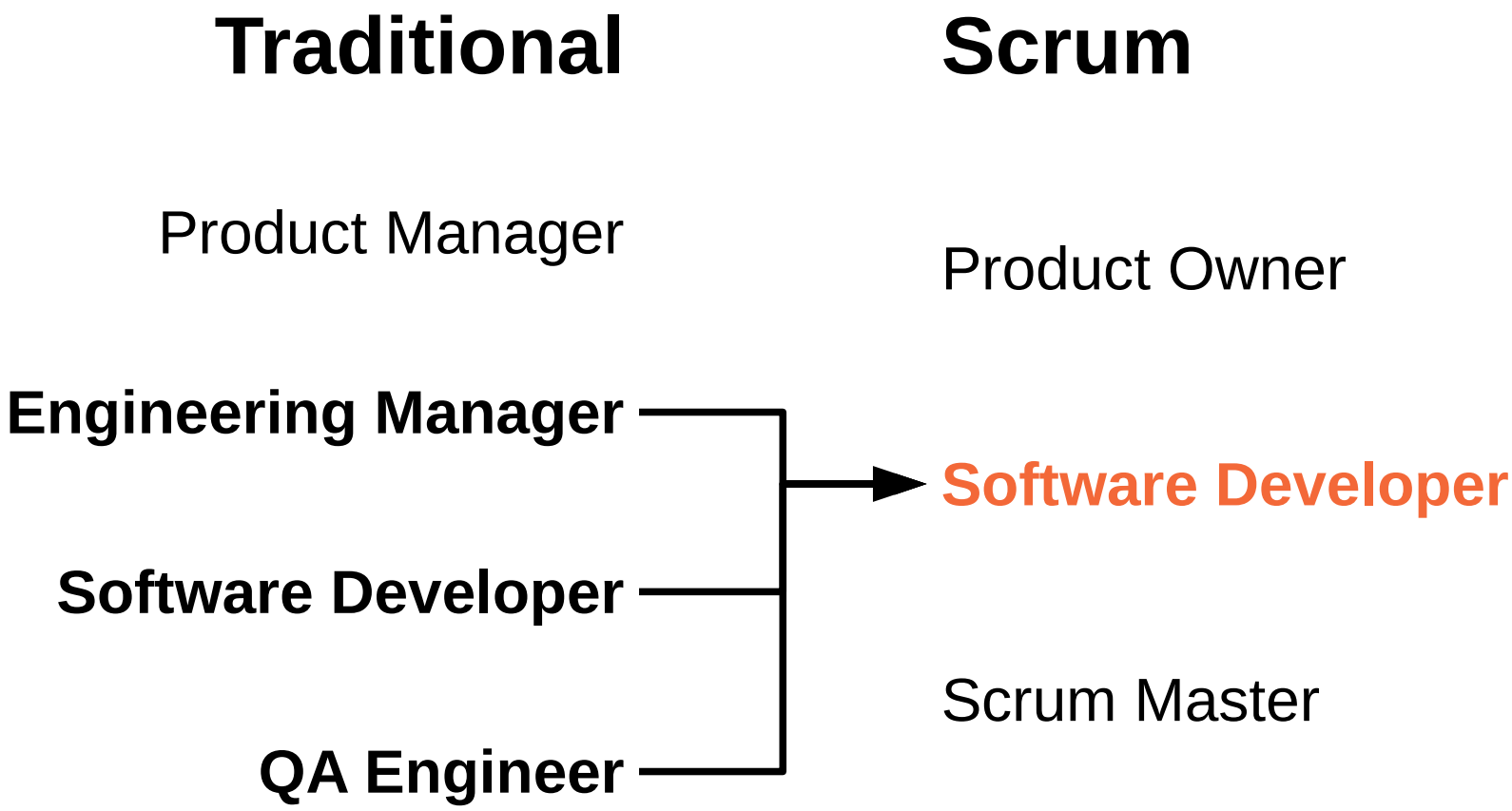
Code Review Practices

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

AMOS F01

Traditional to Scrum Role Mapping (Recap)



Code Review

- Definition and purpose
  - “Code review is systematic examination […] of computer source code.
    - It is intended to find and fix mistakes overlooked in the initial development phase,
      - improving both the overall quality of software
      - and the developers’ skills.
  - Reviews are done in various forms such as
    - pair programming,
    - informal walkthroughs, and
    - formal inspections.” [1]

[1] Adapted from https://en.wikipedia.org/wiki/Code_review [DR]

When to Review Code?

