Der kleine Unterschied

Wie sich Code Reviews zwischen ‚normalen' Projekten und Open Source Projekten unterscheiden und was wir daraus lernen könnten…

Ronald Brill, GEBIT Solutions, 2025

GEBIT Solutions

Über mich

Ronald Brill
Principal

GEBIT Solutions

https://github.com/rbri

HtmlUnit

- @HtmlUnit
- https://github.com/HtmlUnit
- https://github.com/SeleniumHQ/htmlunit-driver
- Mozilla Rhino, ….

BingoBongoTesting

- www.bingobongotesting.org/

WETATOR
SMART WEB APPLICATION TESTING

- www.wetator.org/

Der kleine Unterschied

Wie sich Code Reviews zwischen ‚normalen' Projekten und Open Source Projekten unterscheiden und was wir daraus lernen könnten…

Ronald Brill, GEBIT Solutions, 2025

GEBIT Solutions

Nicht kleckern….

Der Einsatz von Reviews führt zu einer **deutlichen Reduktion von Fehlern**. Laut Capers Jones' Studien von ca. 12.000 Projekten führen **Requirements Reviews** zu einer Reduktion der zu erwartenden Fehler um 20 % bis 50 %, Top-level **Design Reviews** zwischen 30 % und 60 %, detaillierte funktionelle **Design Reviews** zwischen 30 % und 65 % und detaillierte logische **Design Reviews** zwischen 35 % und 75 %.

Das entspricht in etwa der Effektivität von Systemtests (25 % bis 65 % aller Fehler).

Capers Jones, Chief Scientist Emeritus: *Software Quality in 2002: A Survey of the State of the Art.* Software Productivity Research an Artemis company, 23. Juli **2002**, S. 56

Nicht kleckern….

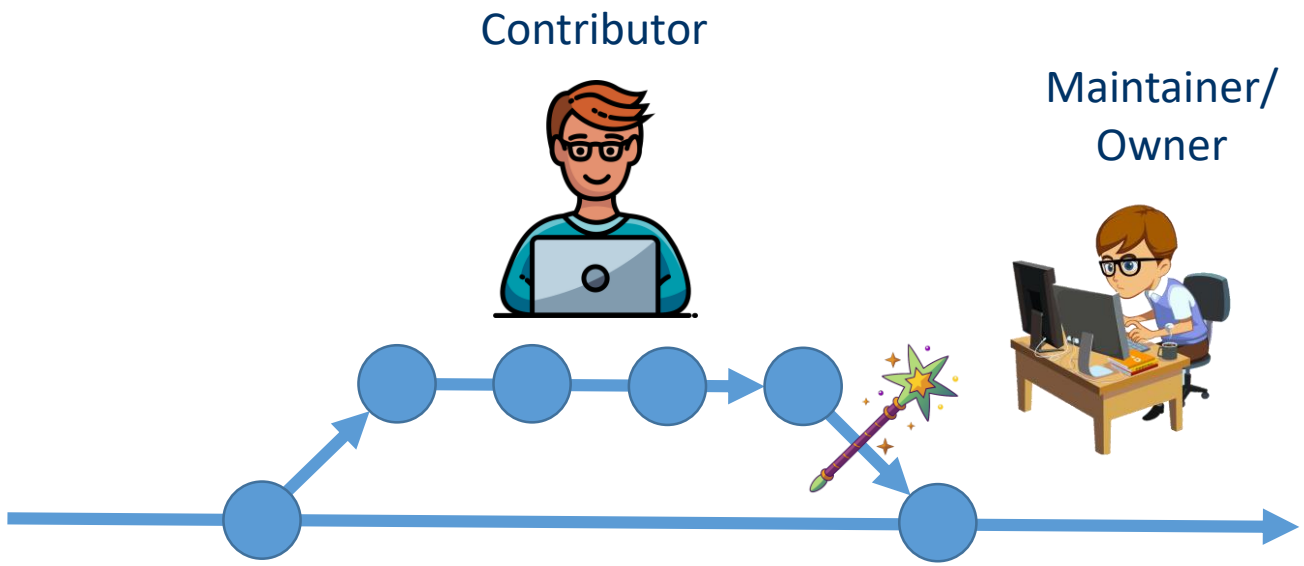
Der Einsatz von Reviews führt zu einer **deutlichen Reduktion von Fehlern**. Laut Capers Jones' Studien von ca. 12.000 Projekten führen **Requirements Reviews** zu einer Reduktion der zu erwartenden Fehler um 20 %, detaill ind detaill

Code Reviews

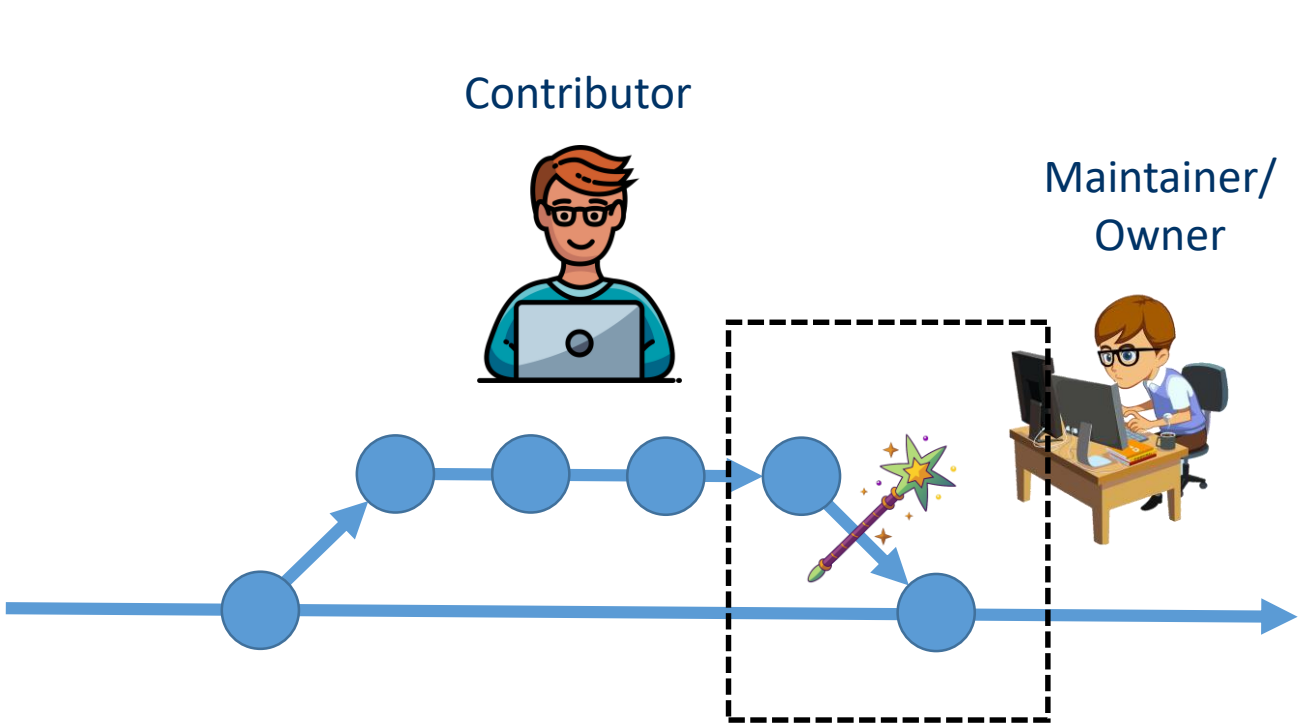
Das entspricht in etwa der Effektivität von Systemtests (25 % bis 65 % aller Fehler).

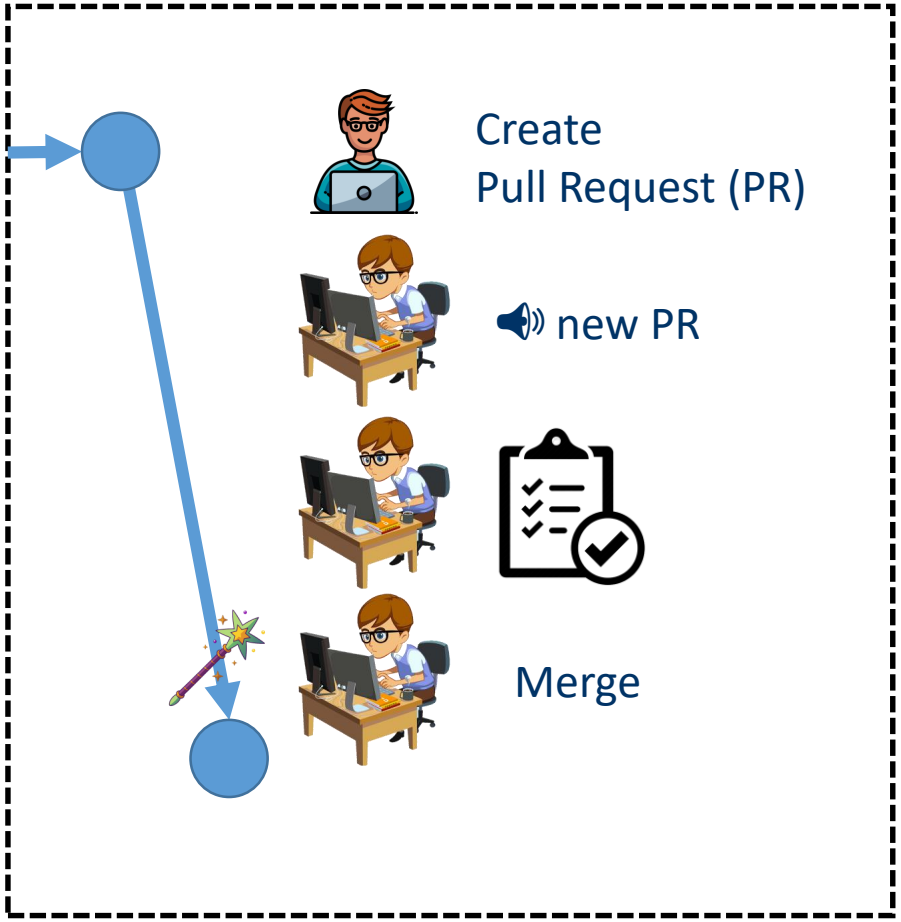
Capers Jones, Chief Scientist Emeritus: *Software Quality in 2002: A Survey of the State of the Art.* Software Productivity Research an Artemis company, 23. Juli **2002**, S. 56

