Reminders

- ❑ Mid-term 1: next **Monday, Sept. 25 in class**
- ❑ Project deliverable 1: due **Sunday, Oct 1, 2017 at 11:59 PM (extended)**

Cpt S 422: Software Engineering Principles II

Code reviews

Dr. Venera Arnaoudova

WASHINGTON STATE UNIVERSITY

Code inspections - as seen by Fagan

- In 1976, Fagan formalized a process for code reviews/ inspections
  - Line-by-line
  - Types of inspections:
    - ✓ID: after design, before code
    - ✓IC: after code, before unit testing
    - ✓IT: after unit test
  - 2 sessions of 2h each per day is considered acceptable
- Roles of the members participating in the inspections
  - Moderator: the coach, typically someone from an unrelated project
  - Designer
  - Coder/Implementor
  - Tester

Code inspections - as seen by Fagan (cont.)

- Outline of the inspection process
  1. **Overview** (whole team): the designer describes the area that is being addressed and the details of his design; documentation of the design is distributed to all participants. Not needed for IC
  2. **Preparation** (individual): members try to understand the design on their own (intent, logic, etc.). Checklists are recommended.
  3. **Inspection** (whole team): A “reader” chosen by the moderator (usually the coder) describes how he will implement the design. If any errors are found they are noted and classified. If a solution is obvious then it is noted as well. The moderator produces a report summarizing the findings within a day of the inspection.
  4. **Rework:** resolving the problems noted in the inspection phase.
  5. **Follow-up:** the moderator has to ensure that all problems have been fixed. If more than 5% of the material has been reworked then the team should do a re-inspection.

Modern Code Reviews (MCR)

❑ Informal (as opposed to the process described by Fagan)

❑ Tool based

❑ Performed regularly by many companies

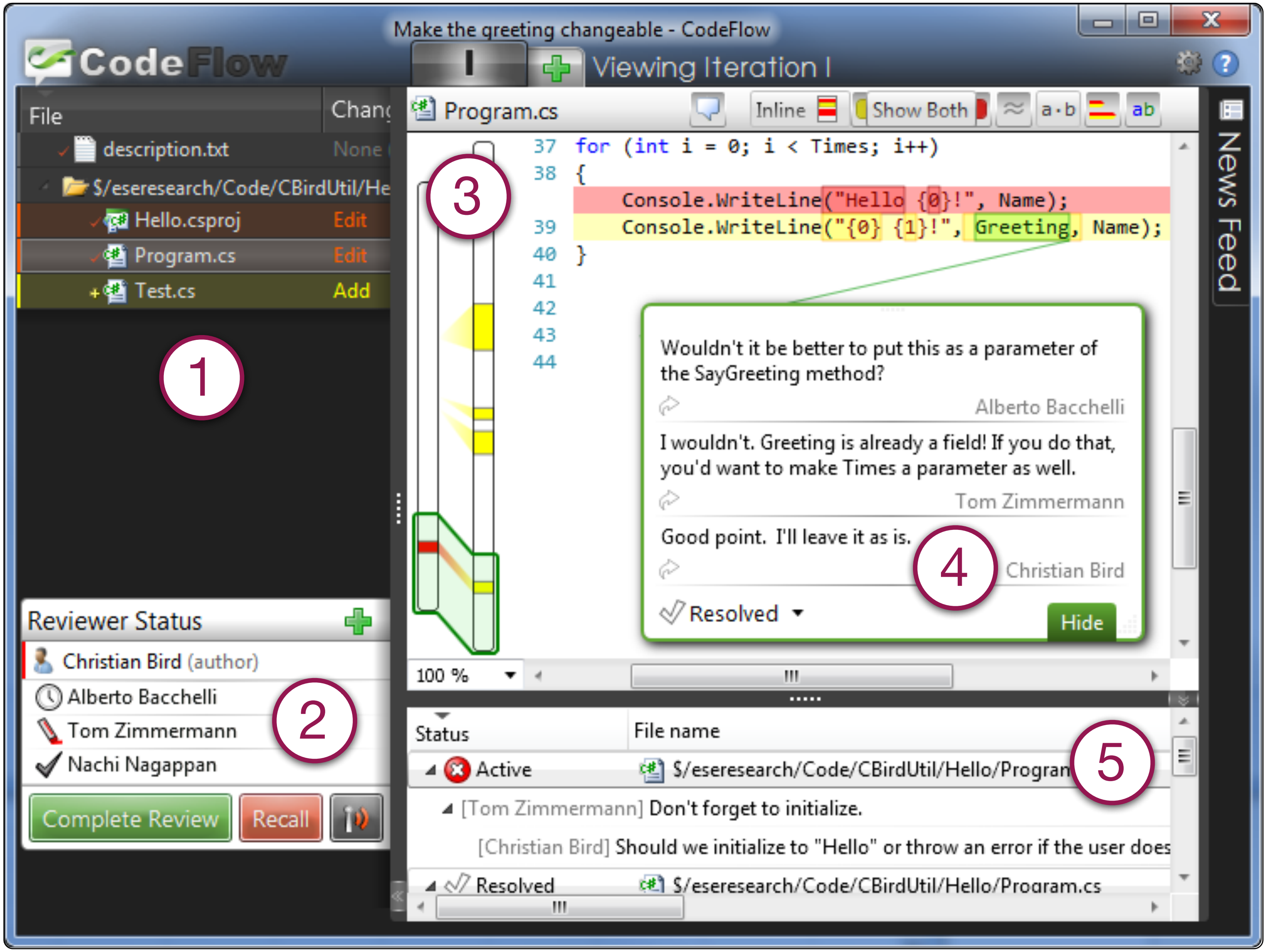
- ➢ Microsoft
- ➢ Google (Mondrian)
- ➢ Facebook (Phabricator)
- ➢ Open-Source Software projects (e.g., Gerrit)