1. In the moment
2. Before commit
3. At another time
4. Before release

Pair Programming

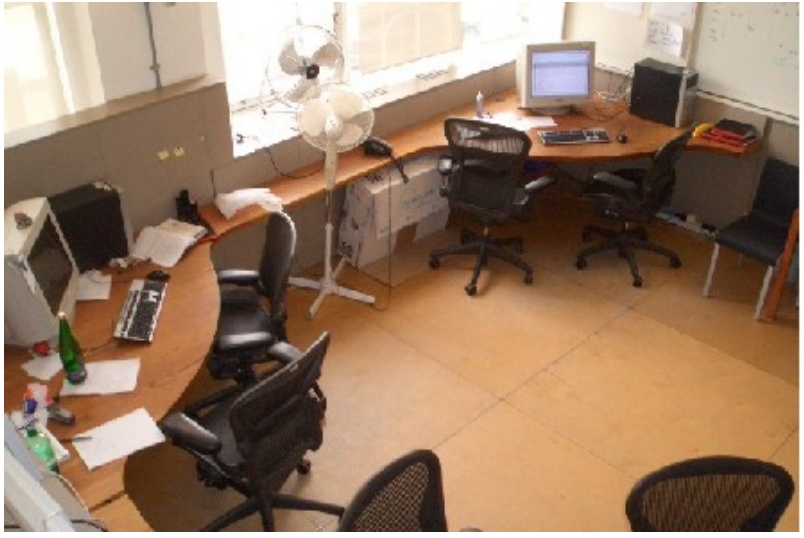
- Definition
  - Is programming carried out by pairs of programmers
  - One programmer implements, and the other programmer reviews
  - Effectiveness is debated; empirical studies show conflicting evidence
- Purpose
  - Quality assurance
  - Collaborative learning
  - Knowledge sharing
- Synonyms
  - Programmer and reviewer
  - Driver and co-driver
  - Pilot and navigator

Agile Code Review Practices

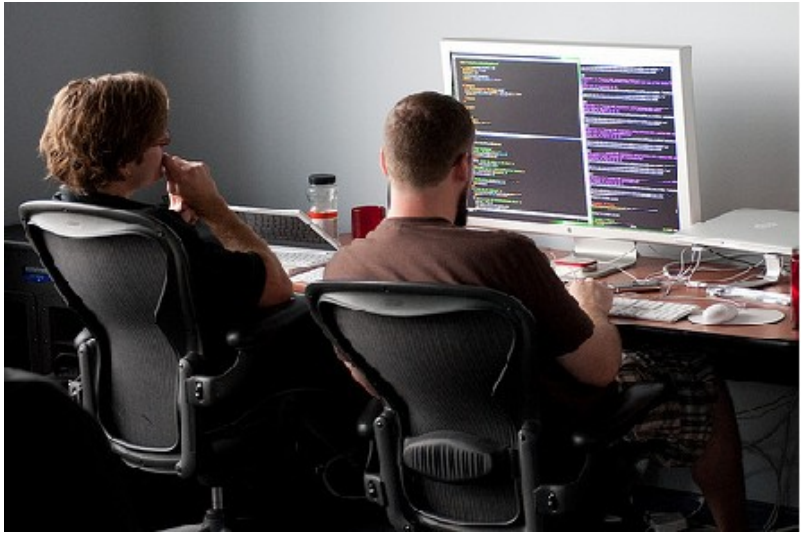
1. **Pair programming**
2. **Pre-commit code review**

Pair Programming (Practices)

- **Process**
  - Find comfortable partner
  - Switch roles often
  - Communicate regularly

