



Home News & Analysis EE Life Design Products Education & Training Events Sign In Join

Design

DesignLines

Audio

Automotive

CommsDesign

EDA

Embedded.com

Embedded Internet

Graphical System

Design

Industrial Control

MCU

Medical

Memory

Microwave & RF

Military & Aerospace

Planet Analog

Power Management

Programmable Logic

Signal Processing

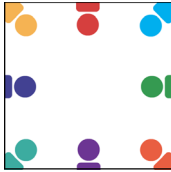
Smart Energy

Design Tools

Reference Designs

Evaluation Kit

Source Code



EE Times Home > EE Life

Guest Editor

Modernizing the peer code review process

Alen Zukich, Klocwork

7/6/2010 3:50 PM EDT

Code reviews play a critical role in the software development process and their roots can be traced to the birth of software engineering. But while the practice of software development has changed greatly since its origins, the code review processes employed by most organizations have not.

Forrester Consulting recently surveyed 159 U.S.-based software development professionals to garner details on the state of today's peer code review process. This [study](#) found that, while the code review process is held in high regard and is often a mandated development milestone, the best practices and processes being used to conduct code reviews have not kept pace with the rapid evolution of the software development process.

The study uncovered some troubling process gaps in today's organizations. While the software development industry has seen the growth of outsourcing and geographically distributed teams, their code review processes do not take these practices into consideration when conducting code reviews. The Forrester report noted the following problems:

- Most code reviews are done in person using local resources rather than resources with the best skill or experience.
- Technology is used inconsistently. Despite the availability of tools that can help automate and simplify the code review process, respondents indicated a very low rate of tool usage, with 18% using no tools at all.
- Despite their 'big picture' insight, QA and architects are not always involved in code reviews. More than half of the software development professionals surveyed indicated limited or irregular participation of architects and QA staff.

Although many organizations set out clear mandates and policies for code reviews, the processes used continue to be informal. In particular:

- Code is reviewed on an ad hoc basis. For example, 30% of the IT professionals surveyed don't have a formal process for determining when a code review should take place.
- Code review members are selected informally. In fact, 57% of IT professionals reported that there was no formal process in place at their organization for selecting reviewers.

When asked to describe their top review challenges, IT professionals stated that they don't have enough time to prepare for code reviews, and they can't get the right people interested and motivated to perform the reviews. Although the professionals surveyed identified a number of challenges, all respondents believed that code reviews bring real value to the software development lifecycle. Specifically, respondents listed finding bugs earlier, sharing best practices, continuous code refactoring and overall time savings as the key benefits of code review.

While developers clearly recognize the value of code reviews, this survey highlights that most organizations are not conducting reviews as efficiently or effectively as they could. And, by not taking advantage of modern tools and workflow practices, organizations are limiting the impact and value of the code reviews currently being performed.

With the complexity of today's software and the realities of outsourcing and geographically dispersed teams, it's time to re-evaluate the current approach to code review. Software development organizations can modernize their code review process with some key strategies. By grounding code reviews to stable, predictable development milestones that already exist, code reviews become an organic, natural part of development rather than a disruptive process.

Organizations can begin by tying code reviews to Agile stories (i.e. requirements) and code check-in. In short, code review should be part of every feature story. Before implementing a story, decide whether the feature requires a code review.

Code review also should flow naturally from code check-in, since that is when a developer has determined that code is "done." As code is ready to be checked in, a code review can be launched and the appropriate people can be notified that there are changes in a component or module ready for review.

Software developers are already consumers of social media. Forrester Consulting found that more than 80% of developers surveyed use social media in their

Navigate to related information

[Technical Paper: Best Practices for Peer Code Review](#)

[Technical Paper: Peer Code Review: An Agile Process](#)

[Technical Paper: Improve Quality and Morale: Tips for Managing the Social Effects of Code Review](#)

[Technical Paper: The Value and Importance of Code Reviews](#)

[News: ESC - Embed-X incorporates first Agile dev framework for critical software processes](#)

How do you implement wireless technology?