Code review process

- Rigby et al. study different policies used by the Apache Server project:
  - Review-then-commit (RTC)
  - Commit-then-review (CTR)
  - Lazy consensus (silence implies consent)

Peter C. Rigby, Daniel M. German, and Margaret-Anne Storey. 2008. Open source software peer review practices: a case study of the apache server. In Proceedings of the International Conference on Software Engineering (ICSE '08), pp. 541-550.

Tools: CodeFlow (Microsoft)



File	Chang
description.txt	None
$/eseresearch/Code/CBirdUtil/He	
Hello.csproj	Edit
Program.cs	Edit
Test.cs	Add

```
for (int i = 0; i < Times; i++)
{
    Console.WriteLine("Hello {0}!", Name);
    Console.WriteLine("{0} {1}!", Greeting, Name);
}
```

Wouldn't it be better to put this as a parameter of the SayGreeting method?
Alberto Bacchelli

I wouldn't. Greeting is already a field! If you do that, you'd want to make Times a parameter as well.
Tom Zimmermann

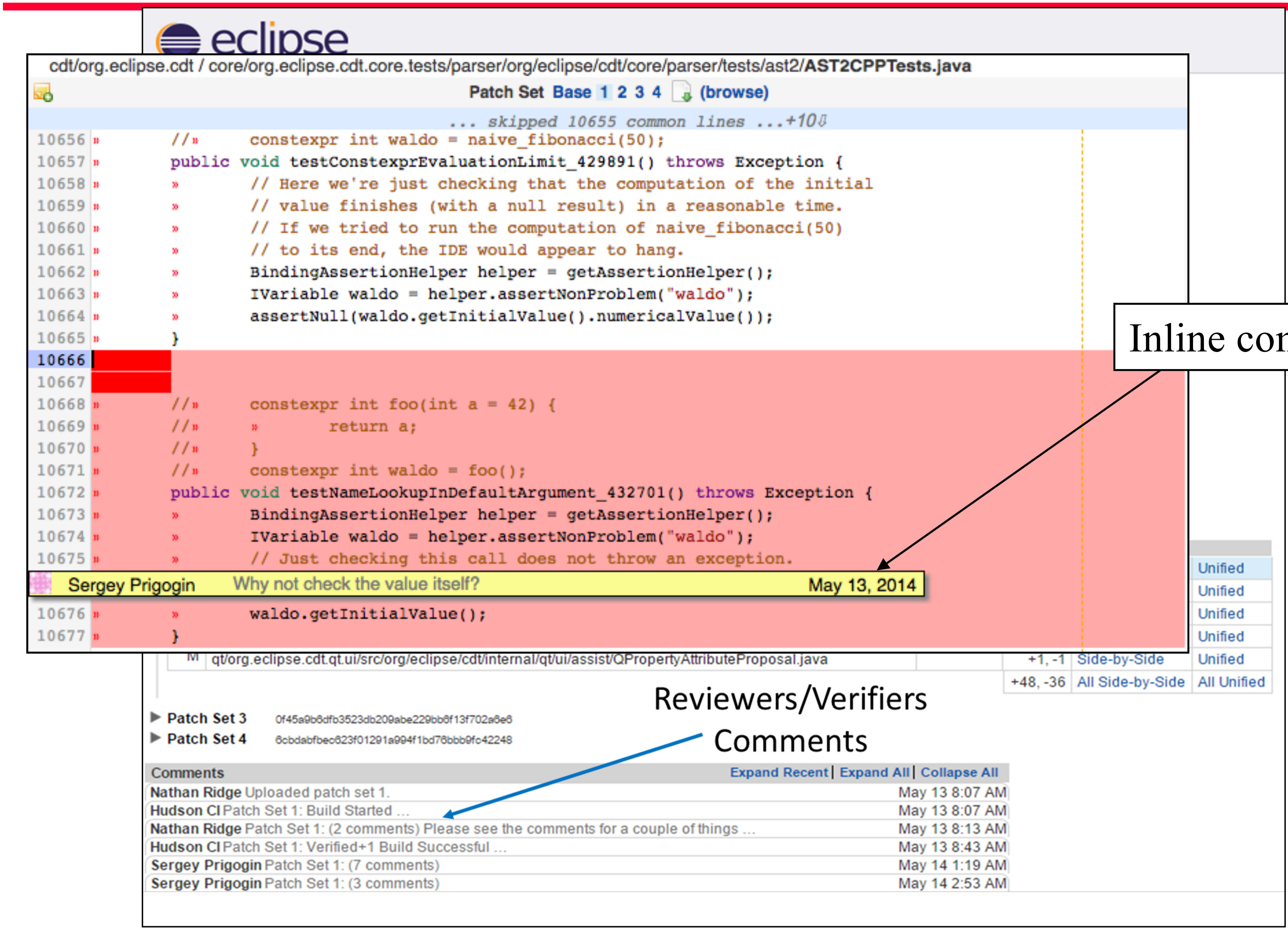
Good point. I'll leave it as is.
Christian Bird

