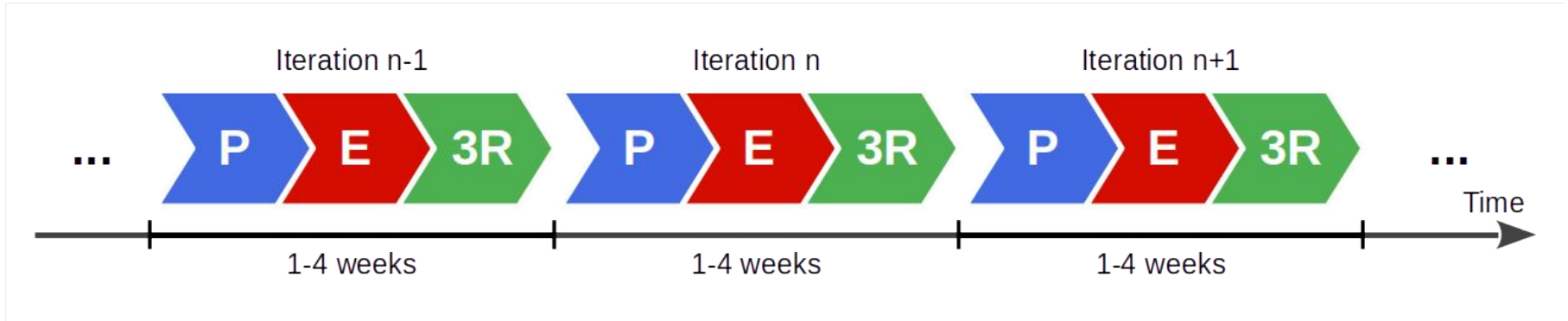
## Responding to change

- Over following a plan

[1] See <https://agilemanifesto.org>

# Agile Development Process

- Succession of equal-length iterations (“time-boxes”, “sprints”)
- Intervention points are during planning and review
- User feedback only available during review



P = Planning  
E = Execution  
3R = Review, release, and retrospective

# Benefits of Fast Feedback Loops

---

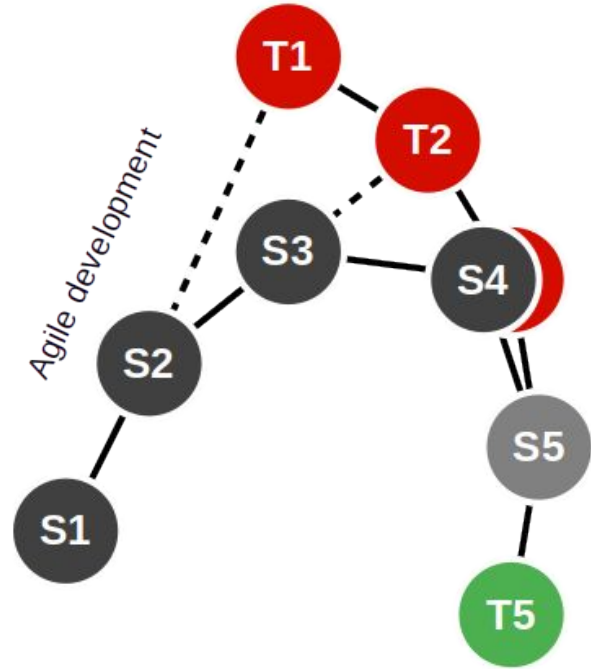
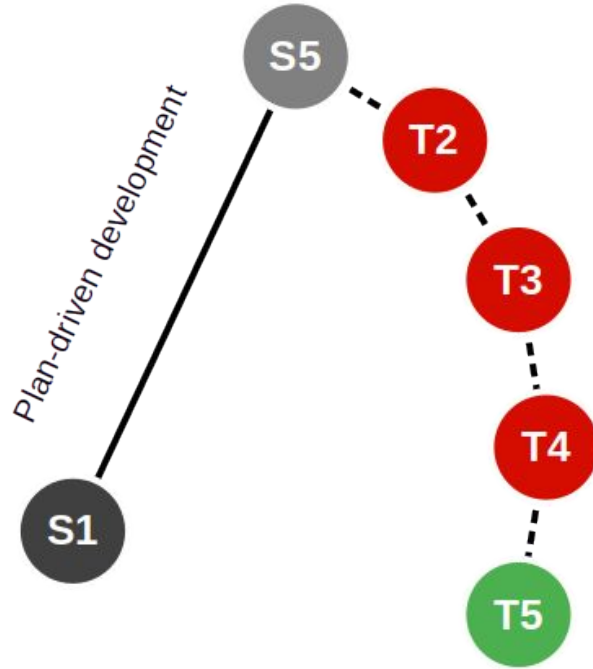
## Short iterations

