CS499 - Open Source software development

Lecture #03: Code Review – Guidelines

Dr. Igor Steinmacher

e-mail: Igor.Steinmacher@nau.edu

Twitter: @igorsteinmacher

NAU NORTHERN ARIZONA UNIVERSITY

But, Before

- Assignments and reminders

Code Review

- Finding issues prior to go to the repo
  - Sharing knowledge
  - Consistency in a code base
  - **Legibility**
  - **Accidental errors**
  - **Structural errors**
  - **Compliance**

What is Peer Code Review?

Manually analyzing a software artifact from other team members

It is a Quality Assurance practice

Here is the code!
Looks good! What about removing that comment and treating the exception?
</>

Code Review – What to review

Correct Syntax

Indentation

Alignment

Removing commented (non-useful comments)

Grammar / Naming

Spelling mistakes

Correct English

Variable, Function, Method names

Code Review – What to review

Duplicate Code

- DRY (Don’t Repeat Yourself)
- Maintaining duplicate code is hard

Technical Quality

- Code Logic
- Code conventions
  - Follow project conventions for style/naming
- Is it possible to condense code?
- Security vulnerabilities

Code Review – What to review

Error Handling

Are exceptions being captured/treated correctly?

Human readable messages being displayed

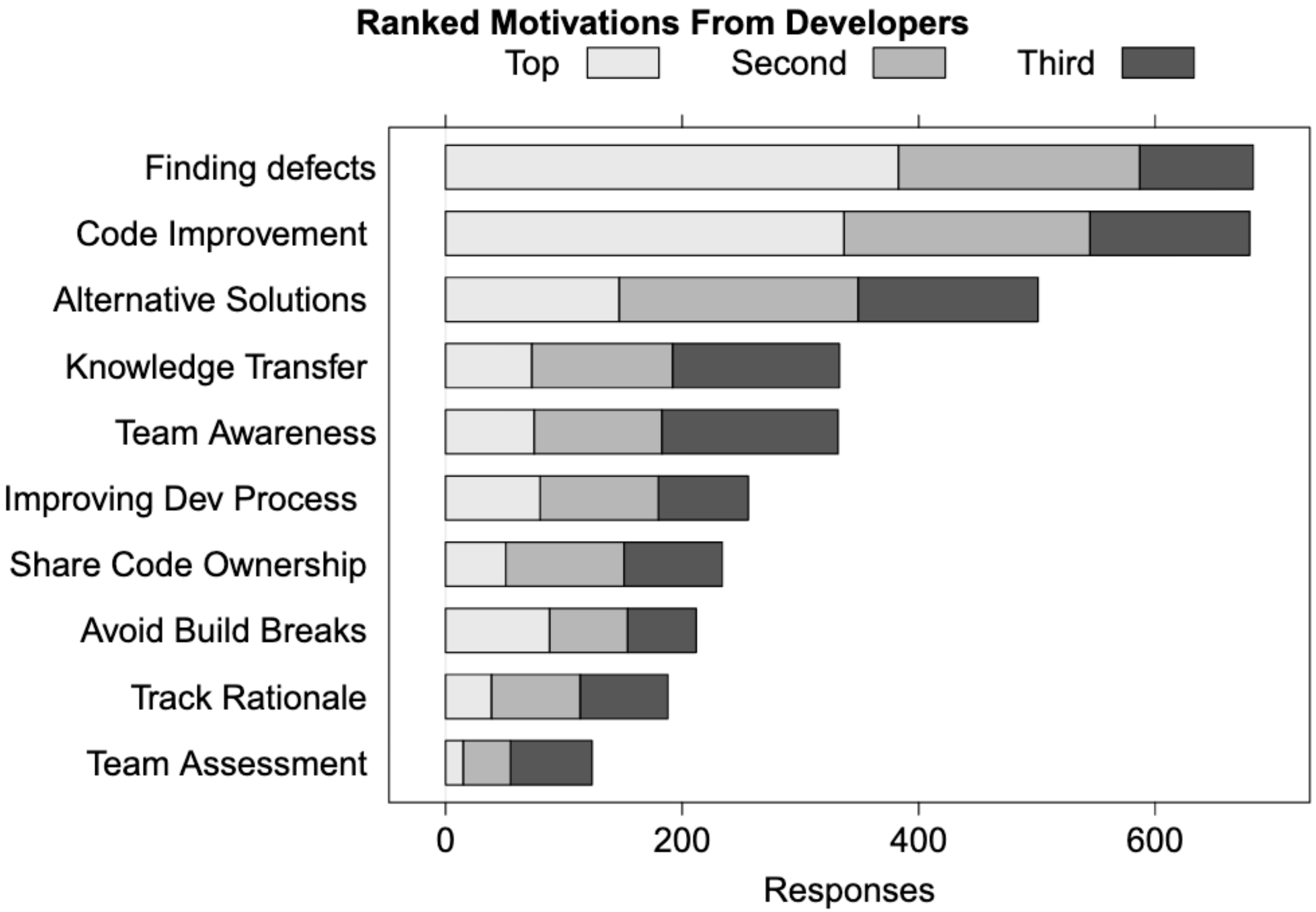
Test coverage/Unit tests

Code review is a learning experience.