Open Source - Code Reviews



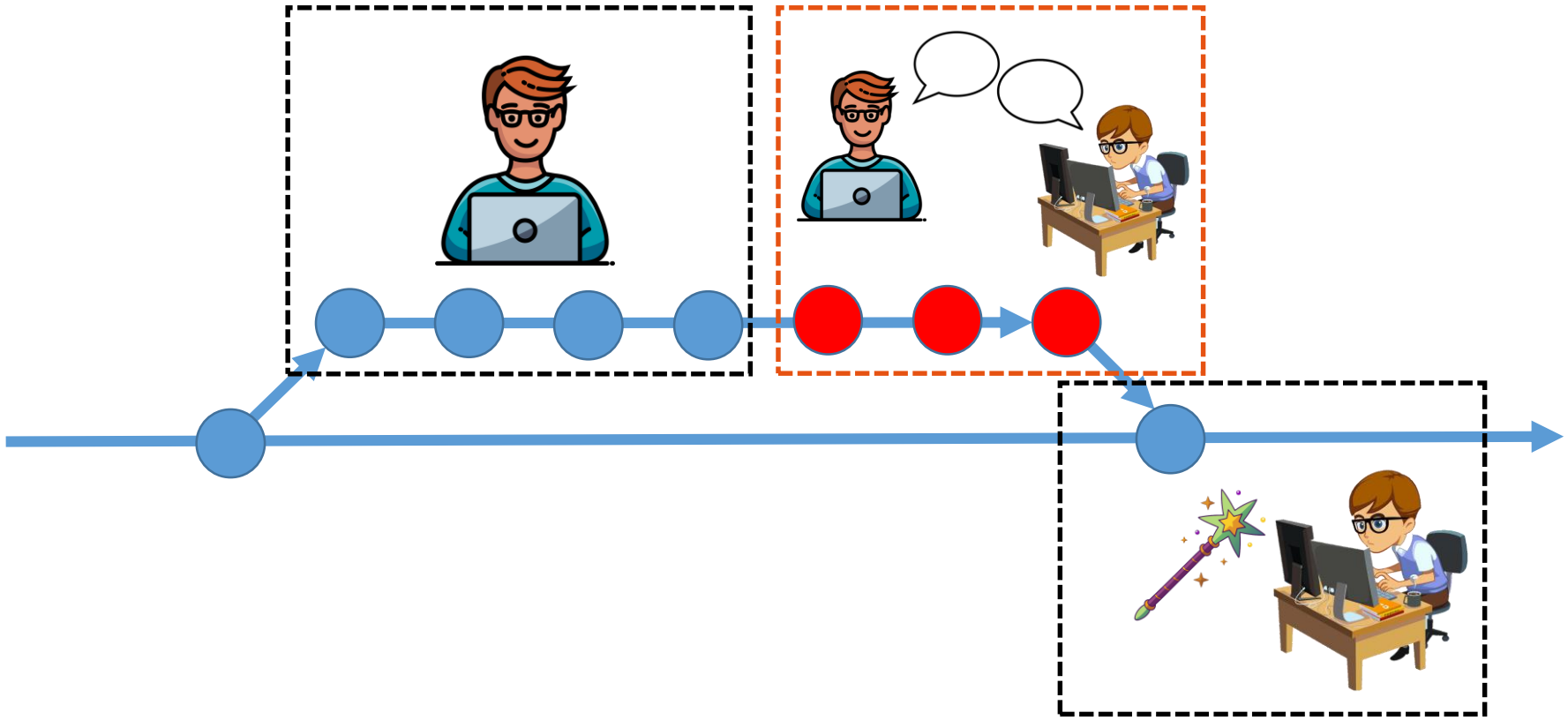
Open Source - Code Reviews





Open Source - Code Reviews

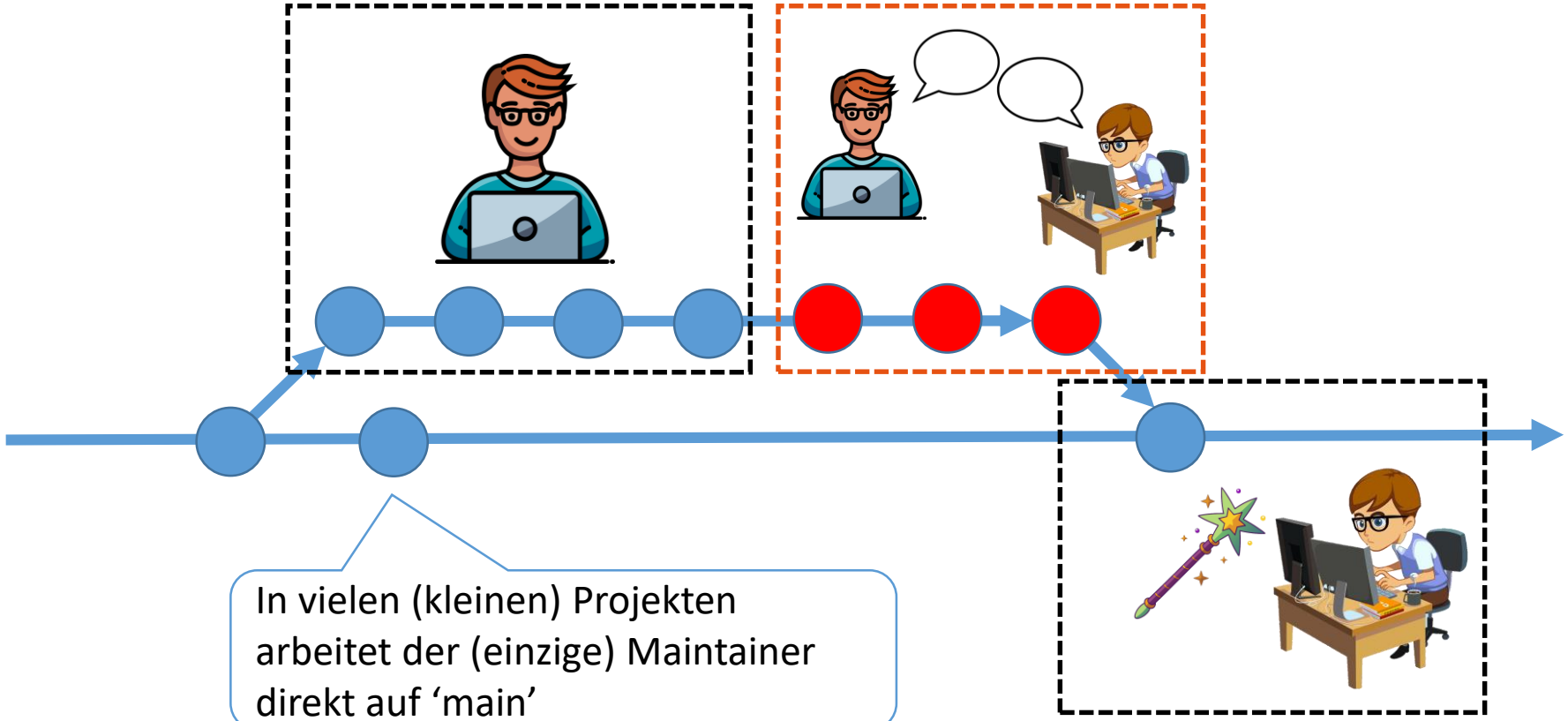
Review - Process



Open Source - Code Reviews

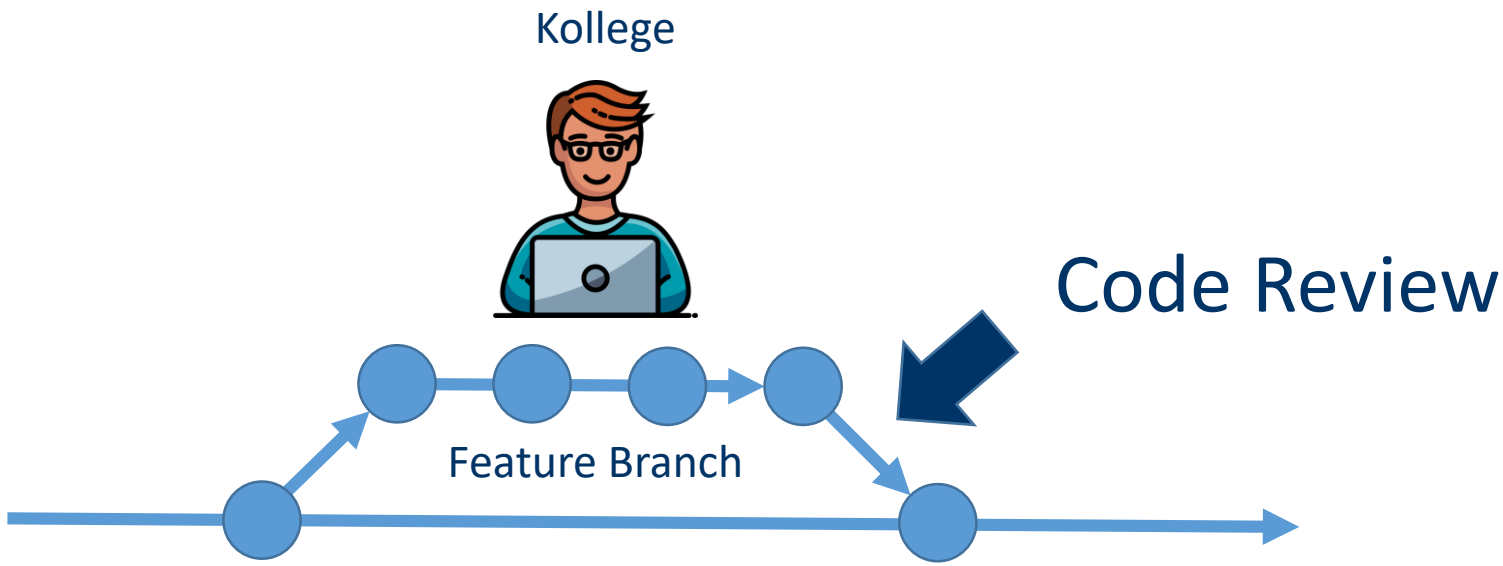
‘offline’
GitHub Tools

Review - Process

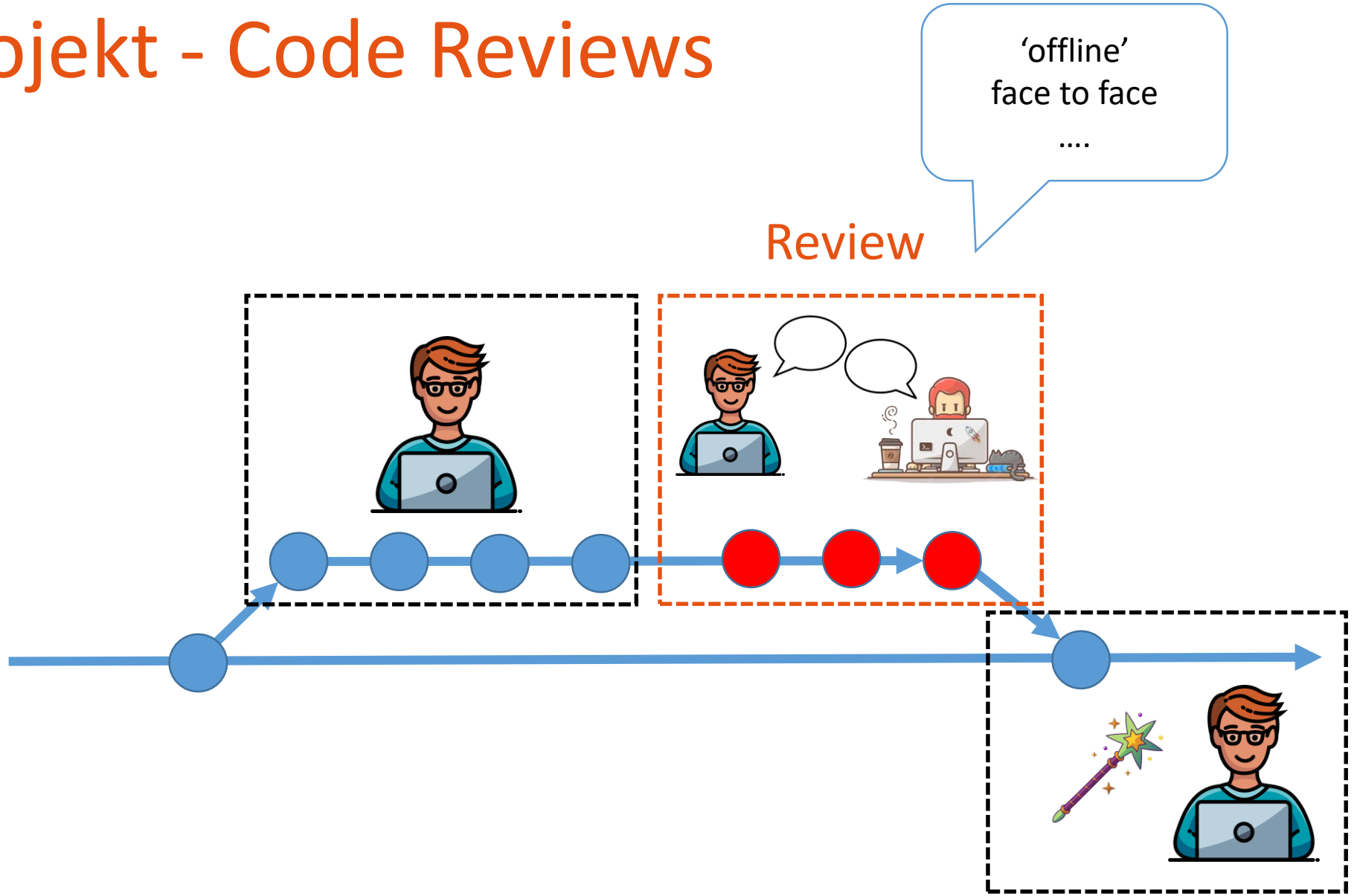


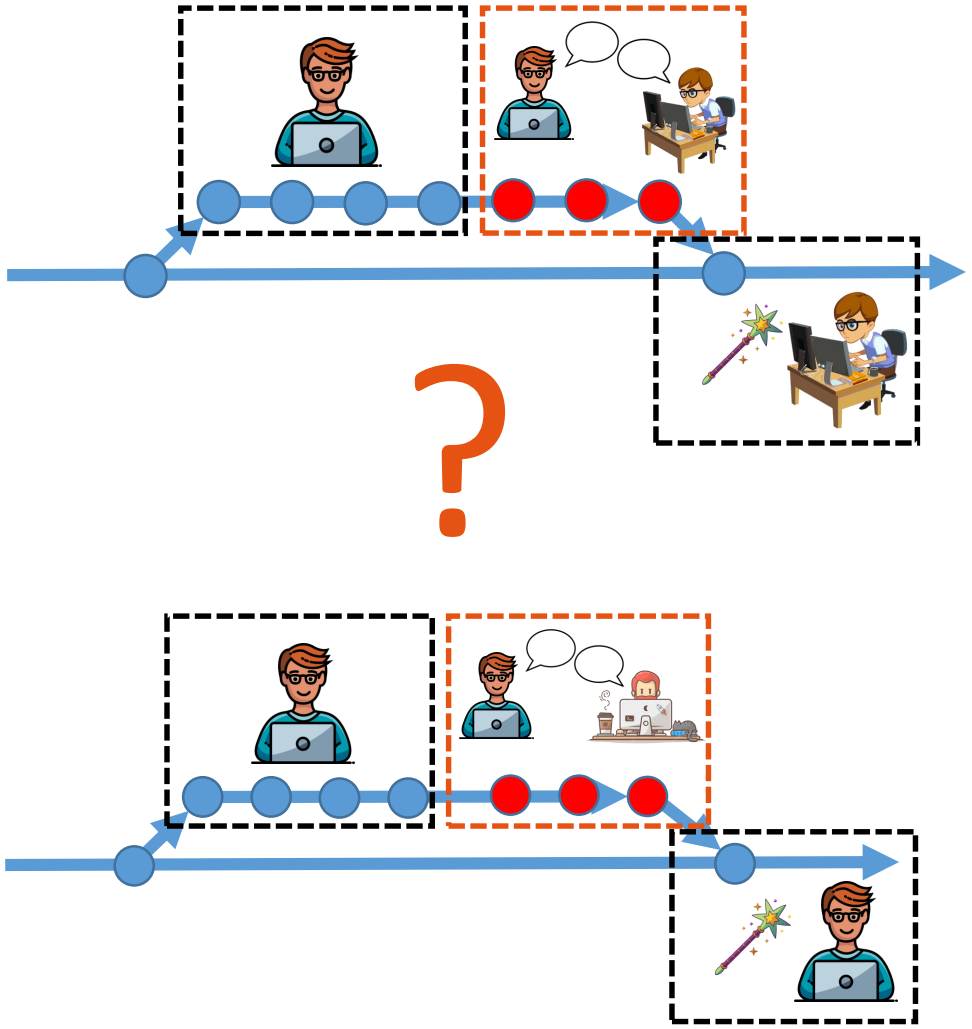
In vielen (kleinen) Projekten arbeitet der (einzige) Maintainer direkt auf ‘main’

Projekt - Code Reviews



Projekt - Code Reviews





Code Reviews

Ziel

Open Source - Ziel

Perfektion

- Code
- fehlerfrei
- Performance
- backward compatibility
- Security
- …

Projekt - Ziel

Business

- Produkt

Projektleiter

- Kosten

Entwickler

- Perfektion

Tester

- Fehlerfrei

….