Most Popular

- 1 [USB Explained: An Introduction to USB and Its Future](#)
- 2 [Digital Signal Processing: A Practical Guide \(Part 1\)](#)
- 3 [Fundamentals of Wireless](#)
- 4 [The Inefficiency of C++, Fact or Fiction?](#)
- 5 [Digital Signal Processing: A Practical Guide \(Part 2\)](#)
- 6 [Digital Signal Processing: A Practical Guide \(Part 3\)](#)
- 7 [Comment: Andy Grove, startups and job creation](#)
- 8 [Digital Signal Processing: A Practical Guide \(Part 4\)](#)
- 9 [BLDC motor for Electronic Bike Application](#)
- 10 [Digital Signal Processing: A Practical Guide \(Part 5\)](#)

Product Parts Search

Enter part number or keyword

SEARCH

Get Involved

[Start a Forum](#)

[Subscribe to a Newsletter](#)

Are you connected?

There's a new place for designers of embedded internet

[click here to get connected](#)



personal lives and 50% use social media to get answers to technical questions. This presents an opportunity to use some social media concepts and technologies to improve the peer code review process:

- Allow opt-in and self-organization. A key precept of any social media community is that it is self-organizing. Allowing team members to subscribe to various code reviews, take part in threaded discussions about code and provide feedback as they wish, creates a more flexible approach to the code review process.
- Make reviews open and asynchronous. Why the need for in-person meetings? Establishing an asynchronous process that does not require in-person meetings can support geographically-dispersed teams and broaden the number of people who can provide input into the review.

Embracing tools that facilitate collaboration and automation while simplifying the entire process will help software development teams modernize the code review process:

- Support reviews with simple collaboration tools. The technology needed to support open and flexible collaboration for code reviews is simple, and it should offer features such as support of asynchronous reviews, compatibility with geographically dispersed teams and access to static analysis results.
- Equip developers with static analysis tools. The peer code review process should not become bogged down by issues that can be discovered automatically by a static analysis tool. Static code analysis can help developers find and eliminate simple mistakes prior to the code review milestone, allowing the code review team to focus on determining if the code they're reviewing matches the original intention.

By incorporating these modern techniques and strategies into the code review process, organizations can overcome the common barriers to effective reviews and start unleashing their full benefit.

About the author

Alen Zukich is Klocwork's Director of Product Management and sets the company's product direction. With a technical and consulting background in the telecom equipment and software tools markets, Alen has helped hundreds of organizations successfully deploy source code analysis technology within some of the most demanding and complex development environments in the world.



[print](#) [email](#) [rss](#) [SHARE](#) [post comment](#) 1 Comments

Comments



KB3001

7/7/2010 12:29 PM EDT

It's the constant pressure to bring new products to market at a faster pace that is driving people away from such good practices. Unless the cost of not doing so is clear and measurable, software developers will continue relying on ad-hoc practices. But what about Education? Do not you think that it could play a major role in embedding such practices into the minds of future software developers. How could industry and academia work together to allow for this?

[Sign in to Reply](#)

Please sign in to post comment

[Submit Comment](#)

[Follow Comments](#)

More EE Times

- [Subscriptions](#)
- [Newsletters](#)
- [Reprints](#)
- [RSS Feeds](#)
- [Media Kit](#)
- [Sitemap](#)

- [About Us](#)
- [Privacy Policy](#)
- [EE Times Career Center](#)
- [Contact Us](#)
- [Email: feedback@eetimes.com](mailto:feedback@eetimes.com)

EE Times Network

- [EE Times Asia](#)
- [EE Times-China](#)
- [EE Times-India](#)
- [EE Times Europe](#)
- [EE Times Japan](#)
- [EE Times Korea](#)
- [Electronic Supply & Manufacturing China](#)
- [Microwave Engineering Europe](#)
- [MCAD Online](#)
- [TechOnline India](#)
- [DeepChip.com](#)
- [Design & Reuse](#)



All materials on this site copyright ©2010 EE Times Group, A UBM company. All rights reserved.