- Short iterations lead to focus on high-value features first
- Established well-worn rhythm is sustainable, avoids burnout
- Partial functionality is better than none

## User feedback

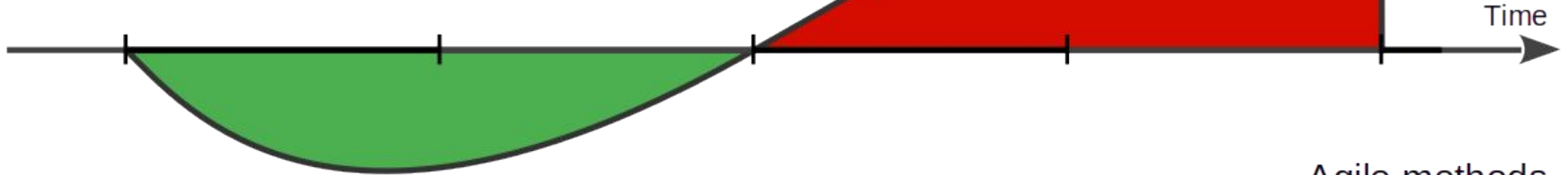
- User feedback helps steer project or product to meet user needs
- Feedback loop ensures that problems surface early
- Feedback helps recognize and realize new innovative features

# Plan-Driven vs. Agile Processes

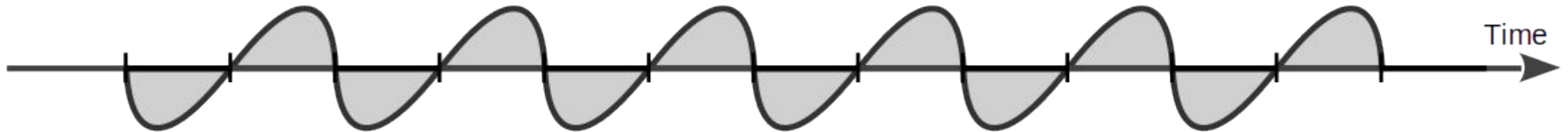


# Plan-Driven vs. Agile Work Rhythms

Plan-driven methods  
Long iterations



Agile methods  
Short iterations





