

Effective Code Review for Agile Java Developers



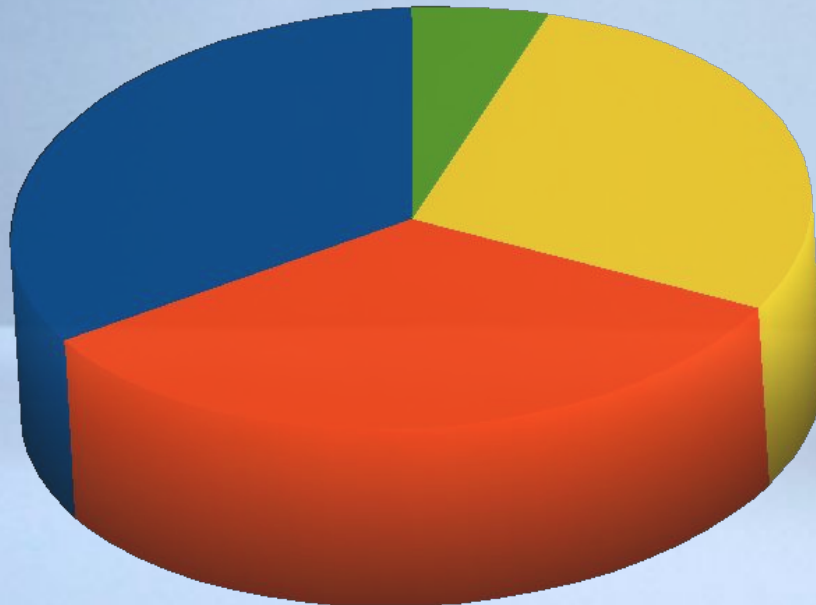
Wojciech Seliga & Sławomir Ginter



An Evil Plan

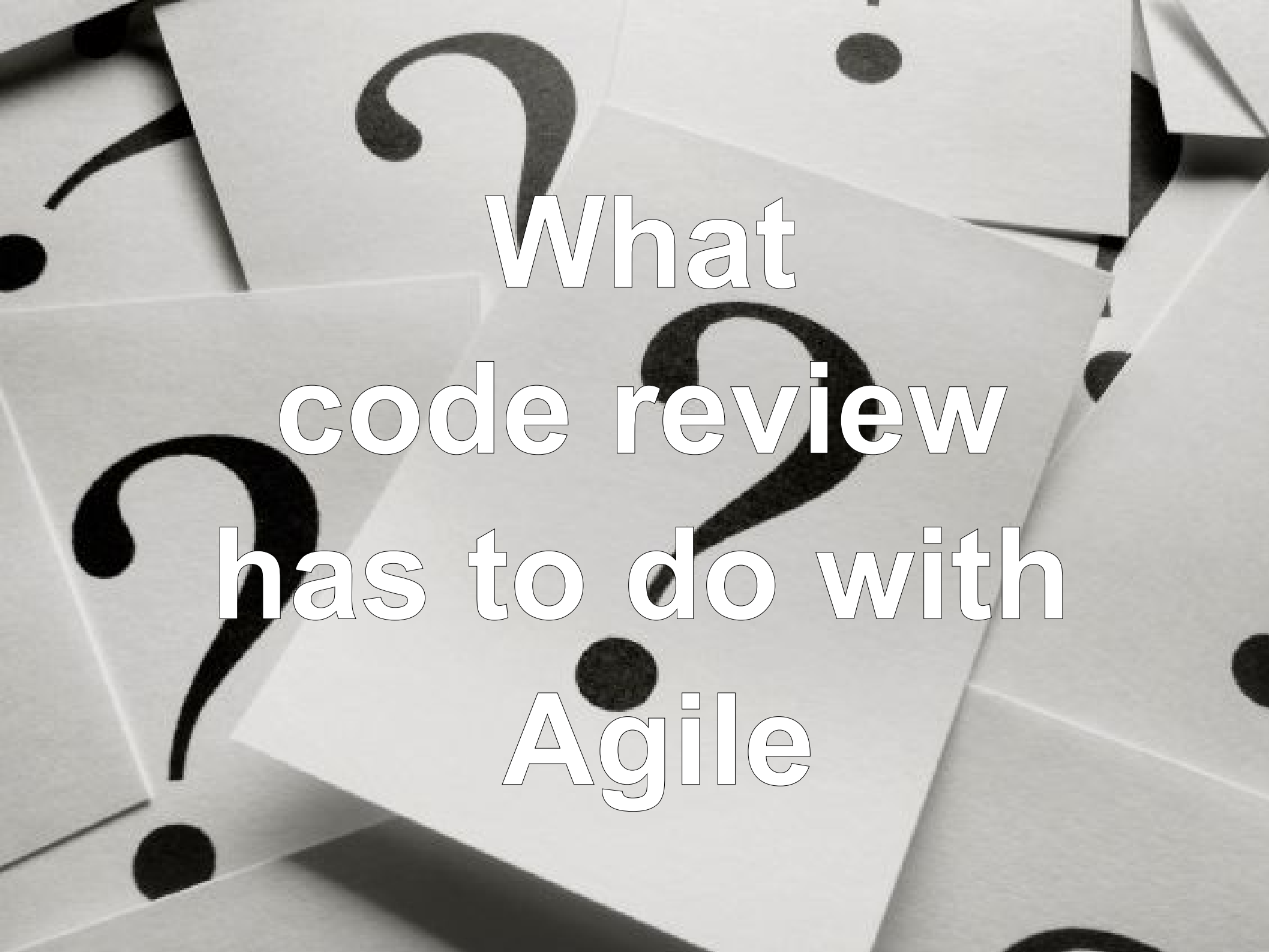


■ Talking	■ Demoing
■ Q&A	■ Fail Buffer



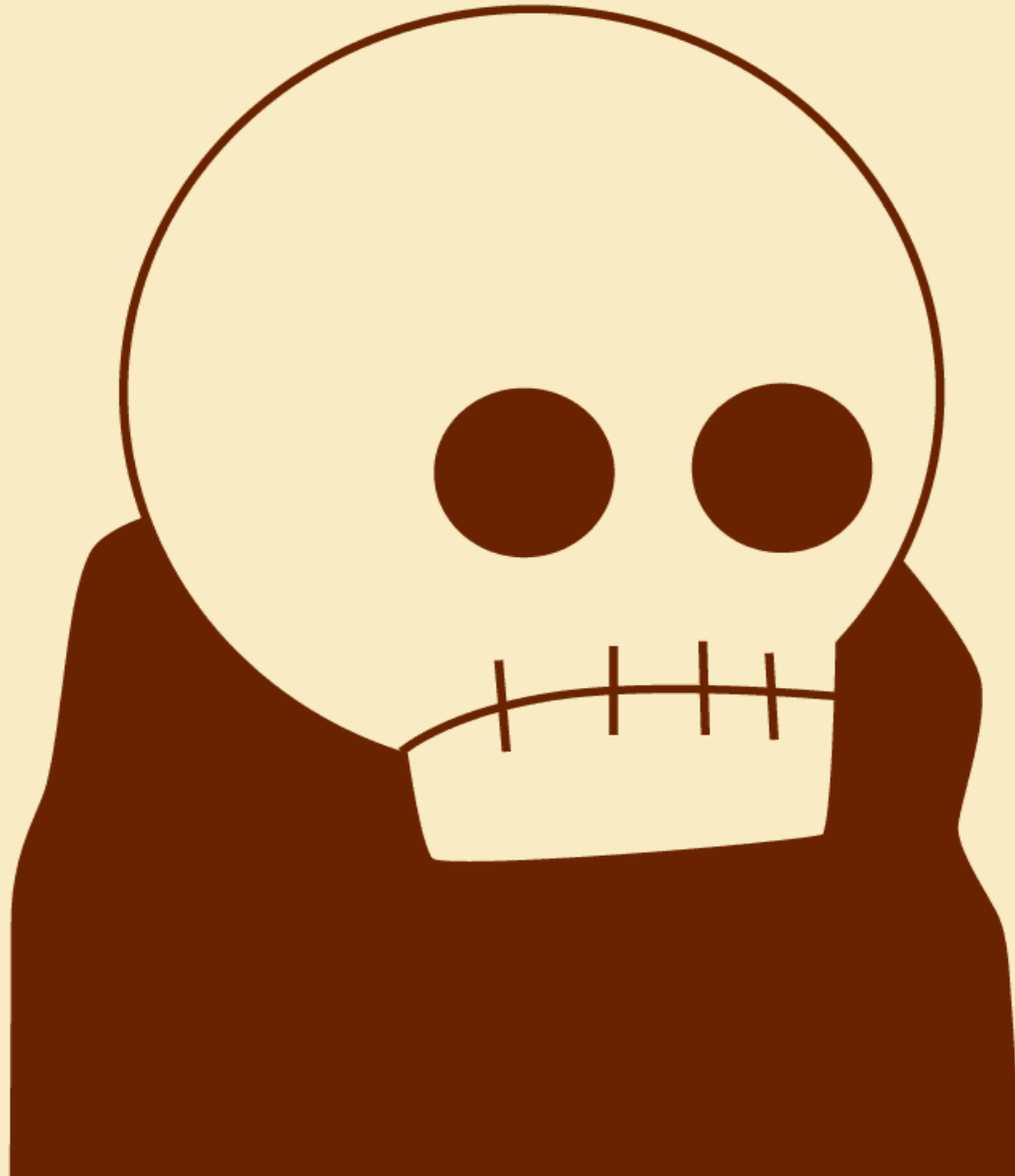
Disclaimer

**Yes,
we do have interest in
popularizing code reviews**

The background consists of several overlapping, light-colored rectangular papers. Each paper has a large, bold, black question mark printed on it. The papers are slightly offset from each other, creating a layered effect. Some papers have circular punch holes visible along their edges.

What
code review
has to do with
Agile

WHY BOTHER



Picture courtesy of [Cat and Girl](#)
CC A-NC-SA 2.5

Introducing people to the code



style/conventions

design

reusable components

APIs

do's and don'ts

Mentoring junior developers



Non-intrusive

Asynchronous

Less frustration / interruption
for senior devs

Sharing good engineering practices



Spotting issues earlier



design problems

inconsistencies

concurrency issues

redundancies

bugs



Facilitating distributed teams



Photo Courtesy of U.S. Army

A close-up photograph of a giant panda sitting on the ground, holding a bamboo stalk in its mouth and using its paws to peel it. The panda's black and white fur is clearly visible. The background is a natural, outdoor setting with dirt and some greenery.

**Why then is code
review so rarely
used?**

Preparation is difficult



selecting the code
organizing reviewers
booking conference
room
scheduling
printing
...



A satellite image of a hurricane, showing a large, swirling cloud system with a distinct eye in the center. The image is in grayscale, with the clouds appearing white and the ocean surface appearing dark. The hurricane is positioned in the lower-left quadrant of the frame.

