

# Code Review Practices

**Prof. Dr. Dirk Riehle**

**Friedrich-Alexander University Erlangen-Nürnberg**

**AMOS F01**

Licensed under CC BY 4.0 International

# Traditional to Scrum Role Mapping (Recap)

## Traditional

## Scrum

Product Manager

Product Owner

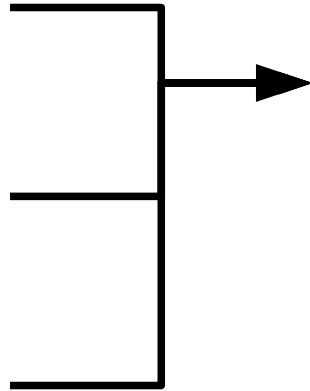
Engineering Manager

Software Developer

QA Engineer

**Software Developer**

Scrum Master



- 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](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

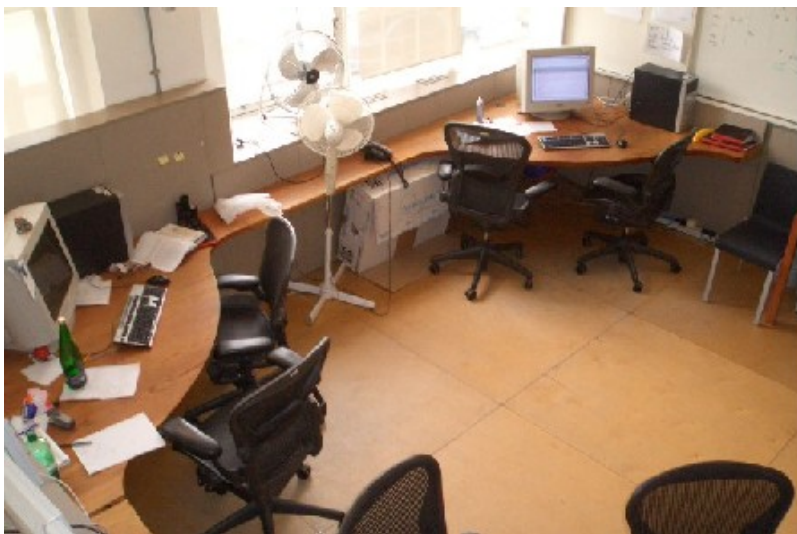
- Definition
  - Is programming carried out by pairs of programmers
  - One programmer implements, and the other programmer reviews
  - Effectiveness is debated; empirical studies show conflicting evidence
- Purpose
  - Quality assurance
  - Collaborative learning
  - Knowledge sharing
- Synonyms
  -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