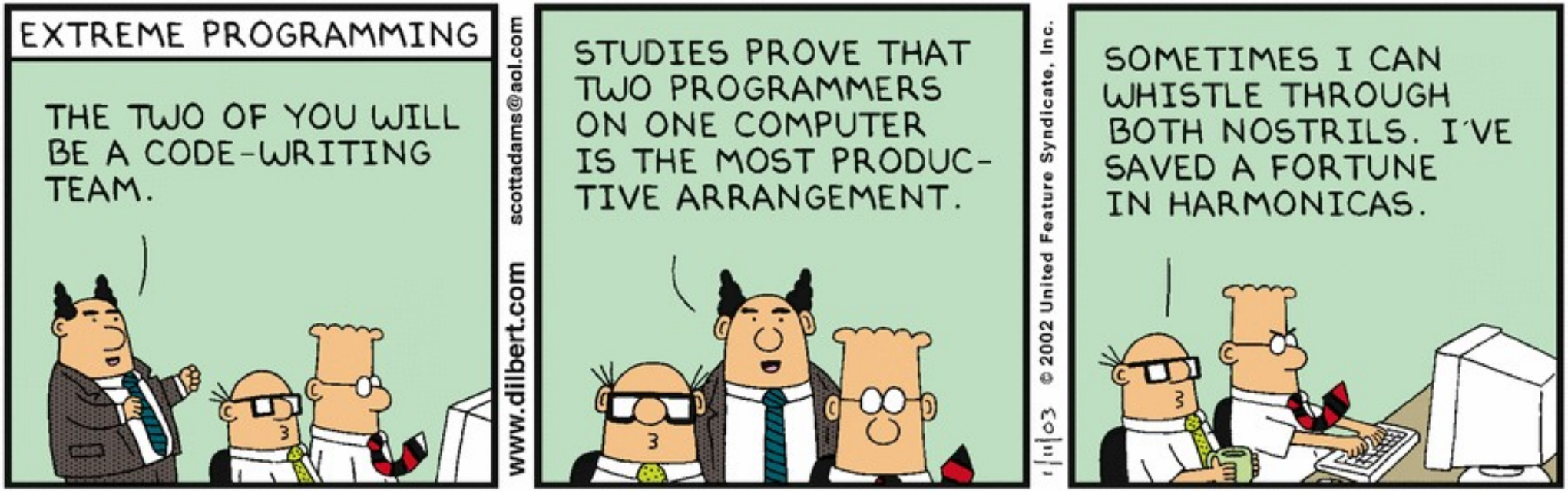


- **Advice**
  - Don’t force it for small stuff
  - Don’t overheat, take a break
  - Switch partners at times