Pay attention to what other people are saying. Ask questions!

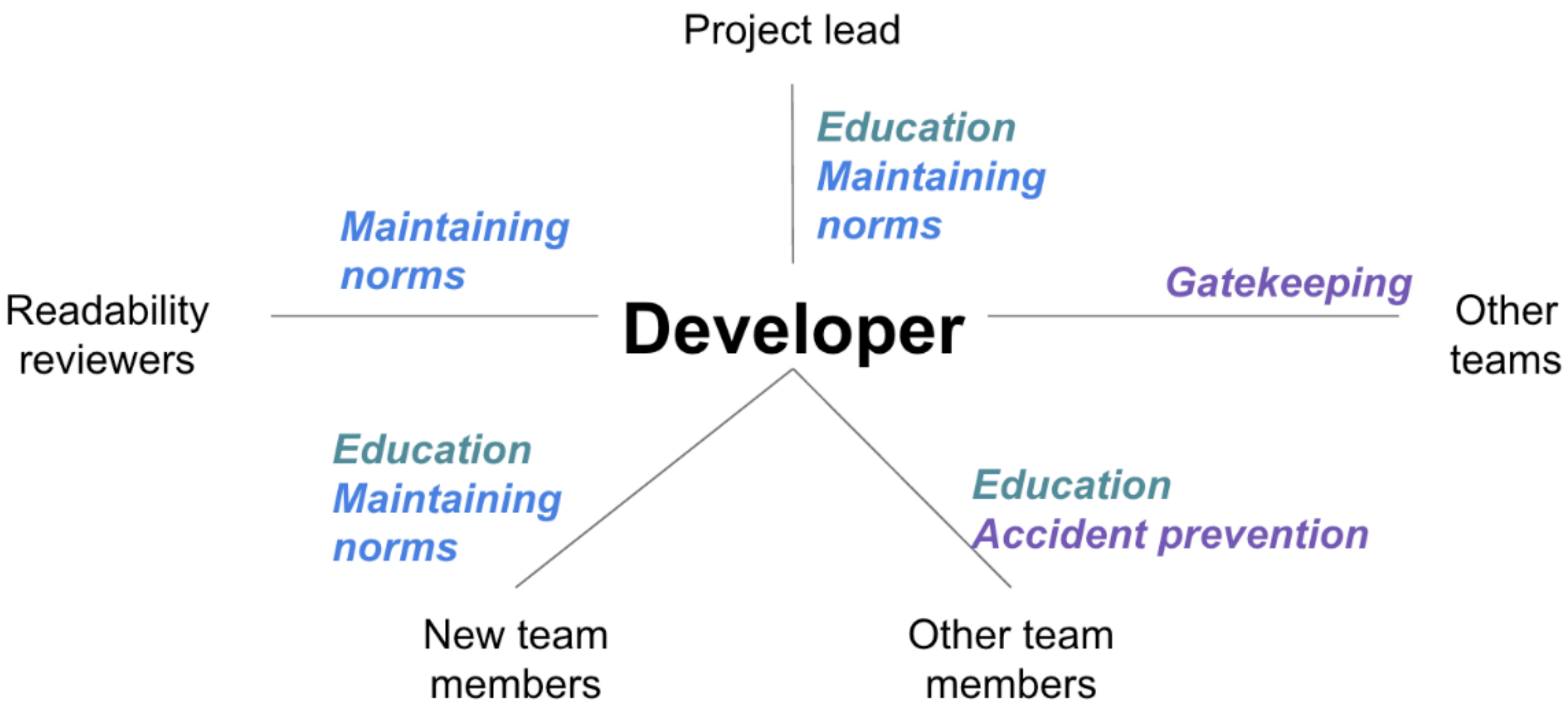
Poll: Why do we code review?

At Microsoft



A. Bachelli and C. Bird, "Expectations, Outcomes, and Challenges Of Modern Code Review," ICSE 2013

At Google



C. Sadowski, E. Söderberg, L. Church, M. Sipko, and A. Bacchelli, "Modern Code Review: A Case Study at Google," ICSE-SEIP 2018

Why???