Yet another disruption



synchronized meetings

de-focusing

Time consuming



a lot of code to read every time

idle time during meetings

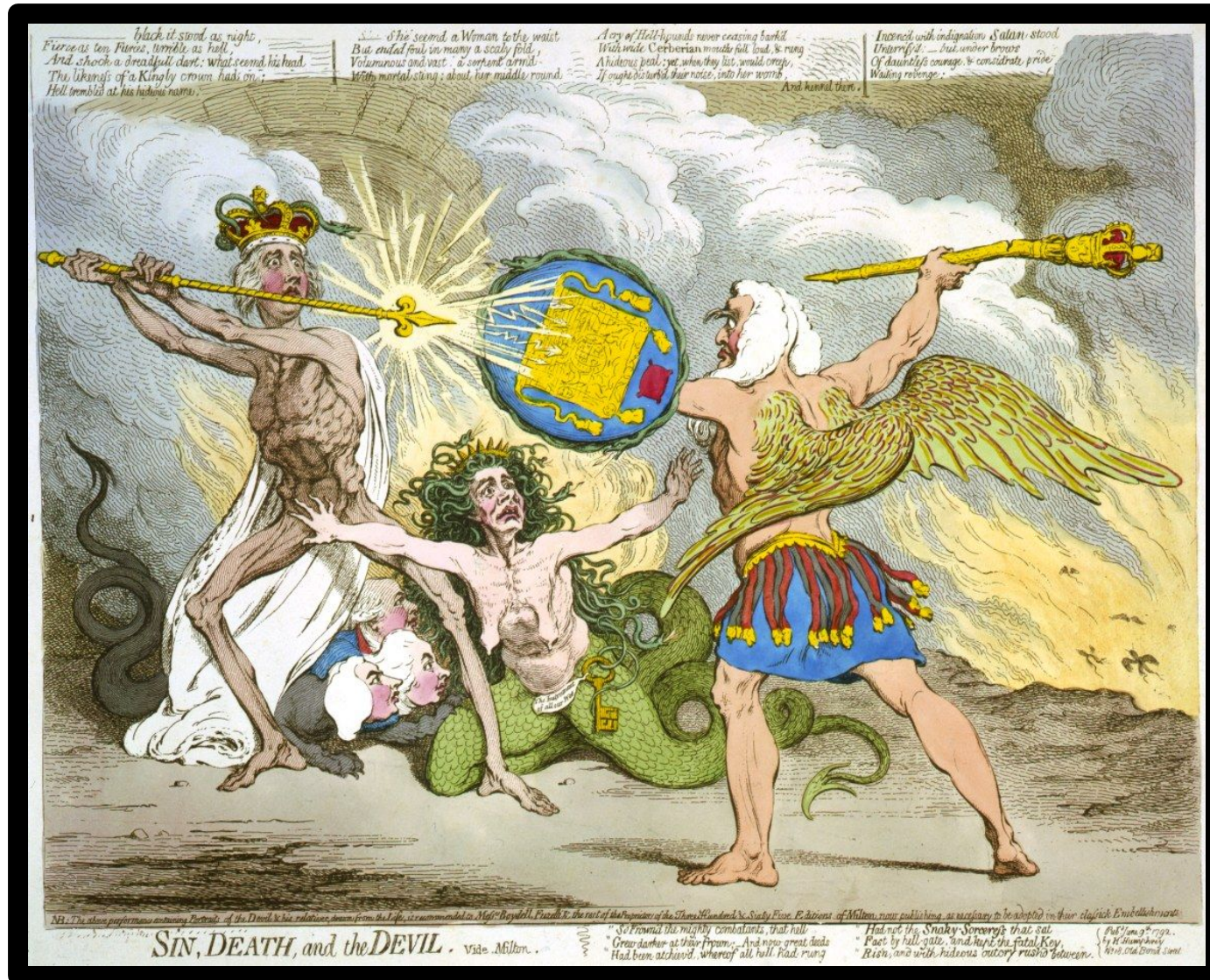
wasted time on simple things:

warnings

coding style

test coverage

Risk of animosities



No concrete measureable results



Successful code review

Lightweight - simple & flexible process

Asynchronous

Continuous

Efficient tool support

Diff-oriented whenever applicable

Transparent and persistent

Time for a longer demo



Agile code review misconceptions

a fanatic bug-hunt

false confidence about
no bugs left

tracking results of every
single comment

expecting hard metrics



Picture courtesy of [Juria Yoshikawa](#) / CC BY-SA 2.0

The greatest initial Enemies

Rigid Process

Metrics

Micro-management



But teams evolve...