Dilbert on Pair Programming 1 / 2

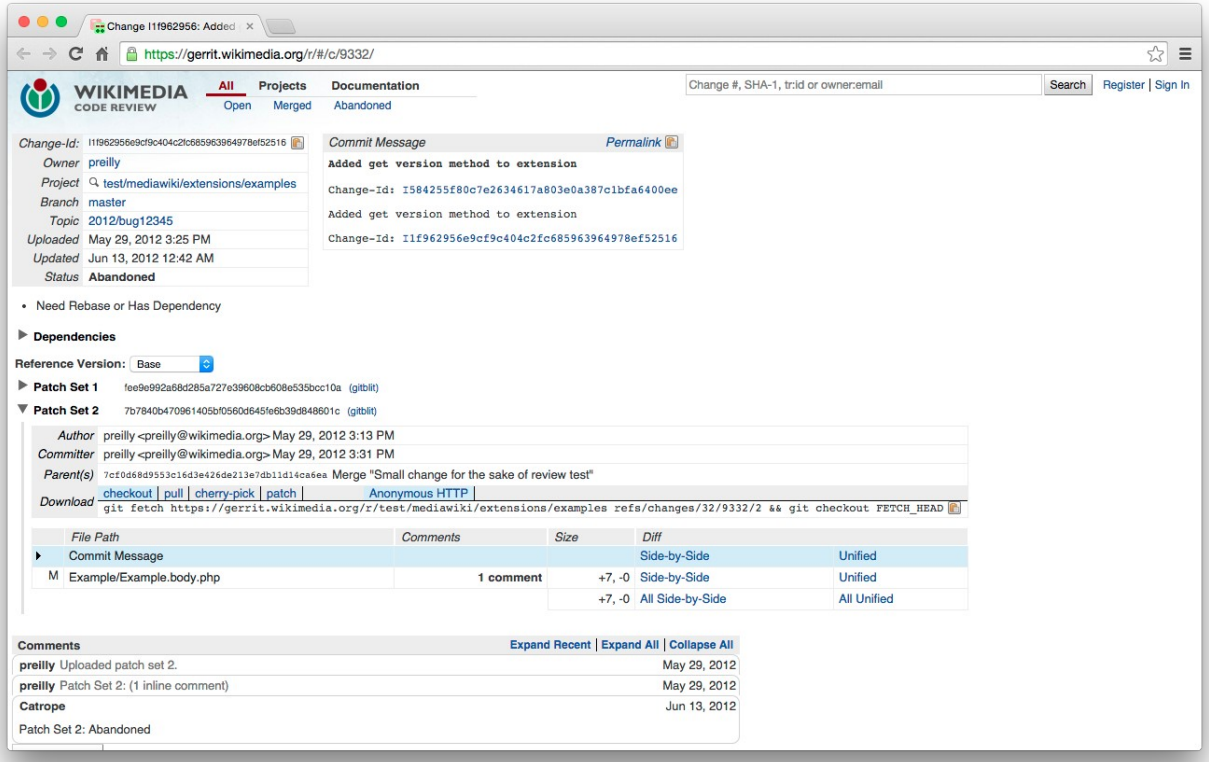
Dilbert on Pair Programming 2 / 2



Pre-Commit Code Review

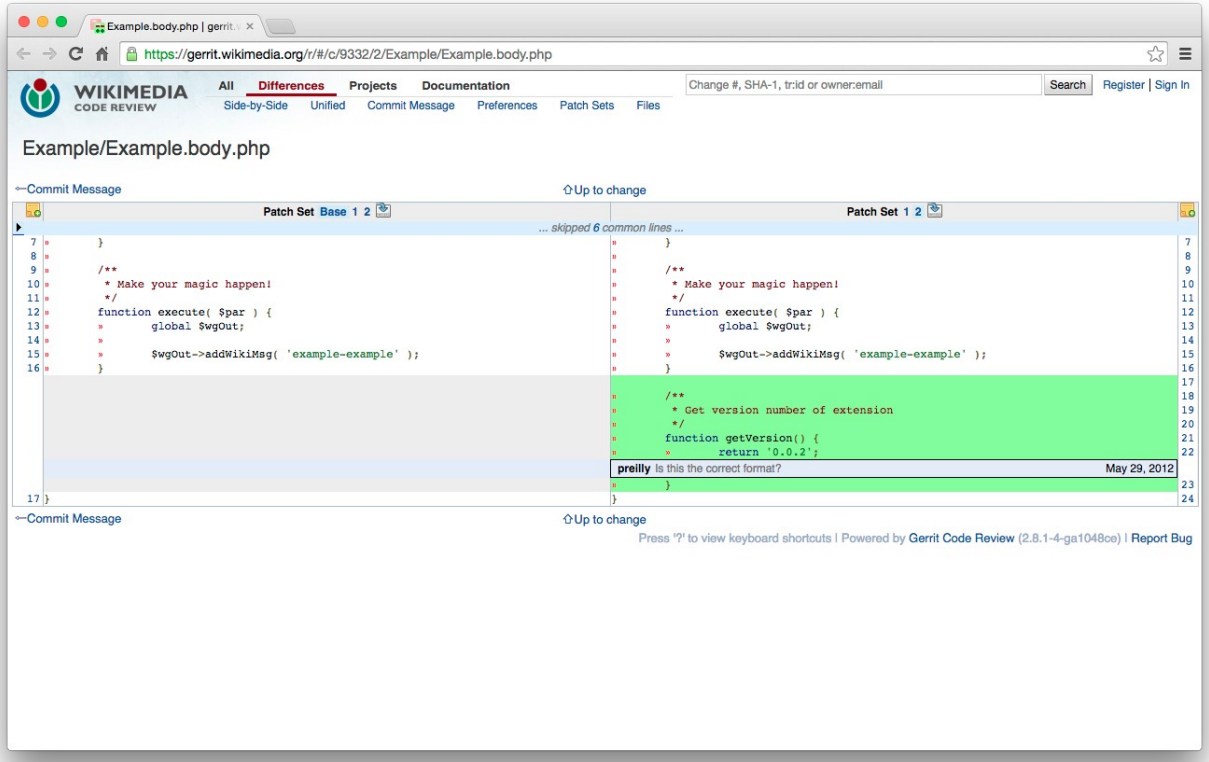
- Code review
  - Is the (peer) review of source code for quality criteria
  - Reviewer has accept or reject responsibility
  - Cf. “Vier-Augen-Prinzip” (in German)
- Pre-commit code review
  - Is the review of source code before it gets committed to a team repository
  - Typically facilitated by a software tool, e.g. Gerrit
  - May lead to back and forth between developers until “LGTM”