Von Allem ein wenig…

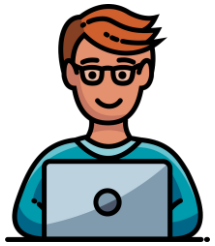
Code Reviews

Teilnehmer / Rollen

Open Source - Rollen

Maintainer/
Owner

Contributor



Potentiell

- pragmatisch
- konsistent
- motivierend
- …

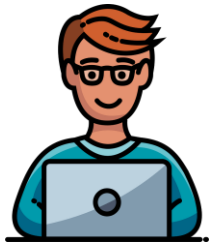
- demotivierend
- intransparent
- …

https://fr.pinterest.com/pin/351912459357165/

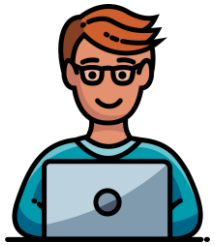
Projekt - Rollen

- Junior
- Senior
- KI
- …

Kollege



Kollege

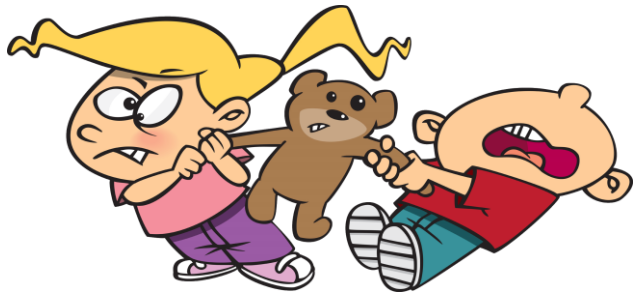


- Junior
- Senior
- KI
- …

Team

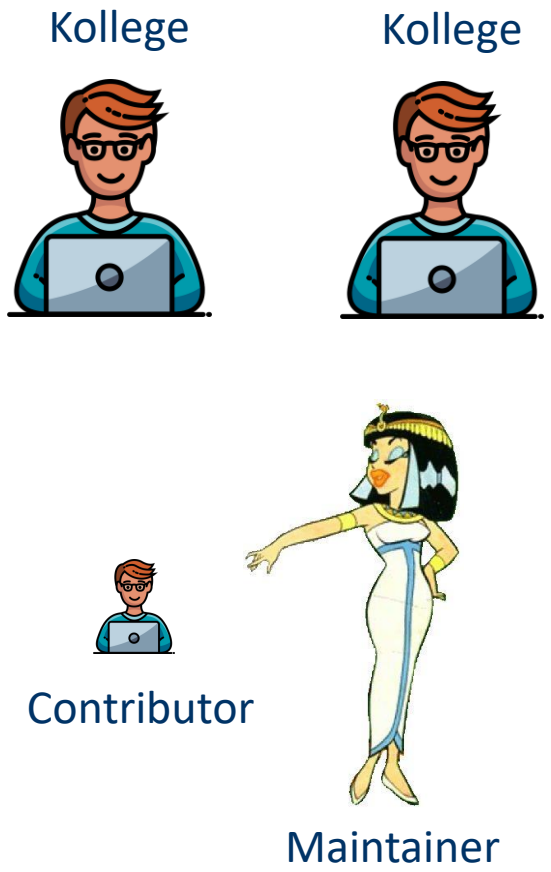
- Auswahl Reviewer
- zeitliche Planung
- Vorbereitungsaufwand für Reviewer
- immer alles reviewen?
- ….

Code Reviews



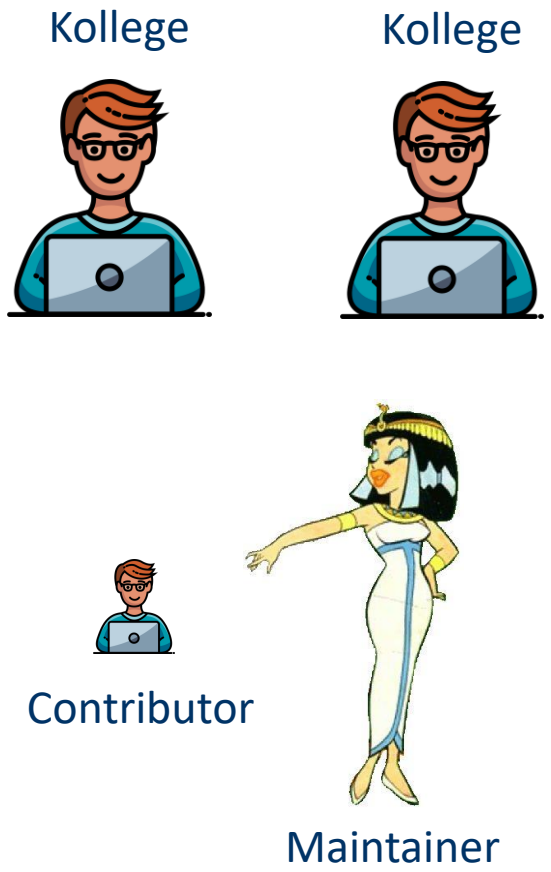
Zwischenmenschliches

Zwischenmenschliches



- unvoreingenommen
- taktvoll
- kritikfähig / Fehlerkultur
- psychologische Aspekte
- Teamzusammensetzung
- positiven Erfahrung für den Autor
- Lernen / Lehren
- ….

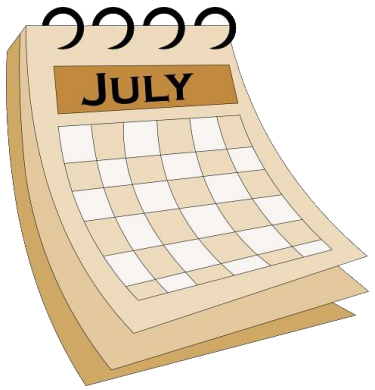
Zwischenmenschliches



- The Infinite Nitpick Loop: Death by a Thousand Cuts
- The Architectural Ambush: Rejecting the Approach After Implementation
- The Impossible Standard: Demanding Production-Ready Perfection in Experimental Code
- The Moving Goalpost: Changing Requirements During Review
- The Style Over Substance Critique: Enforcing Personal Preferences as Objective Standards
- The Cryptic Drive-By: Vague Criticisms Without Solutions
- ….

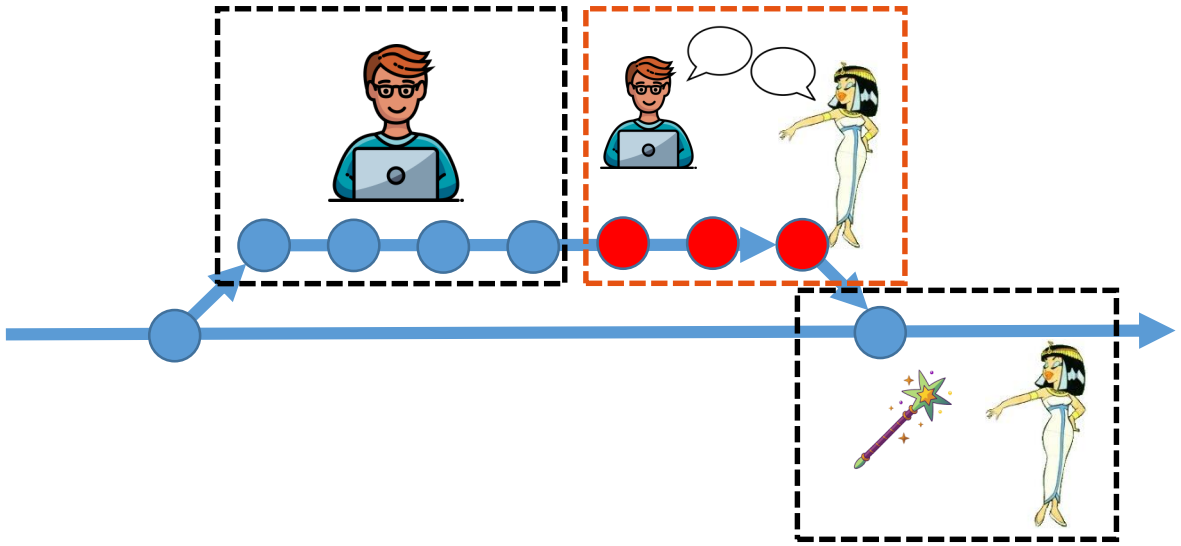
https://blog.stackademic.com/the-dark-psychology-of-code-reviews-6-ways-theyre-designed-to-crush-your-spirit-6d157d7759d2