Knowledge Transfer

- Newcomers can learn
- Team members can receive new information

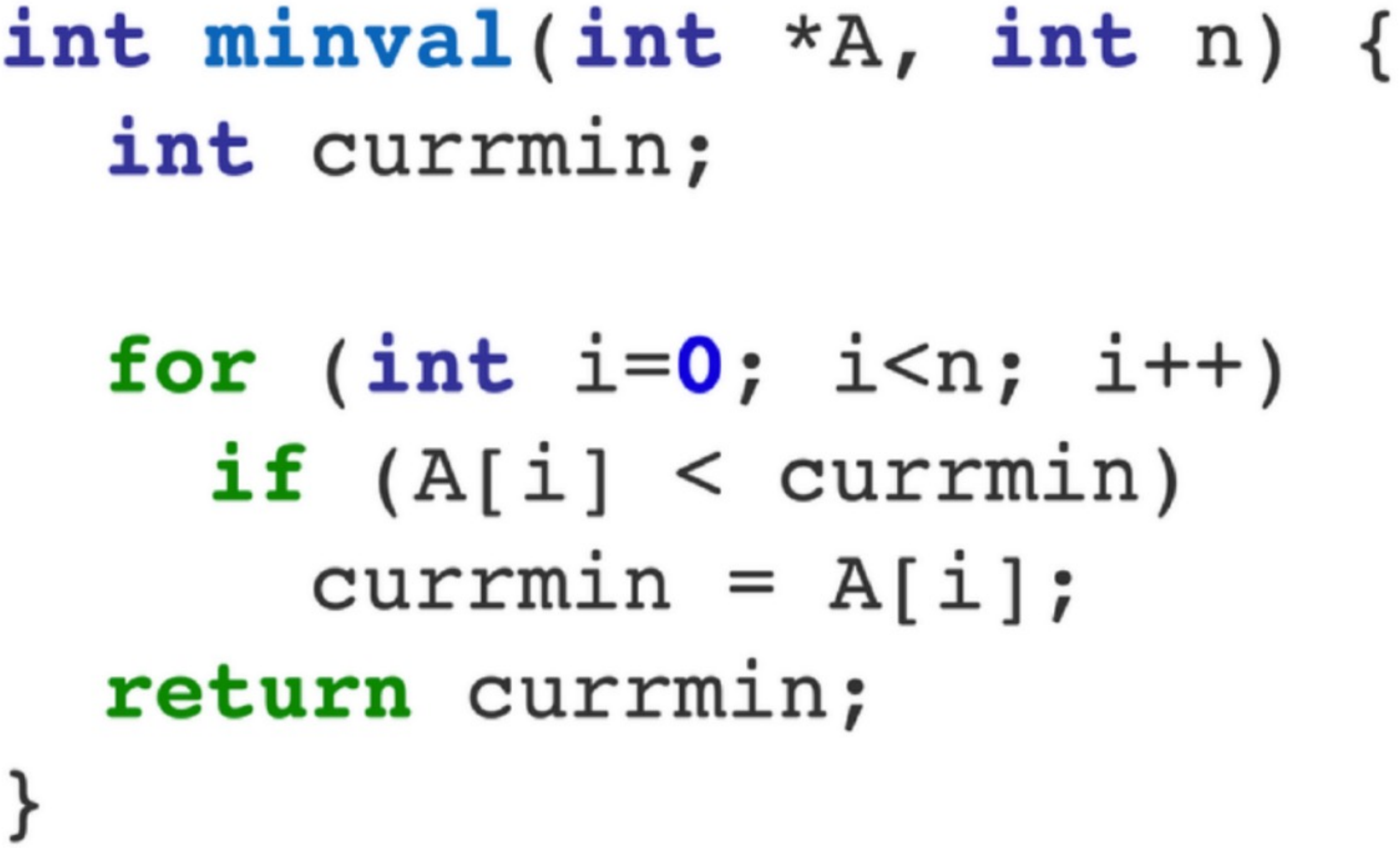
Team Awareness

- Sharing and updating the team with news and changes

Share Code Ownership

- The code with more people knowledgeable

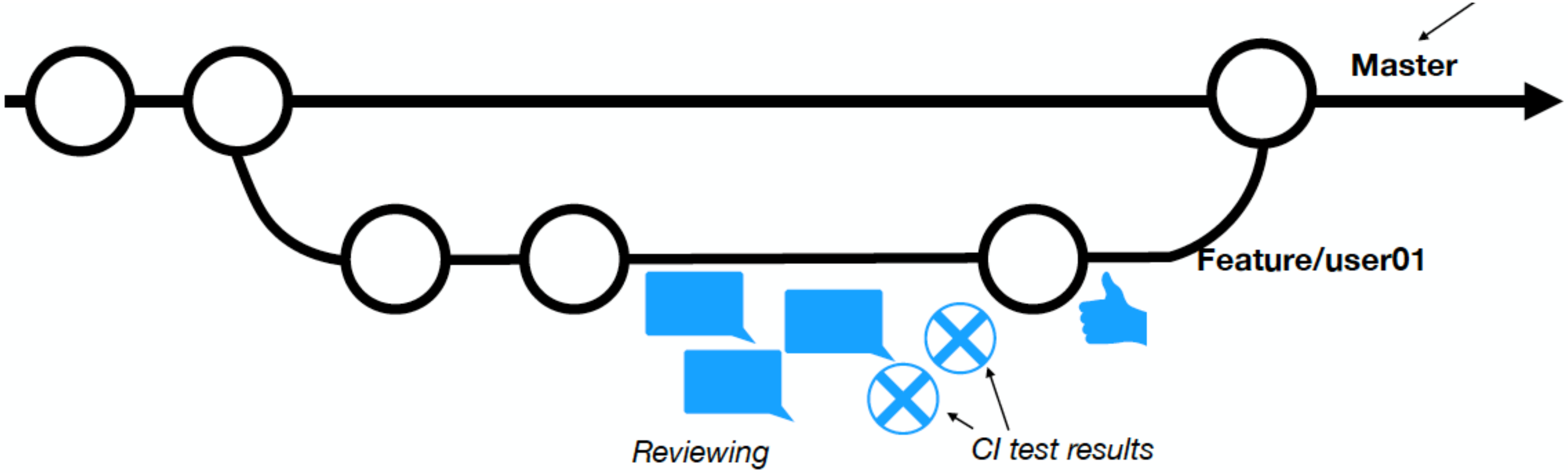
Where is the issue??



```
int minval(int *A, int n) {
  int currmin;

  for (int i=0; i<n; i++)
    if (A[i] < currmin)
      currmin = A[i];
  return currmin;
}
```

How?



How?

- **As a team**, you should
  - Build and maintain ***a positive review culture***.
  - Develop, reflect on, and revise ***code-reviewing policies***.
  - Ensure that ***time spent is counted*** and expected, but watch for negative impacts of assessments.
  - Ensure that the appropriate tools are available and used.
  - Promote the development of appropriate review checklists.
  - Have sufficient training in place for code review activities.
  - ***Develop a mechanism to watch for bottlenecks*** in the process

Code Review – Questions

- Does this code accomplish the purpose?
- How would you have solved the problem?
- How was the “reading” experience?
- Does the code follow to coding guidelines/style?
- Does this code introduce the risk of breaking builds?

Code Review – Questions

- Does this code break existing tests?
- Does the code need more tests?
- Was the documentation created/updated?
- Are there security vulnerabilities?
- Is this an efficient way? Any O(n^2) or worse algorithm?

Writing the Review

Don't make it personal.

Be nice

Be constructive

Be specific

Justify your points

Ask questions

HOW?

- **As a code author**, you should
  - Carefully check the code changes (including a sanity check) for a review
  - Cluster only related changes