Example Code Review with Gerrit 1 / 3

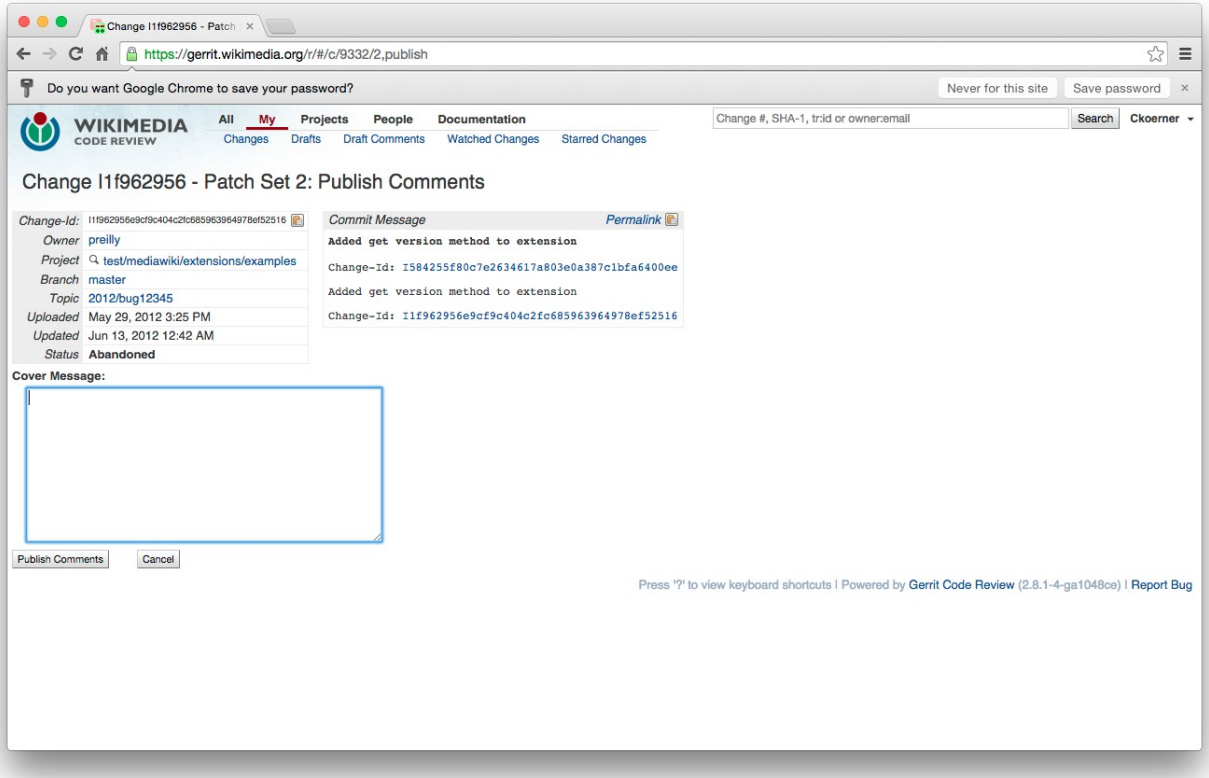


[1] See https://www.mediawiki.org/wiki/Gerrit/Tutorial

Example Code Review with Gerrit 2 / 3



Example Code Review with Gerrit 3 / 3



Benefits of Pre-commit Code Review

- Collaboration
  - Improves knowledge sharing and teamwork
  - Makes it easier to establish topics like security
- Quality assurance
  - Leads to more disciplined developers
  - Prevents (some) errors before they happen
  - Raises overall quality standards
- Feeling of responsibility
  - Specifically, supports collective code ownership
  - Strengthens overall feeling of responsibility

Agile vs. Open Source Code Review

- **Agile methods**
  - Programming guidelines
    - Code reading >> writing
    - Make it easy to get acquainted
  - Collective code ownership
    - Feature-oriented development
    - Typically co-located development
    - Everyone has write access
  - Pair programming
    - Changes are reviewed directly
    - Everyone is a peer
- **Open source**
  - Programming guidelines
    - Code reading >> writing
    - Showing respect for project
  - Individual code ownership
    - Component-oriented development
    - Typically distributed development
    - Strictly regulated write access
  - Patch review
    - Changes are submitted for review
    - Two-class reviewing hierarchy

Review / Summary of Session

- Code review practices
  - Pair programming
  - Pre-commit code review
- Agile vs. open source approach

Thank you! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

Credits and License

- Original version
  - © 2012-2019 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...