# Do Agile Methods Lead to Cowboy Coding?

---

**Agile methods are high discipline**

## 4. Scrum

---

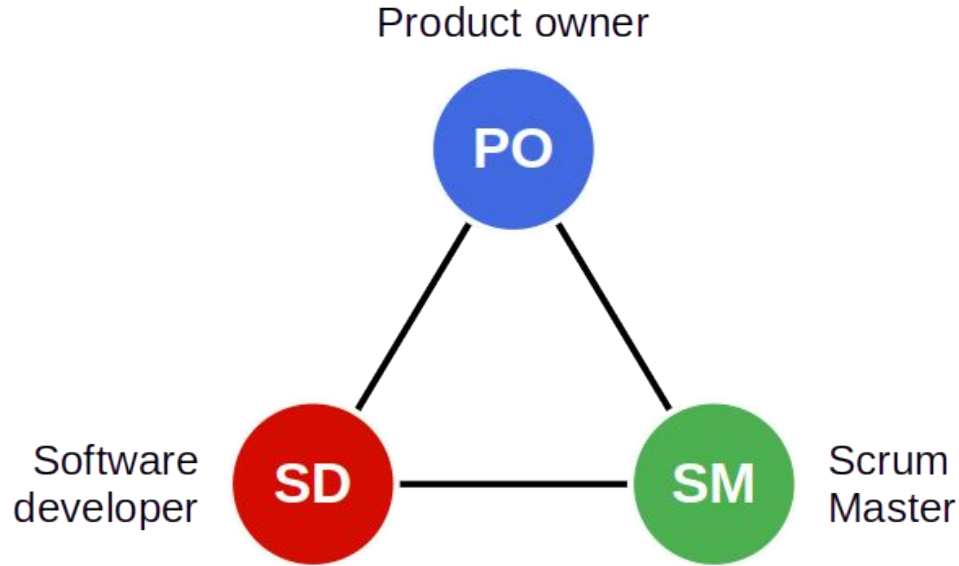
# Scrum [1]

---

Scrum is an agile method (framework) invented around 1993, 1995

- Has a minimal (agile) process model
- Is applicable to any domain, not just software development

# Scrum Roles / Scrum Team [1]



# Committed vs. Involved Roles

---

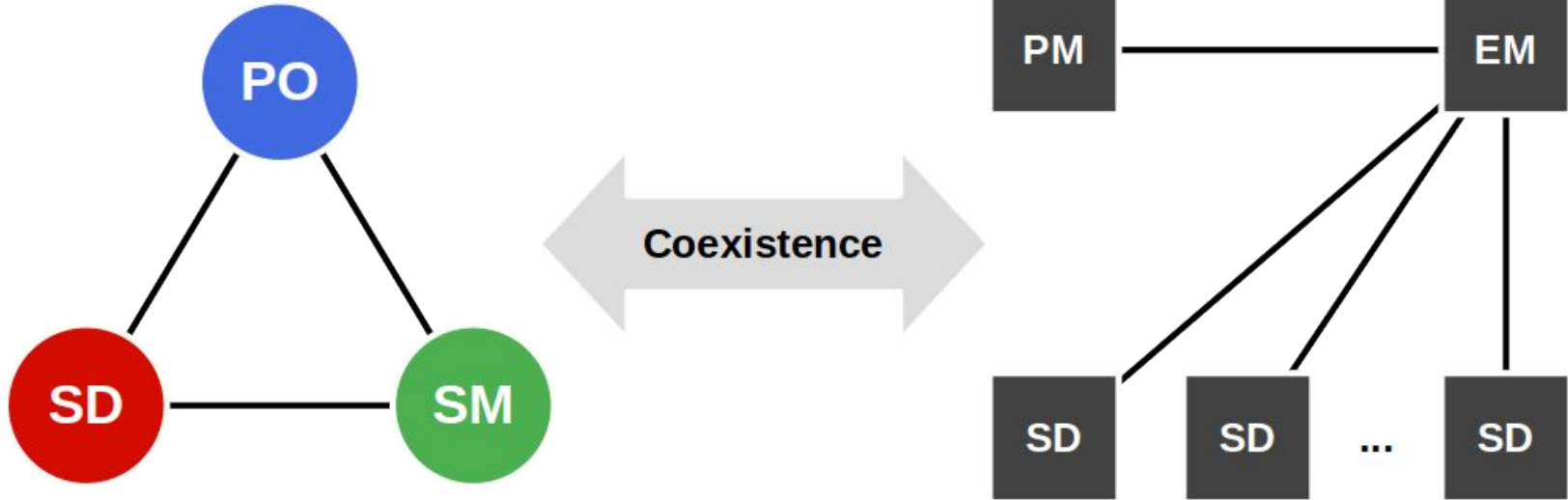
## Committed roles

- Product owner
- Software developer(s)
- Scrum Master

## Involved roles

- Customer
- Sponsor / funder
- Regulators
- ...