  - Describe your changes and the motivation for them
  - Notify reviewers as early as possible
  - Promote an ongoing dialogue with reviewers
  - Track the suggested changes and confirm that they're fixed
  - Confirm that the decisions are documented

L. MacLeod, M. Greiler, M.-A. Storey, C. Bird, and J. Czerwonka, "Code Reviewing in the Trenches," IEEE Software., vol. 35, pp. 34–42, 2018. 33

HOW?

- **As a reviewer**, you should
  - Set aside dedicated, bounded time for reviews
  - Review frequently, doing fewer changes at a time
  - Provide feedback to authors as soon as possible
  - Focus on core issues first; avoid nitpicking
  - Give constructive, respectful feedback
  - Choose communication channels carefully; talk face-to-face for contentious issues (Don’t forget to document the conclusion!)
  - Be prepared to iterate and review again

L. MacLeod, M. Greiler, M.-A. Storey, C. Bird, and J. Czerwonka, “Code Reviewing in the Trenches,” IEEE Software., vol. 35, pp. 34–42, 2018. 33

Resources and More Resources

- There are many resources out there. These slides are based on some of them
  - https://mtlynch.io/human-code-reviews-1/
  - https://medium.com/palantir/code-review-best-practices-19e02780015f
  - https://smartbear.com/learn/code-review/best-practices-for-peer-code-review/
  - https://code.likeagirl.io/the-7-steps-to-a-complete-code-review-abdfd39e75f1
  - https://towardsdatascience.com/teaching-code-review-in-university-courses-using-peer-feedback-5625fe039f2a
  - https://en.wikipedia.org/wiki/Code_review
  - http://web.mit.edu/6.005/www/fa15/classes/04-code-review/

Let’s practice a Bit

- I will give you some code examples
- You will write the reviews for them
- We will discuss after some minutes