Code Reviews



Zeitlicher Ablauf

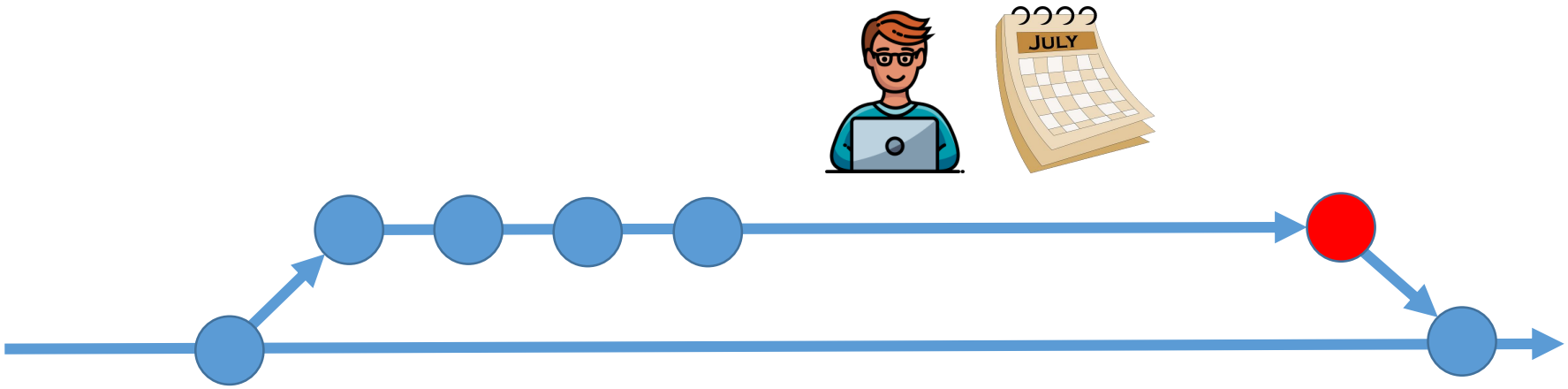
Open Source – Zeitlicher Ablauf



- Arbeit komplett beim Contributor
- Contributor hat keine Entscheidungsbefugnis

- keine Planung möglich (insbesondere keine zeitliche)
- ….