# Mapping Roles to Posts



# Roles / Posts Correspondence

Scrum	Custom projects (consulting firms)	Product development (vendors)
Product owner	Project manager	Product manager Engineering manager
Software developer	Project manager Software developer	Software developer Engineering manager Quality assurer
Scrum Master	Project manager	Engineering manager

# Terms (The Scrum Terminology Mess)

Scrum	Custom projects	Product development
Product owner	Business analyst, Requirements engineer	Product manager
Product goal	Project mission [1]	Product vision [1]
Product backlog	Requirements specification	Product requirements document (PRD)

[1] This resolution is specific to AMOS, though the terms are generally known and used



# Scrum Scope / Time Horizons

---

Scrum proper covers

1. Day
2. Sprint (weeks)

Arguably implied by Scrum

3. Releases (months)
4. Project/product (years)

Further evolutions e.g. SAFe cover

5. Product life-cycle (years)
6. Portfolio

# Daily Scrum (a.k.a. Daily Standup)

The **daily scrum** is

- A daily status meeting to sync on problems and upcoming work
- To be kept as short as possible

Committed parties are mandatory

- Only committed parties may speak
- Everyone else is optional

Discussions are not allowed, but

- Can be had one-on-one after the meeting
- Scrum Master will follow-up on problems

# Daily Scrum



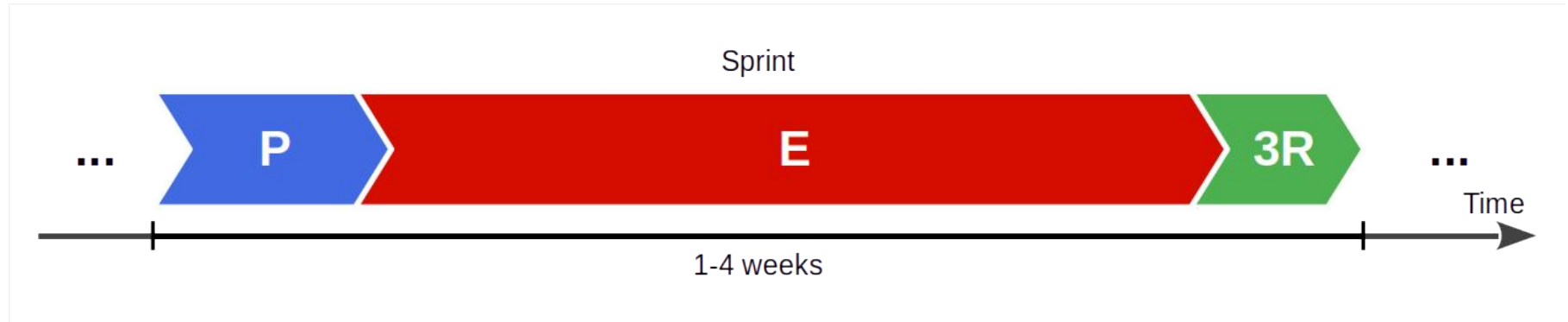