Some rules of agile code review



everyone can join review
& comment

everyone can modify the
scope of the review

everyone can invite other
people

everything is public
across the company

it's about learning, it's not
about blaming

Unexpected advantages

Facilitation of distributed teams

Collaboration on low level design

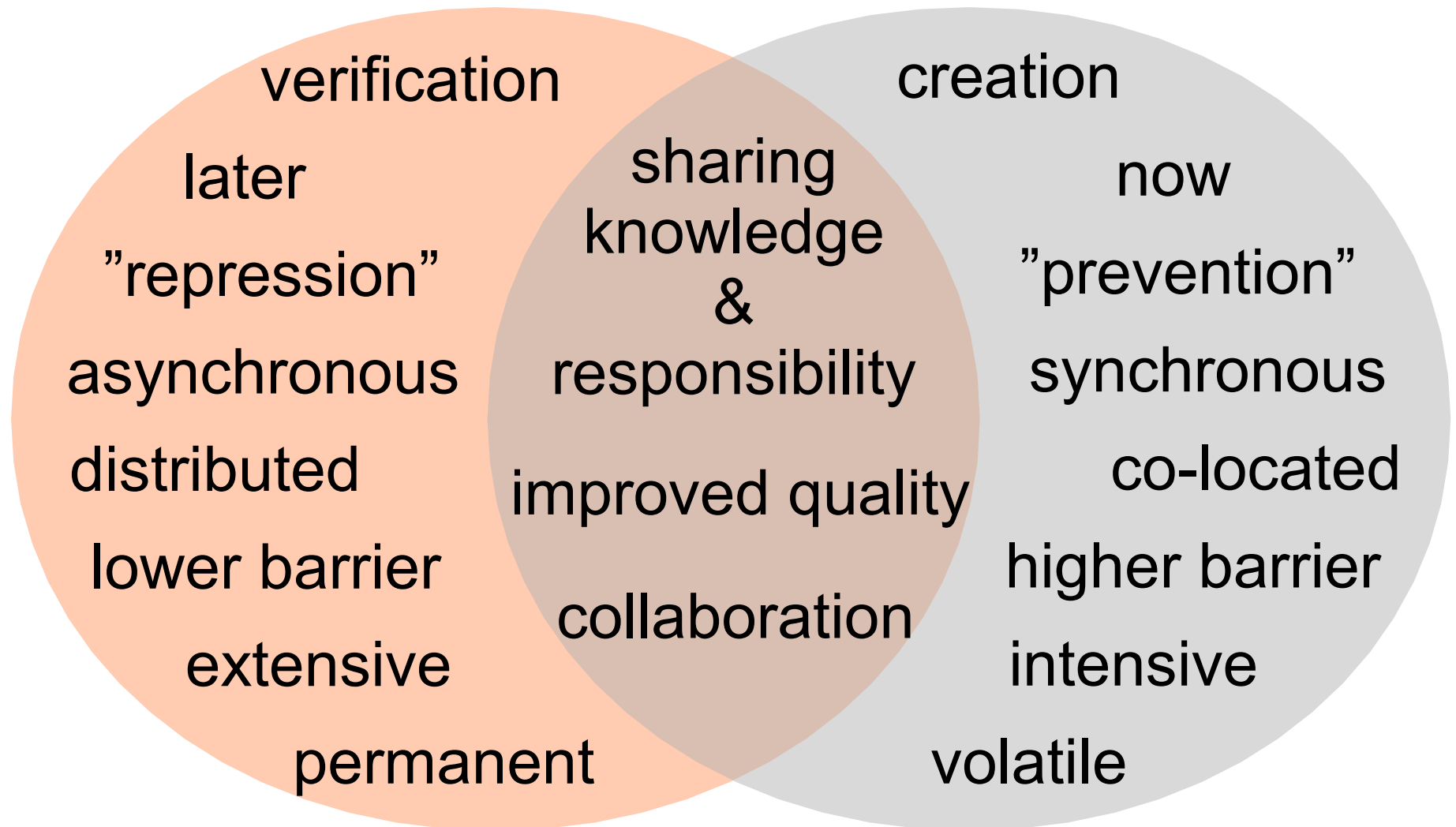
Easier to introduce than pair-programming

Time-zone difference may help you

Knowledge Base



Code Review vs. Pair Programming



The answer

Individuals and interactions

Over processes and tools

Working software

Over comprehensive documentation

Customer collaboration

Over contract negotiation

Responding to change

Over following a plan

XP: Collective Code Ownership

About us



wojciech.seliga@spartez.com

slawomir.ginter@spartez.com

Word cloud for Wojciech Seliga: agile, developer, Java, Scrum, project manager, IDE, Connectors, Atlassian, common sense, computer graphics, mentoring, GIS, XP, aviation, technical leader, mentor, process, C++, OOP.

Word cloud for Slawomir Ginter: Atlassian, developer, Java, ScrumMaster, IntelliJ, SparteZ, agile, integration, XPC/C++, Scrum, middleware, assembler, project manager, mentor, technical leader, guru, e-learning, Clover, Intel.

Q&A



Thank you