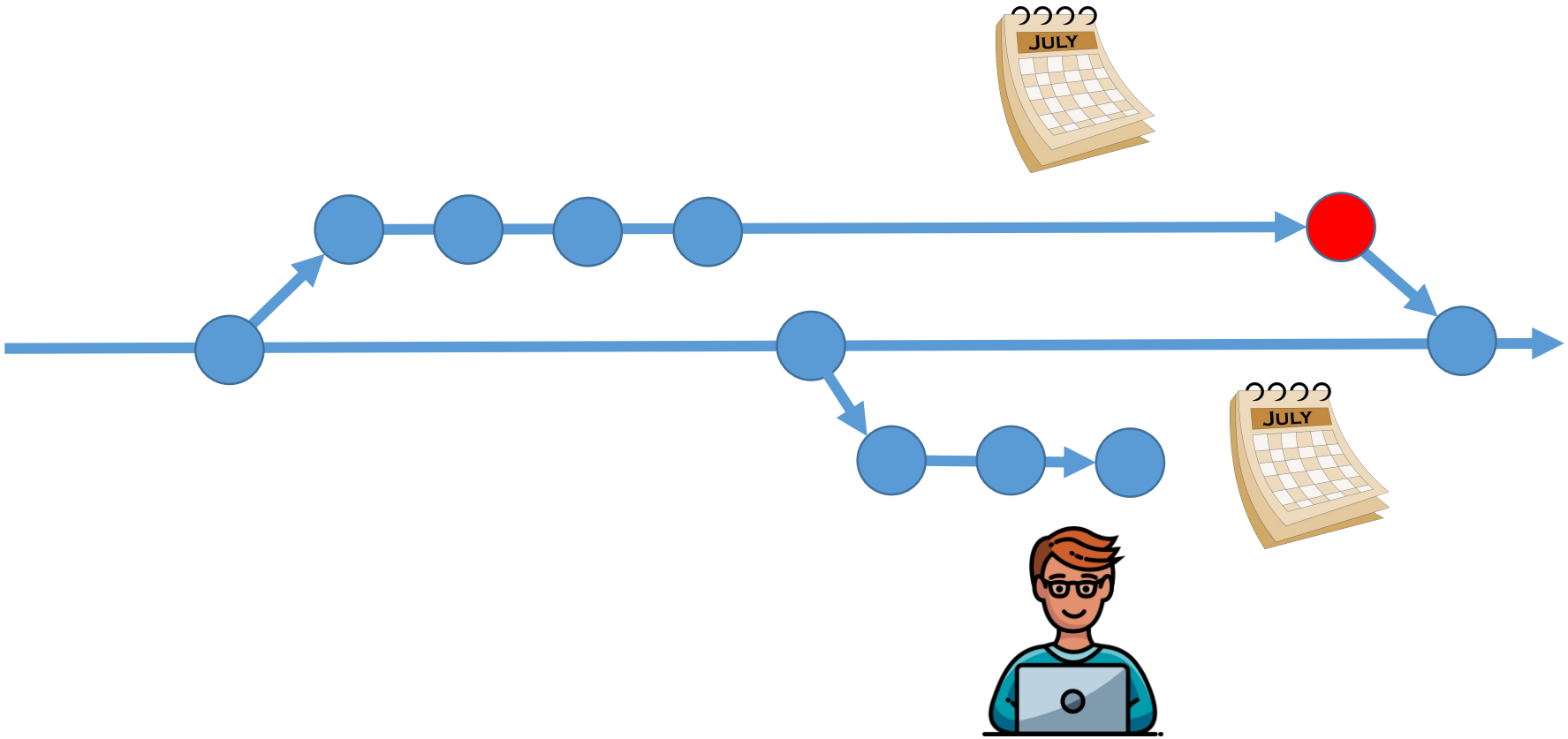
Projekt - Zeitlicher Ablauf



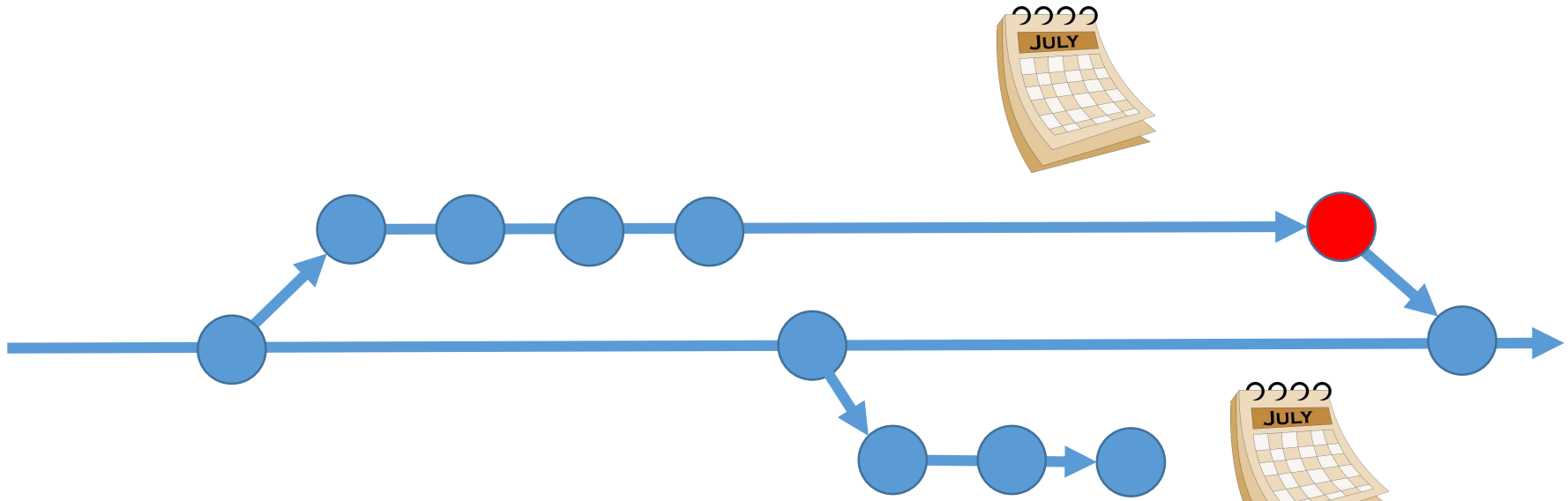
Review wird eingeplant -> Warten

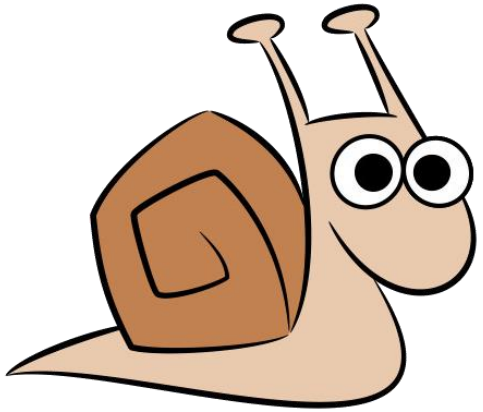
Häufung von Reviews am Sprintende

Projekt - Zeitlicher Ablauf

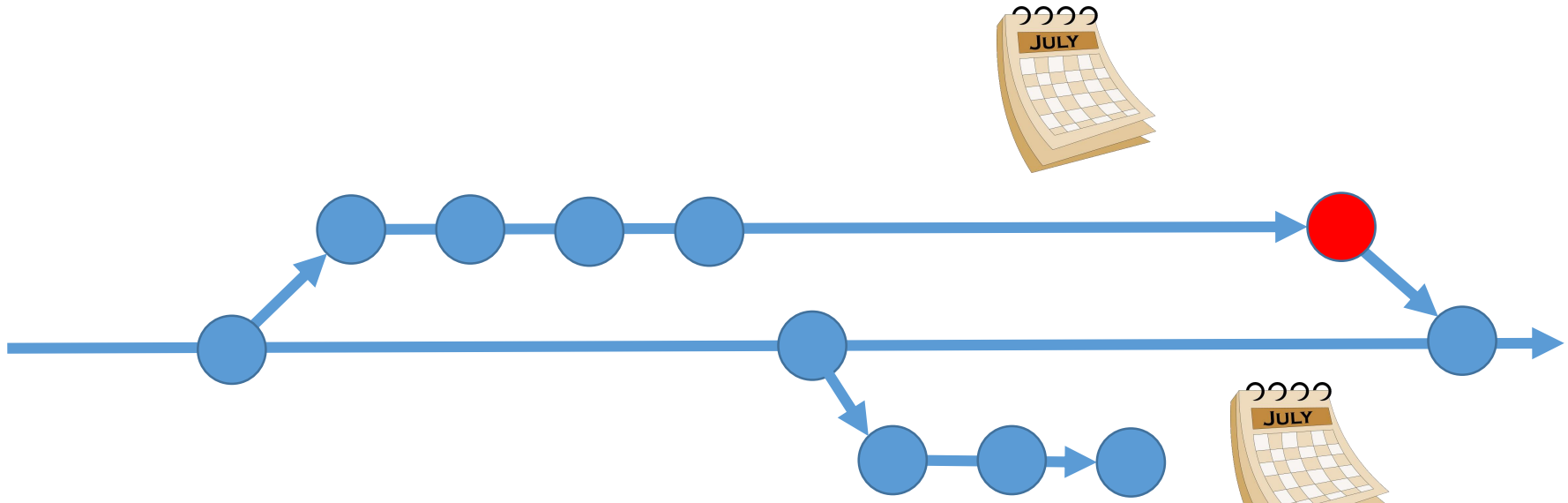


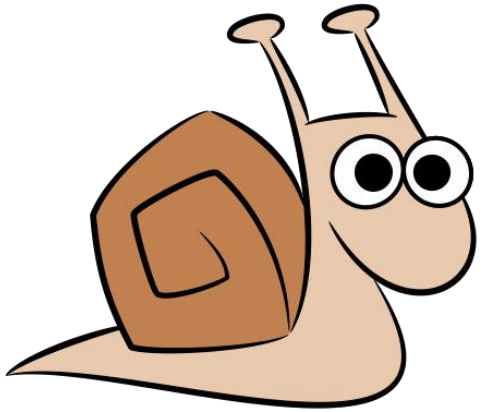
Projekt - Zeitlicher Ablauf





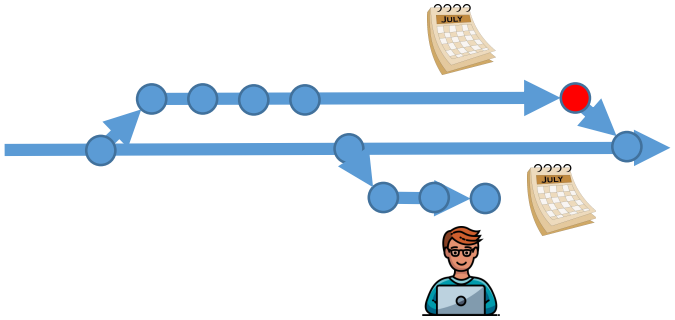
Projekt - Zeitlicher Ablauf



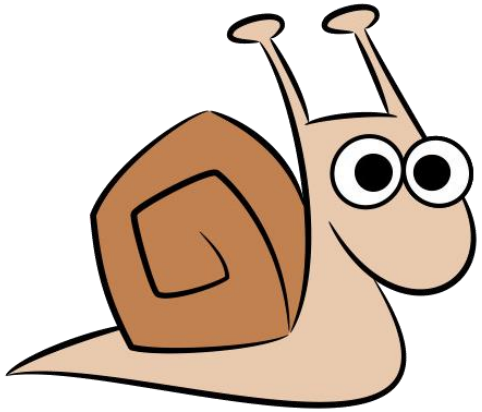


- bei vielen kleinen Änderungen wird es noch schlimmer

Projekt - Zeitlicher Ablauf



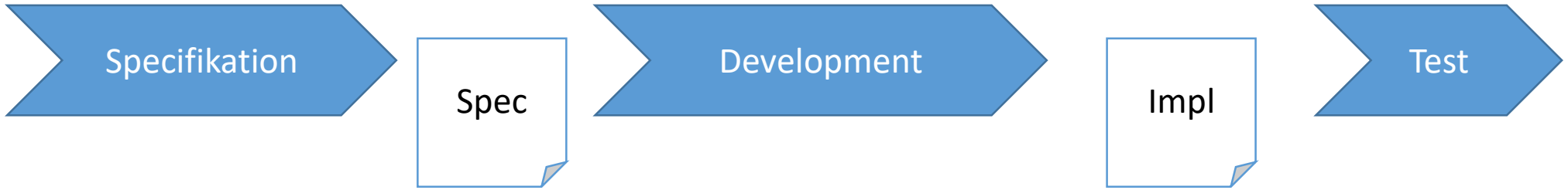
- implizite Motivation für größere Änderungen pro Review
  - komplexer, Fehler werden leichter übersehen
- Review – Lock
- fast-review policy