# Daily Plank



# Scrum Sprint

A sprint is Scrum's iteration; it is an equal-length time-box

It is a highly structured process with defined feedback points



# Increment of Value

---

An increment of value is

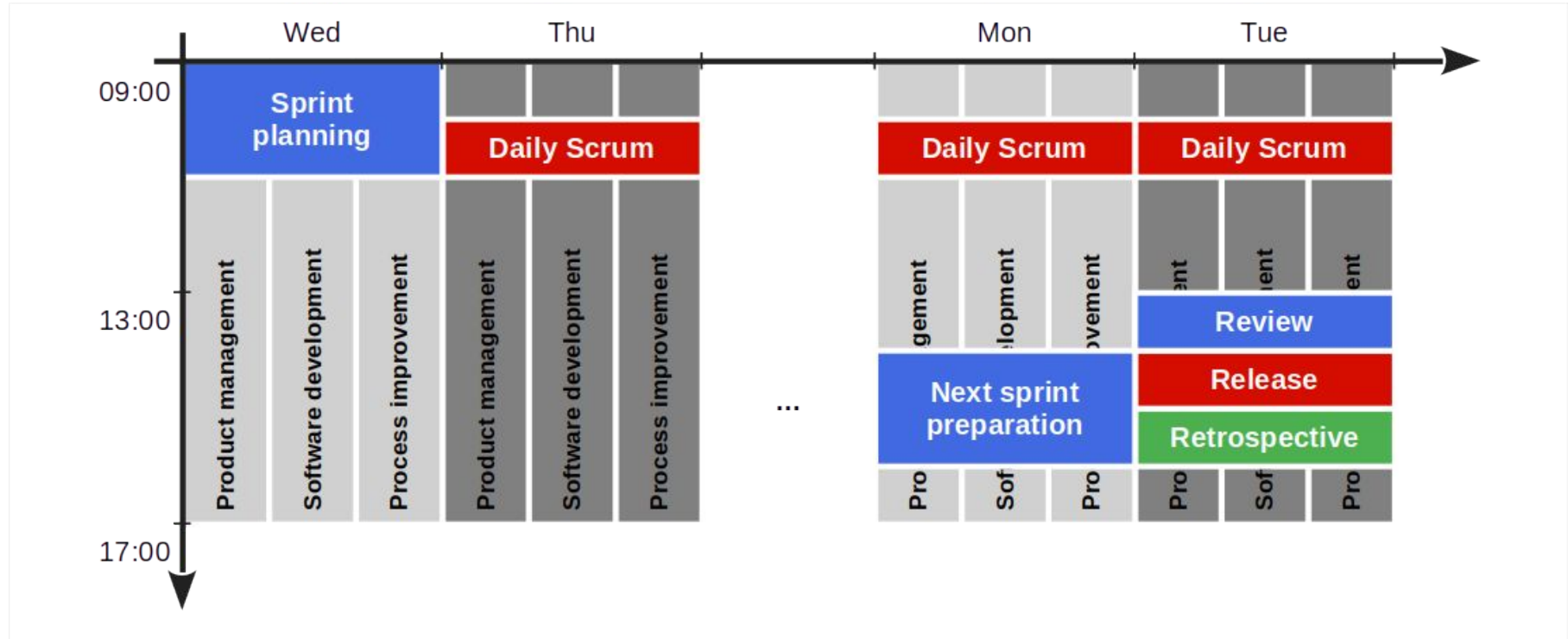
- Some added value to current or new artifacts

Increments of value are mainly provided by

- Sprints (via sprint release)

Basically, anything that has a definition of done

# Sprint Structure



# Sprint Meetings

## 1. Next sprint preparation

- a. Product owner and senior developer groom the product backlog

## 2. Sprint review

- a. Team reviews this sprint's results, signs off on them

## 3. Sprint release

- a. Team decides on sprint release

## 4. Sprint retrospective

- a. Team reviews process, commits to improvements

## 5. Sprint planning

- a. Team discusses upcoming work, commits to it

## 6. Daily Scrum

- a. Team members update each other on work progress

The AMOS  
team meeting



# Sprint Workstreams

---

Product management; the product owner

- Builds and grooms the product backlog
- Answers questions to developers

Software development; software developers

- Break down backlog items into tasks, self-organize
- Design and implement sprint backlog items

Process improvement; the Scrum Master

- Observes problems and opportunities
- Facilitates impediments resolution and improvements