Resolved

Christian Bird (author)
Alberto Bacchelli
Tom Zimmermann
Nachi Nagappan

Status	File name
Active	$/eseresearch/Code/CBirdUtil/Hello/Progran
[Tom Zimmermann] Don't forget to initialize.	
[Christian Bird] Should we initialize to "Hello" or throw an error if the user does	
Resolved	$/eseresearch/Code/CBirdUtil/Hello/Program.cs

Tools: Gerrit (Eclipse)



cdt/org.eclipse.cdt / core/org.eclipse.cdt.core.tests/parser/org/eclipse/cdt/core/parser/tests/ast2/**AST2CPPTests.java**

Patch Set Base 1 2 3 4 (browse)

```
... skipped 10655 common lines ...+10
//      constexpr int waldo = naive_fibonacci(50);
public void testConstexprEvaluationLimit_429891() throws Exception {
        // Here we're just checking that the computation of the initial
        // value finishes (with a null result) in a reasonable time.
        // If we tried to run the computation of naive_fibonacci(50)
        // to its end, the IDE would appear to hang.
        BindingAssertionHelper helper = getAssertionHelper();
        IVariable waldo = helper.assertNonProblem("waldo");
        assertNull(waldo.getInitialValue().numericalValue());
}


//      constexpr int foo(int a = 42) {
//              return a;
//      }
//      constexpr int waldo = foo();
public void testNameLookupInDefaultArgument_432701() throws Exception {
        BindingAssertionHelper helper = getAssertionHelper();
        IVariable waldo = helper.assertNonProblem("waldo");
        // Just checking this call does not throw an exception.
```

Sergey Prigogin Why not check the value itself? May 13, 2014

```
        waldo.getInitialValue();
}
```

				Unified
				Unified
				Unified
				Unified
M	qt/org.eclipse.cdt.qt.ui/src/org/eclipse/cdt/internal/qt/ui/assist/QPropertyAttributeProposal.java	+1, -1	Side-by-Side	Unified
		+48, -36	All Side-by-Side	All Unified

▶ **Patch Set 3** 0f45a9b6dfb3523db209abe229bb6f13f702a6e6

▶ **Patch Set 4** 6cbdabfbec623f01291a994f1bd76bbb9fc42248

Comments	Expand Recent | Expand All | Collapse All
Nathan Ridge Uploaded patch set 1.	May 13 8:07 AM
Hudson CI Patch Set 1: Build Started ...	May 13 8:07 AM
Nathan Ridge Patch Set 1: (2 comments) Please see the comments for a couple of things ...	May 13 8:13 AM
Hudson CI Patch Set 1: Verified+1 Build Successful ...	May 13 8:43 AM
Sergey Prigogin Patch Set 1: (7 comments)	May 14 1:19 AM
Sergey Prigogin Patch Set 1: (3 comments)	May 14 2:53 AM