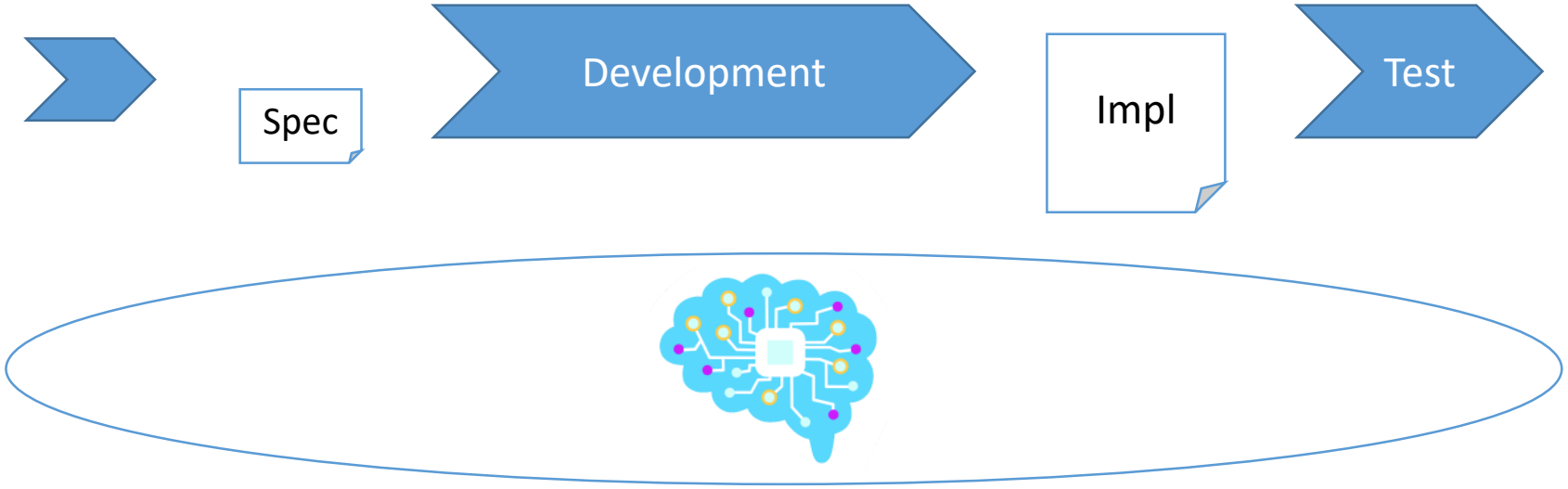
Code Reviews

Scope

Scope - Business Application

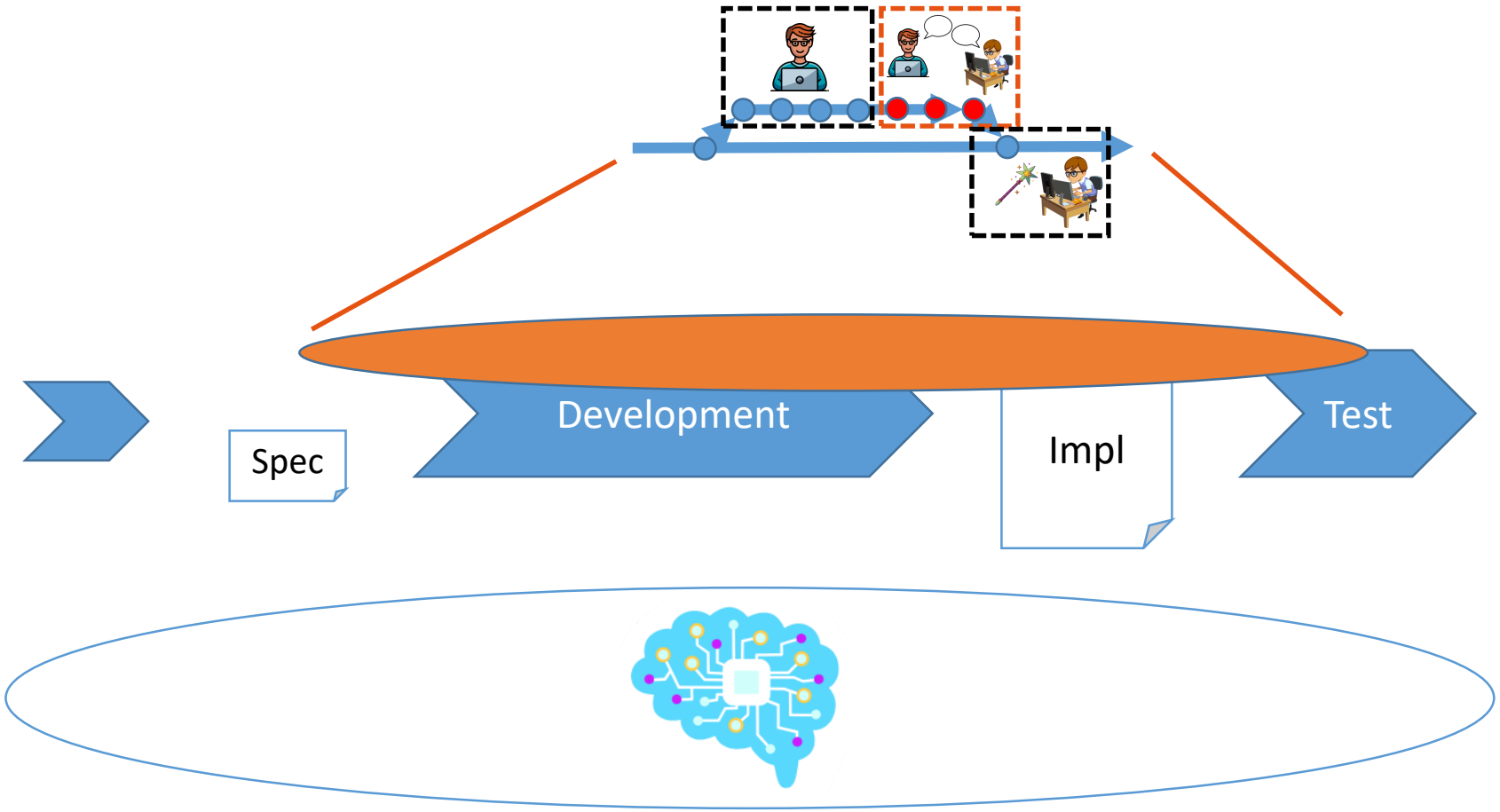


Open Source – Scope



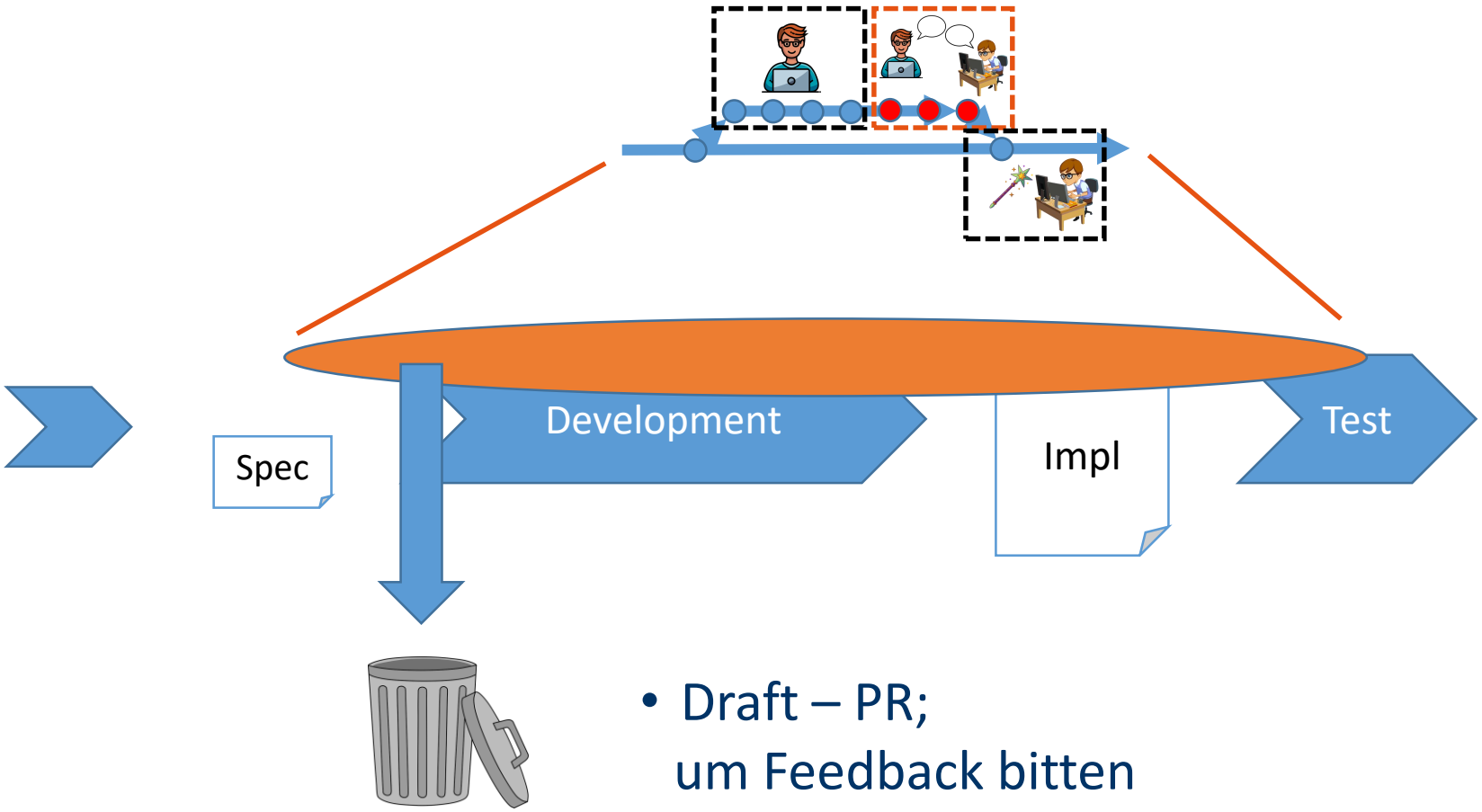
Personalunion

Open Source – Scope



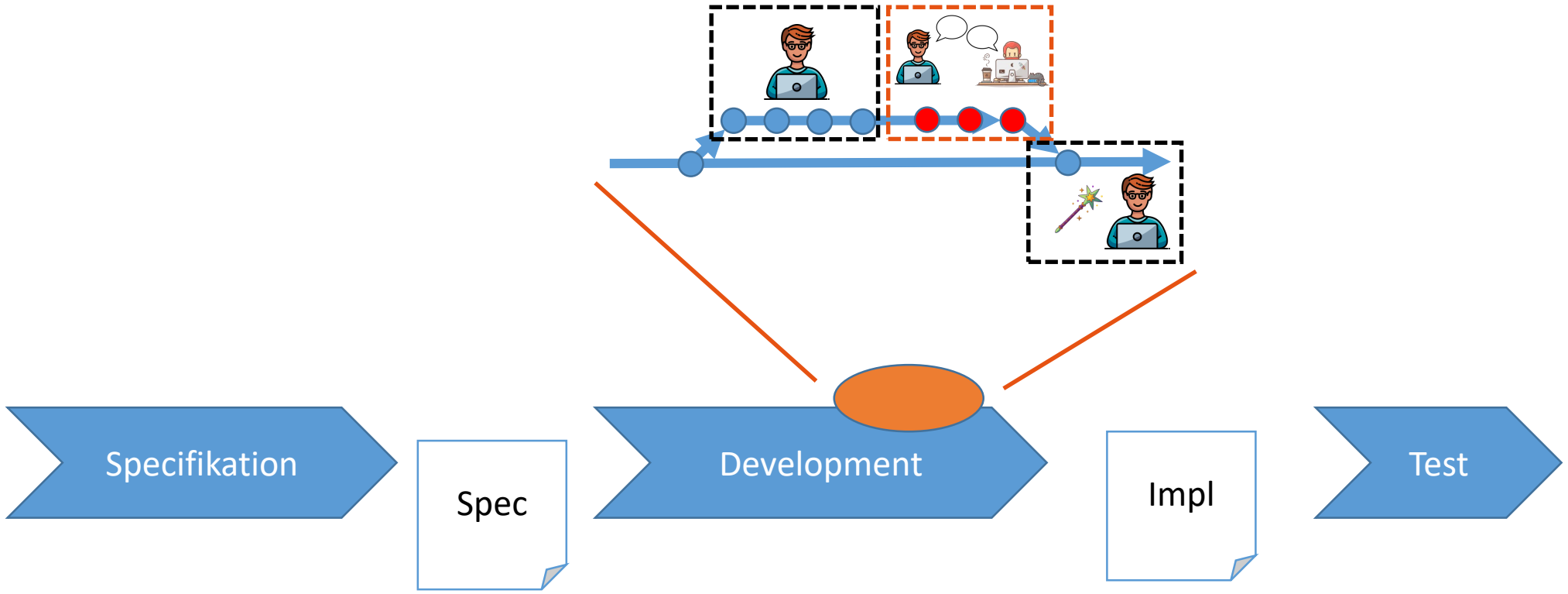
Personalunion

Open Source – Scope



- Draft – PR;
  um Feedback bitten

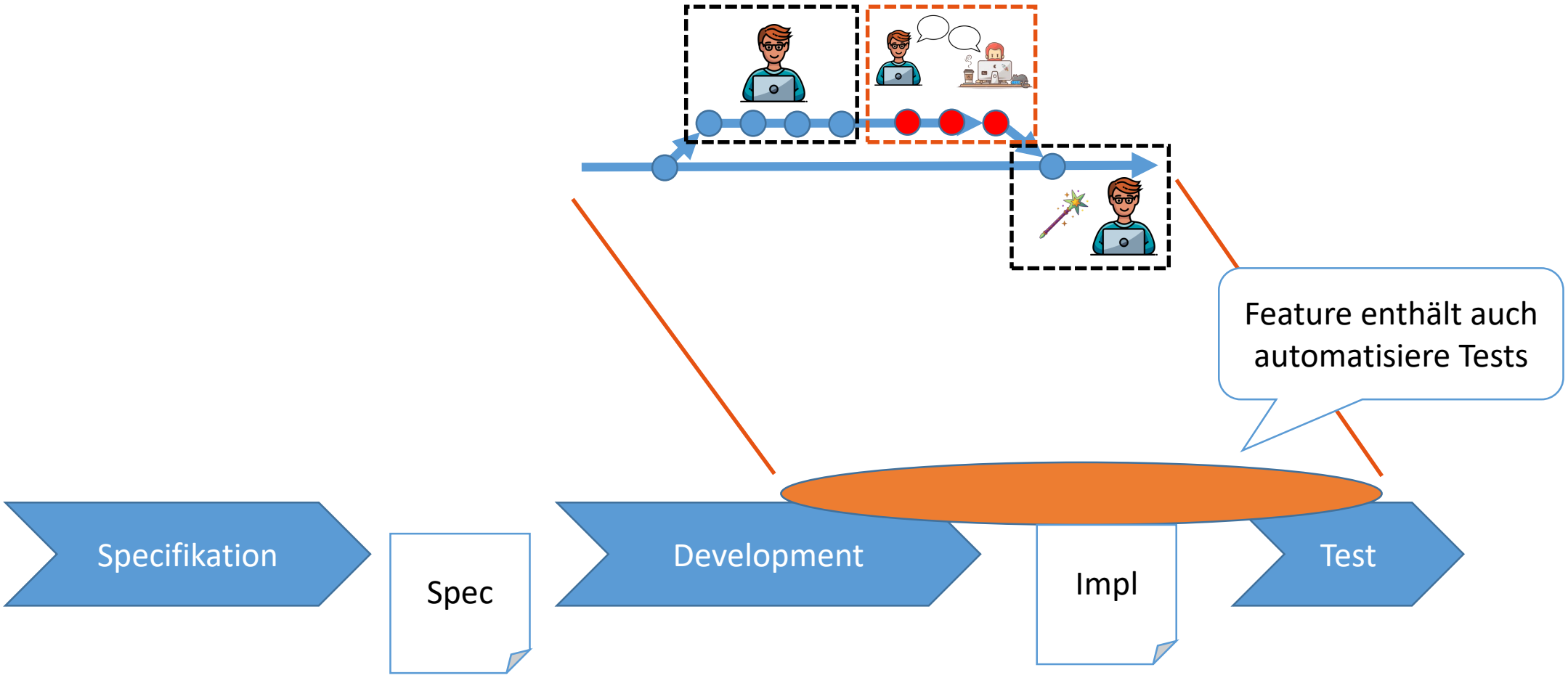
Projekt – Scope



- Fokus auf
  - Style Guides, Patterns, (inline) Doku

- keine Abstimmung mit dem Testern – ‘Testplanung’

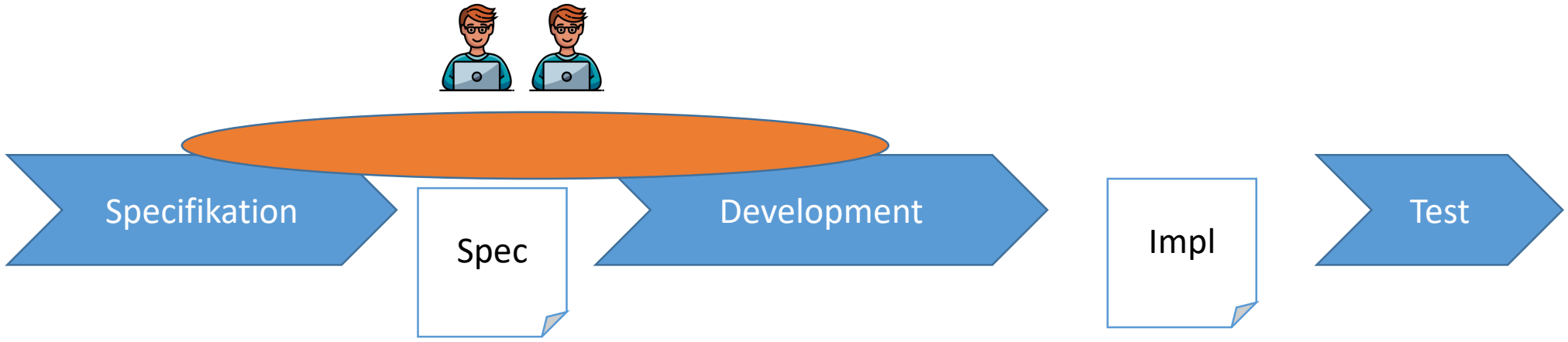
Projekt – Scope



Projekt – Scope

- Refinement
- Design Diskussionen der Entwickler
  VOR der Implementierung

- Review ist zu spät!