## **5. Build Process Review**

---

# Milestone: Build Process Review

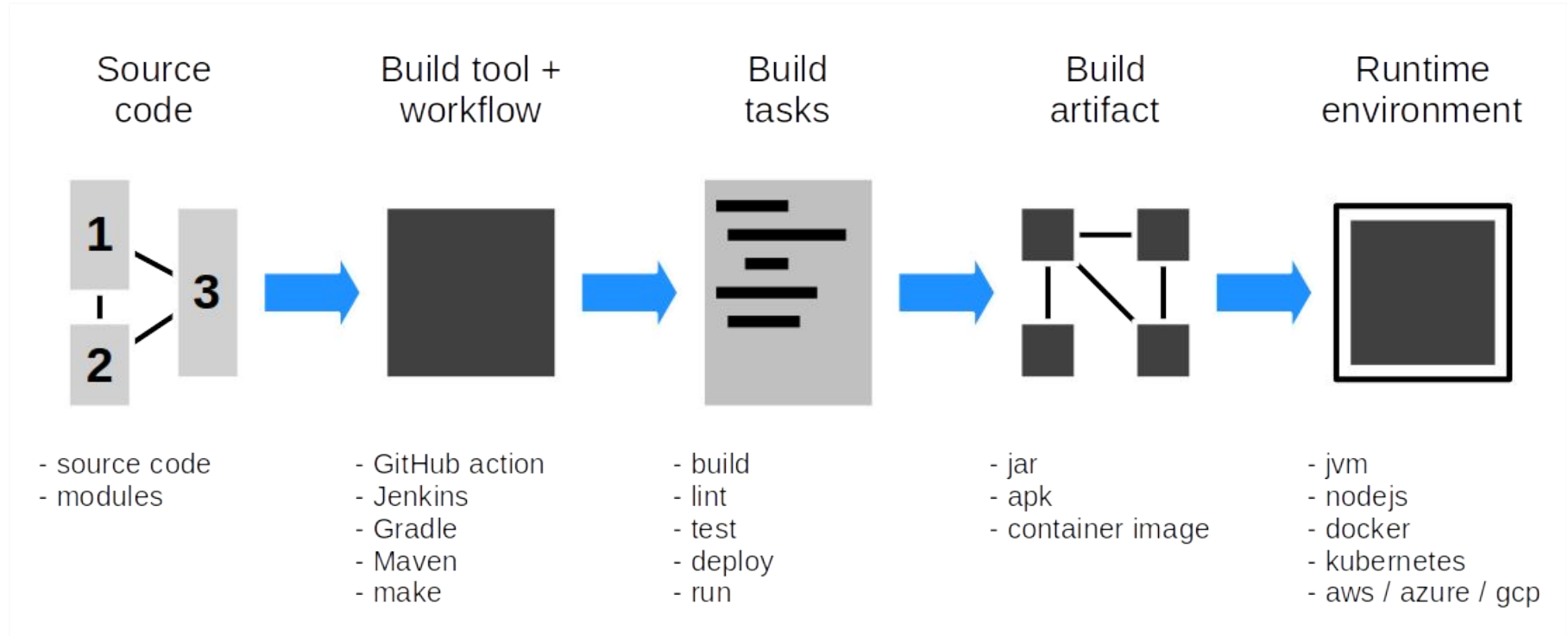
---

We expect you to demo a working build and run process

- One command to trigger and perform a full build
- One command to start the software for demoing purposes
- All of this independently of a particular person's machine
- We will call arbitrarily on people to show this

Feel free to coordinate with and learn from other teams

# How to Discuss Build Input, Pipeline, and Artifacts



# Summary

---

1. Software development
2. Plan-driven development
3. Agile methods
4. Scrum
5. Build process review

# Thank you! Any questions?

---

[dirk.riehle@fau.de](mailto:dirk.riehle@fau.de) – <https://oss.cs.fau.de>

[dirk@riehle.org](mailto:dirk@riehle.org) – <https://dirkriehle.com> – [@dirkriehle](#)

# Legal Notices

---

## License

- Licensed under the [CC BY 4.0 International](https://creativecommons.org/licenses/by/4.0/) license

## Copyright

- © Copyright 2009, 2024 Dirk Riehle, some rights reserved

# Megawoosh! [1]



[1] See <https://www.youtube.com/watch?v=D7rbiLNf-JI>



# Video Lessons [1]



[1] See [https://youtu.be/\\_n065KE00J0](https://youtu.be/_n065KE00J0)