Code Review Practices

Prof. Dr. Dirk Riehle

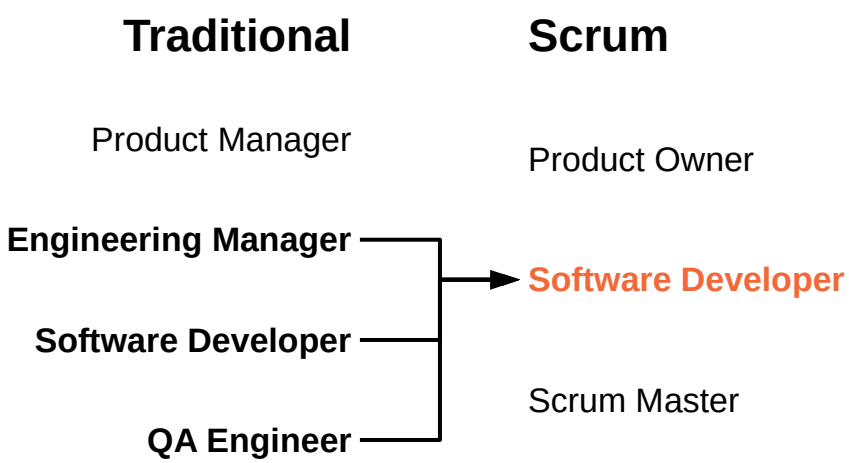
Friedrich-Alexander University Erlangen-Nürnberg

AMOS F01

It is Friedrich-Alexander University Erlangen-Nürnberg – FAU, in short.

Corporate identity wants us to say “Friedrich-Alexander University”.

Traditional to Scrum Role Mapping (Recap)



Code Review

- Definition and purpose
  - “Code review is systematic examination […] of computer source code.
    - It is intended to find and fix mistakes overlooked in the initial development phase,
      - improving both the overall quality of software
      - and the developers’ skills.
  - Reviews are done in various forms such as
    - pair programming,
    - informal walkthroughs, and
    - formal inspections.” [1]

[1] Adapted from https://en.wikipedia.org/wiki/Code_review [DR]

When to Review Code?

1. In the moment
2. Before commit
3. At another time
4. Before release

Pair Programming

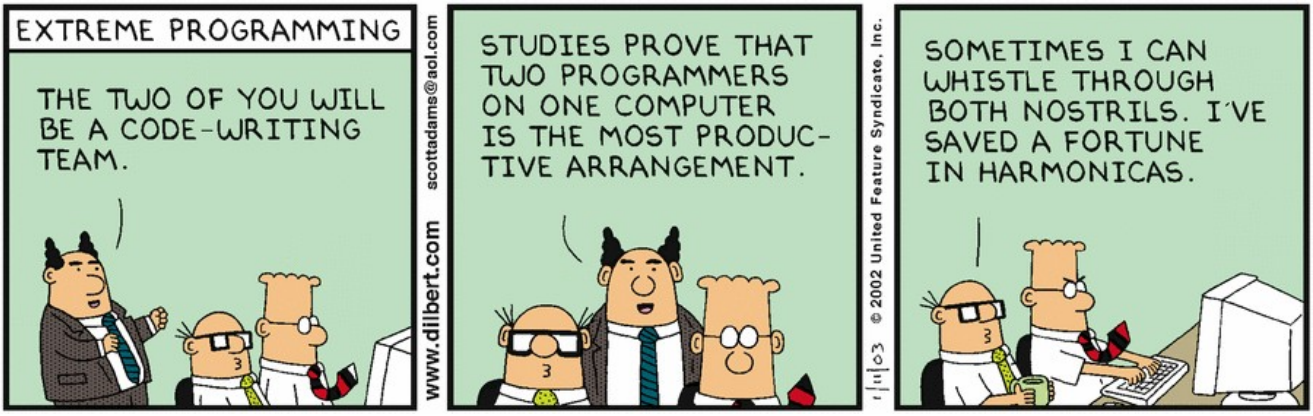
- Definition
  - Is programming carried out by pairs of programmers
  - One programmer implements, and the other programmer reviews
  - Effectiveness is debated; empirical studies show conflicting evidence
- Purpose
  - Quality assurance
  - Collaborative learning
  - Knowledge sharing
- Synonyms
  - Programmer and reviewer
  - Driver and co-driver
  - Pilot and navigator