- Der kle

Der große Unterschied

Ziel

Teilnehmer

Zwischenmenschliches

zeitlicher Ablauf

Scope

Ziel

Teilnehmer

Zwischenmenschliches

zeitlicher Ablauf

Scope

Qualifikation – Motivation – Kommunikation

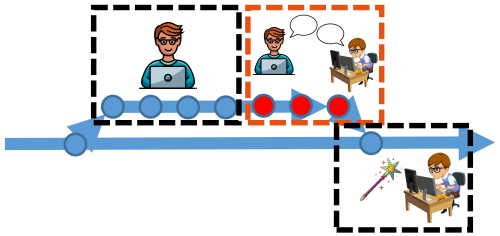
Ziel

Teilnehmer

Zwischenmenschliches

zeitlicher Ablauf

Scope



Open Source Contribution als Lernobjekt

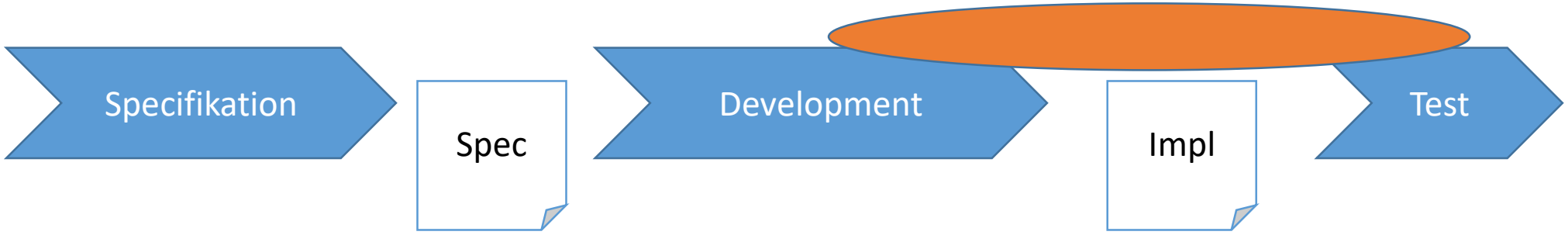
Ziel

Teilnehmer

Zwischenmenschliches

zeitlicher Ablauf

Scope



Der kl

Der große Unterschied

Team – Projekt

Referenzen

Bilder und Cliparts

- https://de.wikipedia.org/wiki/Venus_von_Milo
- https://de.wikipedia.org/wiki/David_(Michelangelo)

- https://clipart-library.com/