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

# Pair Programming (Practices)

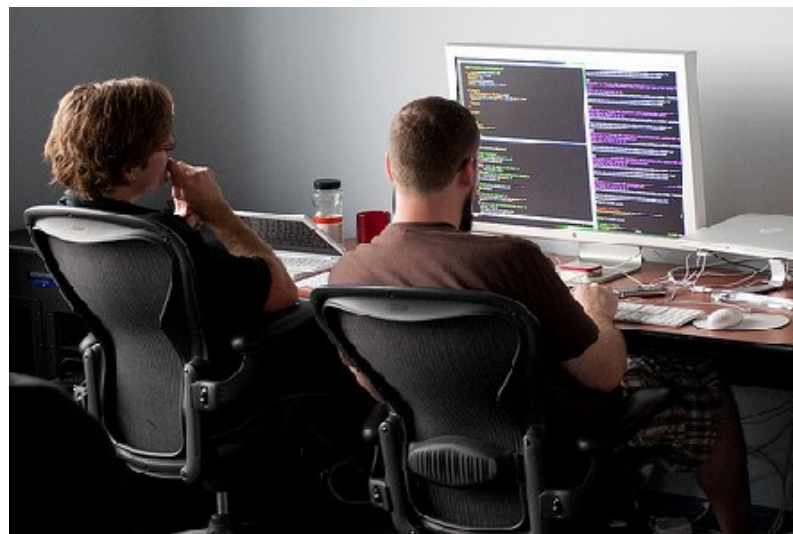
- **Process**

- Find comfortable partner
- Switch roles often
- Communicate regularly

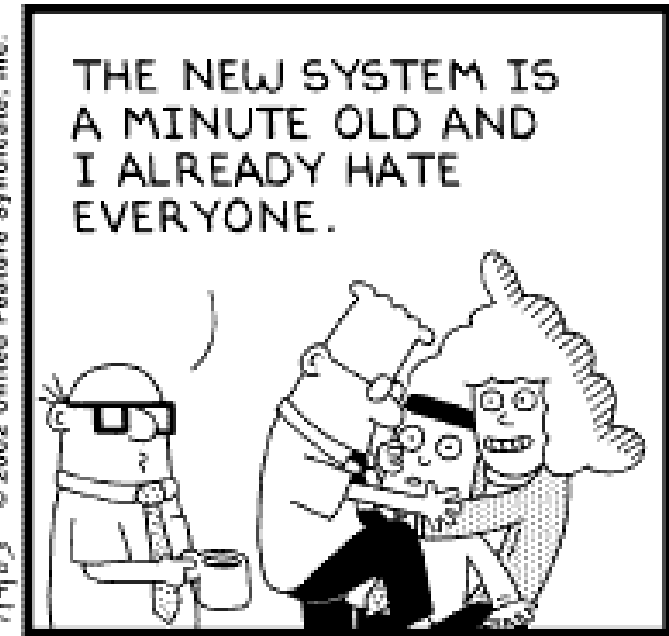


- **Advice**

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

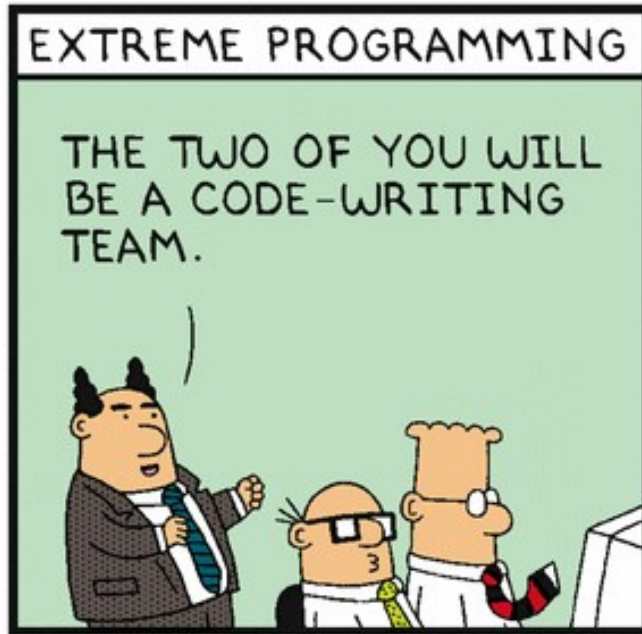


# Dilbert on Pair Programming 1 / 2



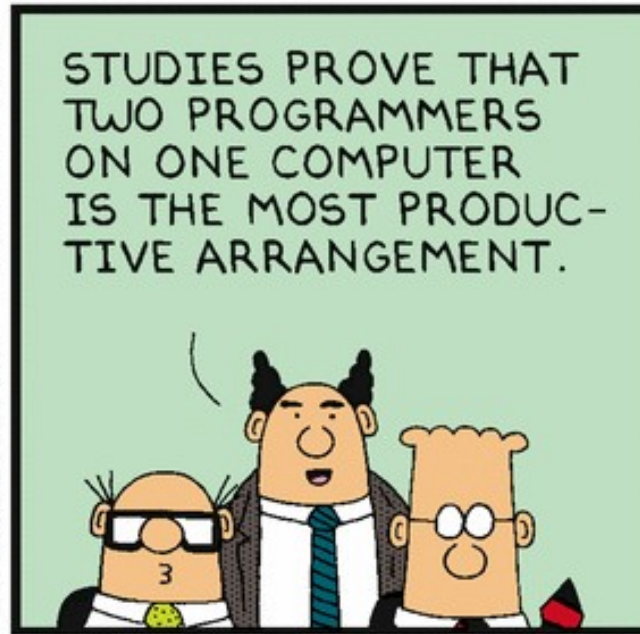


# Dilbert on Pair Programming 2 / 2



scottadams@aol.com

www.dilbert.com



© 2002 United Feature Syndicate, Inc.



# 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

The screenshot displays the Gerrit Code Review web interface. At the top, the browser address bar shows the URL <https://gerrit.wikimedia.org/r/#/c/9332/>. The page header includes the Wikimedia Code Review logo and navigation tabs for 'All', 'Projects', and 'Documentation'. A search bar and links for 'Register' and 'Sign In' are also present.

The main content area shows details for a specific change set:

- Change-Id:** I1f962956e9cf9c404c2fc685963964978ef52516
- Owner:** preilly
- Project:** test/mediawiki/extensions/examples
- Branch:** master
- Topic:** 2012/bug12345
- Uploaded:** May 29, 2012 3:25 PM
- Updated:** Jun 13, 2012 12:42 AM
- Status:** Abandoned

The **Commit Message** section shows the commit details, including the SHA-1 hash and the commit message: "Added get version method to extension".

Below the commit message, there are sections for **Dependencies** and **Reference Version**. The **Reference Version** is set to 'Base'.