Agile Code Review Practices

1. Pair programming
2. Pre-commit code review

Dilbert on Pair Programming 1 / 2

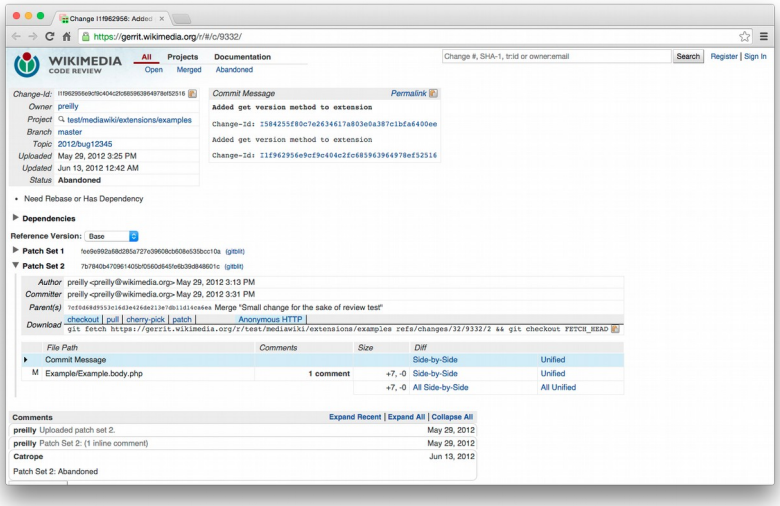
Dilbert on Pair Programming 2 / 2



Pre-Commit Code Review

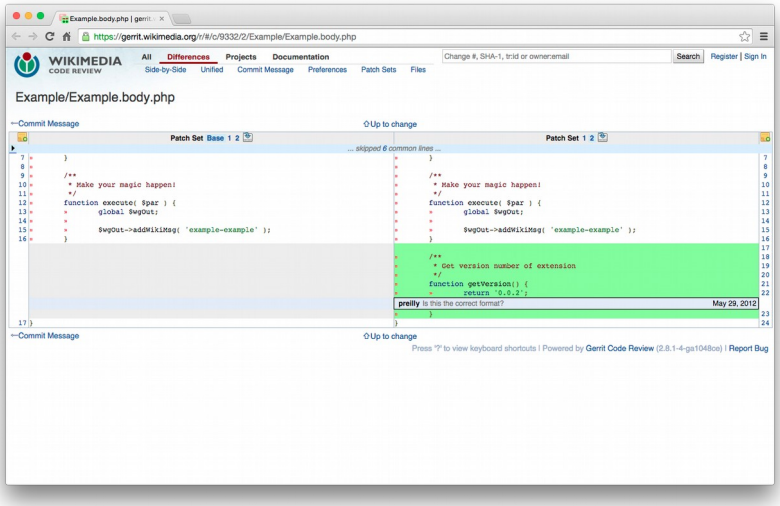
- Code review
  - Is the (peer) review of source code for quality criteria
  - Reviewer has accept or reject responsibility
  - Cf. “Vier-Augen-Prinzip” (in German)
- Pre-commit code review
  - Is the review of source code before it gets committed to a team repository
  - Typically facilitated by a software tool, e.g. Gerrit
  - May lead to back and forth between developers until “LGTM”

Example Code Review with Gerrit 1 / 3

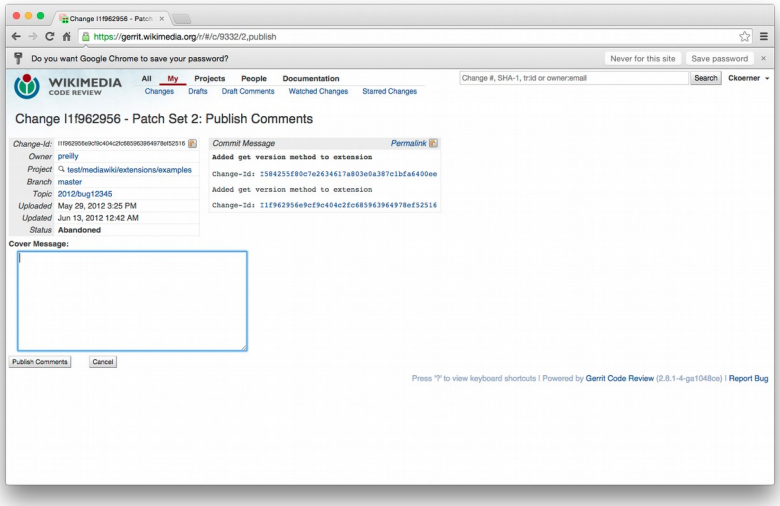


[1] See https://www.mediawiki.org/wiki/Gerrit/Tutorial

Example Code Review with Gerrit 2 / 3



Example Code Review with Gerrit 3 / 3



Benefits of Pre-commit Code Review

- Collaboration
  - Improves knowledge sharing and teamwork
  - Makes it easier to establish topics like security
- Quality assurance
  - Leads to more disciplined developers
  - Prevents (some) errors before they happen
  - Raises overall quality standards
- Feeling of responsibility
  - Specifically, supports collective code ownership
  - Strengthens overall feeling of responsibility