The **Patch Set** section shows two patch sets. Patch Set 2 is selected, showing the author, committer, parent(s), and download links. The **Download** section shows the git fetch command.

The **File Path** table shows the changes made in the patch set:

File Path	Comments	Size	Diff
Commit Message			Side-by-Side
M Example/Example.body.php	1 comment	+7, -0	Side-by-Side
		+7, -0	All Side-by-Side

The **Comments** section shows a list of comments on the patch set, including the upload date and the user who made the comment.

# Example Code Review with Gerrit 2 / 3

Example.body.php | Gerrit

https://gerrit.wikimedia.org/r/#/c/9332/2/Example/Example.body.php

WIKIMEDIA CODE REVIEW

All Differences Projects Documentation

Side-by-Side Unified Commit Message Preferences Patch Sets Files

Change #, SHA-1, trid or owner:email Search Register Sign In

Example/Example.body.php

←Commit Message Up to change

Patch Set Base 1 2 Patch Set 1 2

```
7 8 9 10 11 12 13 14 15 16 17 }
8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
... skipped 6 common lines ...
/**
 * Make your magic happen!
 */
function execute( $par ) {
    global $wgOut;

    $wgOut->addWikiMsg( 'example-example' );
}

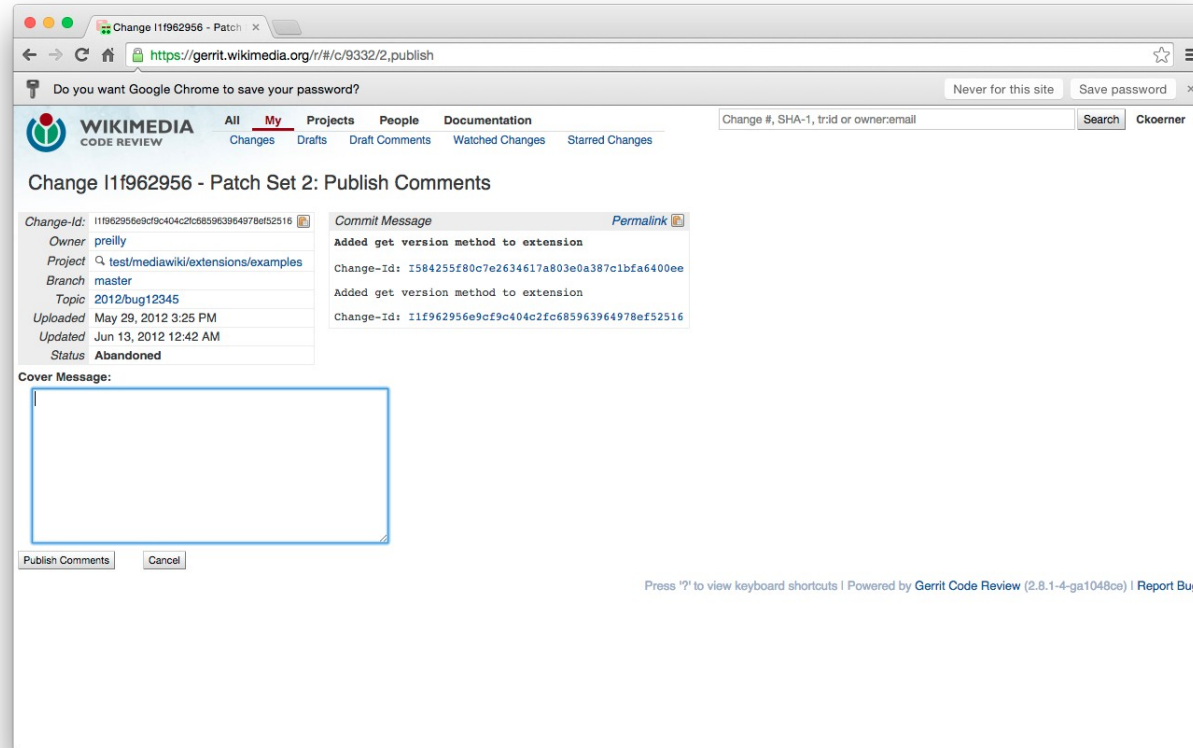
/**
 * Get version number of extension
 */
function getVersion() {
    return '0.0.2';
}

prelilly Is this the correct format? May 29, 2012
```

←Commit Message Up to change

Press '?' to view keyboard shortcuts | Powered by Gerrit Code Review (2.8.1-4-ga1048ce) | Report Bug

# 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](mailto:dirk.riehle@fau.de) – <http://osr.cs.fau.de>**

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

# Credits and License

- Original version
  - © 2012-2020 Dirk Riehle, some rights reserved
  - Licensed under [Creative Commons Attribution 4.0 International License](#)
- Contributions
  - ...