Agile vs. Open Source Code Review

- **Agile methods**
  - Programming guidelines
    - Code reading >> writing
    - Make it easy to get acquainted
  - Collective code ownership
    - Feature-oriented development
    - Typically co-located development
    - Everyone has write access
  - Pair programming
    - Changes are reviewed directly
    - Everyone is a peer
- **Open source**
  - Programming guidelines
    - Code reading >> writing
    - Showing respect for project
  - Individual code ownership
    - Component-oriented development
    - Typically distributed development
    - Strictly regulated write access
  - Patch review
    - Changes are submitted for review
    - Two-class reviewing hierarchy

Review / Summary of Session

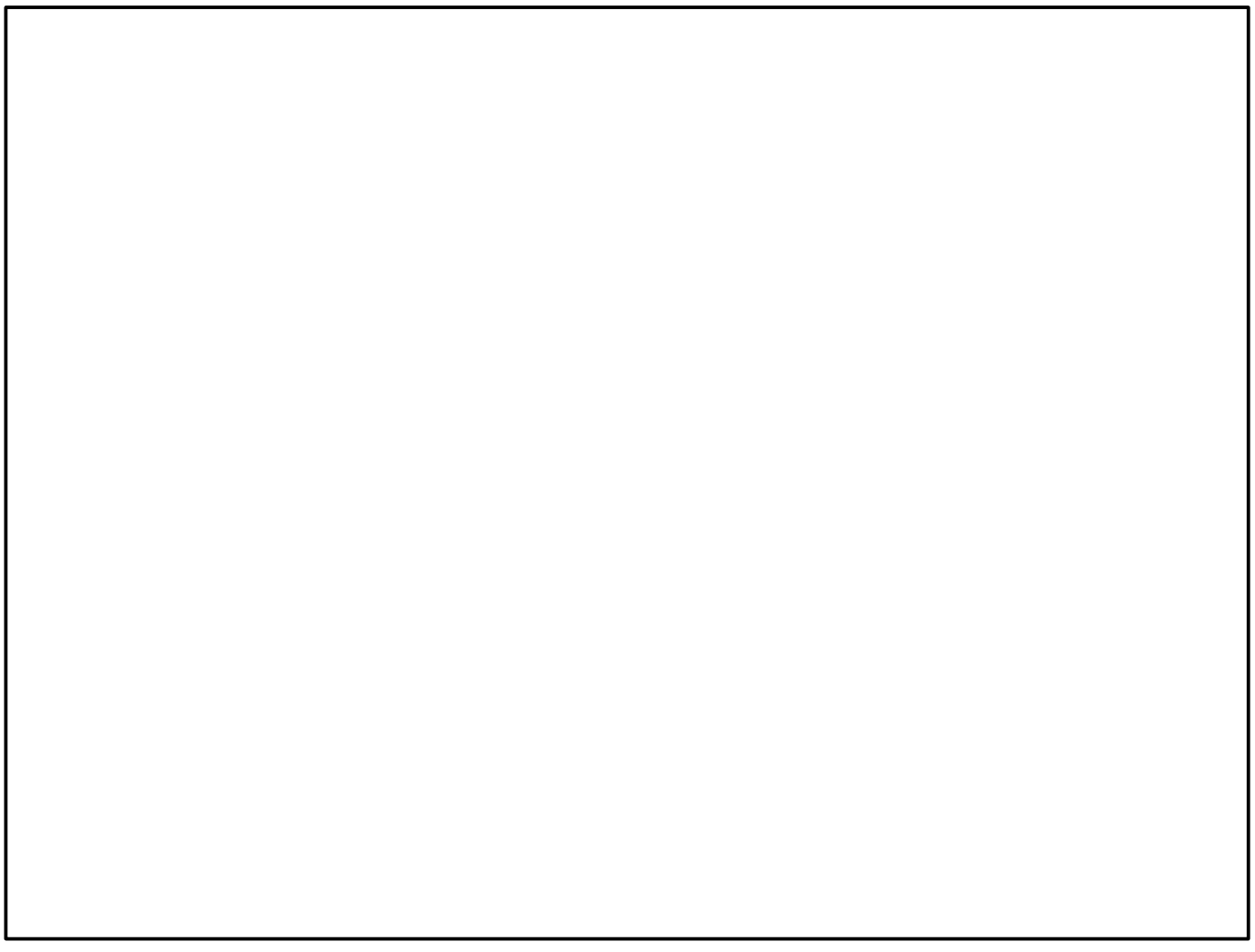
- Code review practices
  - Pair programming
  - Pre-commit code review
- Agile vs. open source approach

Thank you! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle

DR



Credits and License

- Original version
  - © 2012-2019 Dirk Riehle, some rights reserved
  - Licensed under Creative Commons Attribution 4.0 International License
- Contributions
  - ...