Inline comment

Reviewers/Verifiers Comments

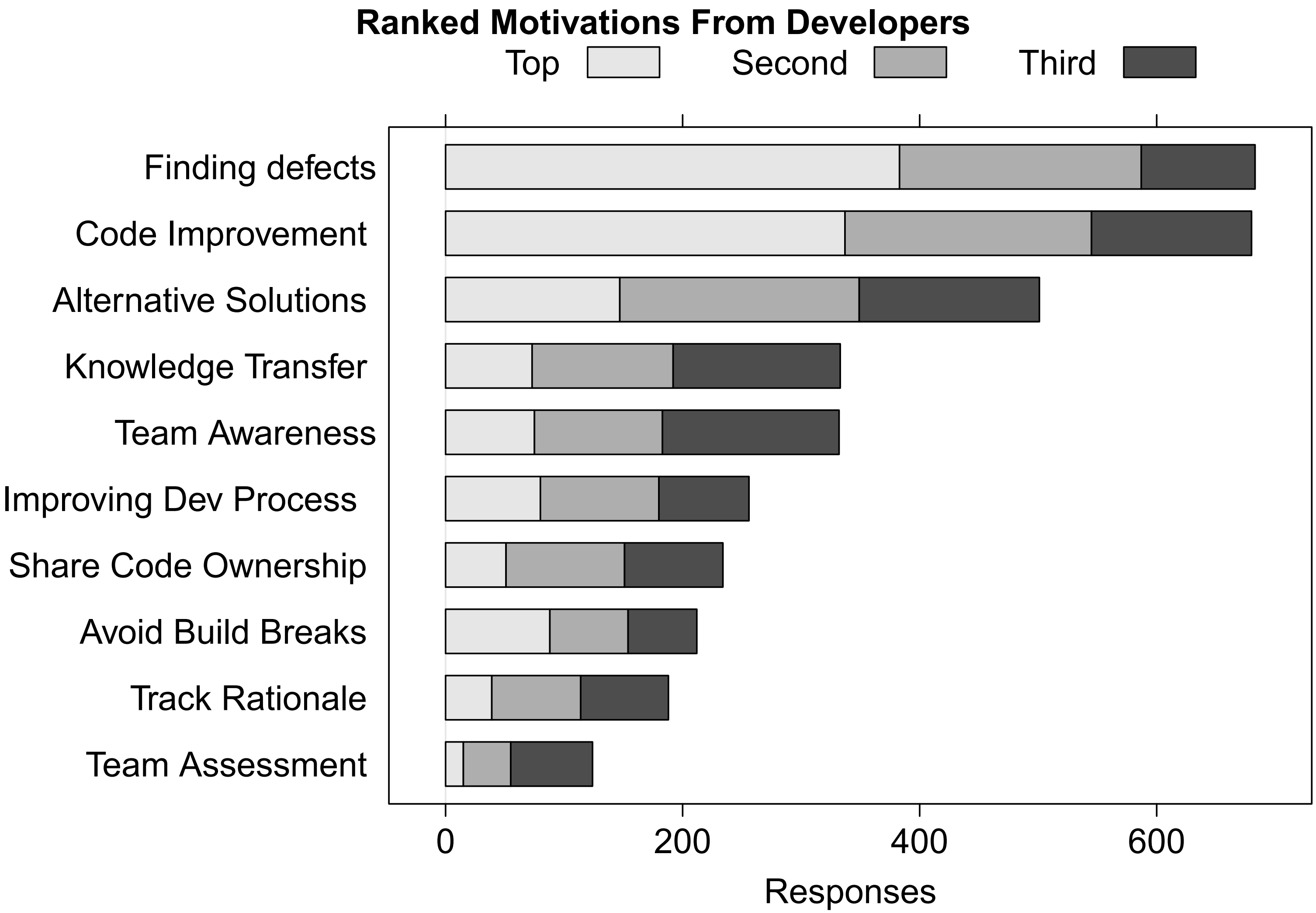
Expectations, outcomes, and challenges of MCR

- ❑ Study at Microsoft by Bacchelli and Bird
  - ➢ What are the expectations for code review nowadays?
  - ➢ What are the actual outcomes of code review?
  - ➢ What challenges do people face in code review?
- ❑ Methodology
  - ➢ Observing and interviewing 17 industrial developers performing code reviews
  - ➢ Manual inspection and classification of 570 code review comments
  - ➢ Surveying 165 managers and 873 programmers

Alberto Bacchelli and Christian Bird (2013). Expectations, outcomes, and challenges of modern code review. In Proceedings of the International Conference on Software Engineering (ICSE), pp. 712-721.

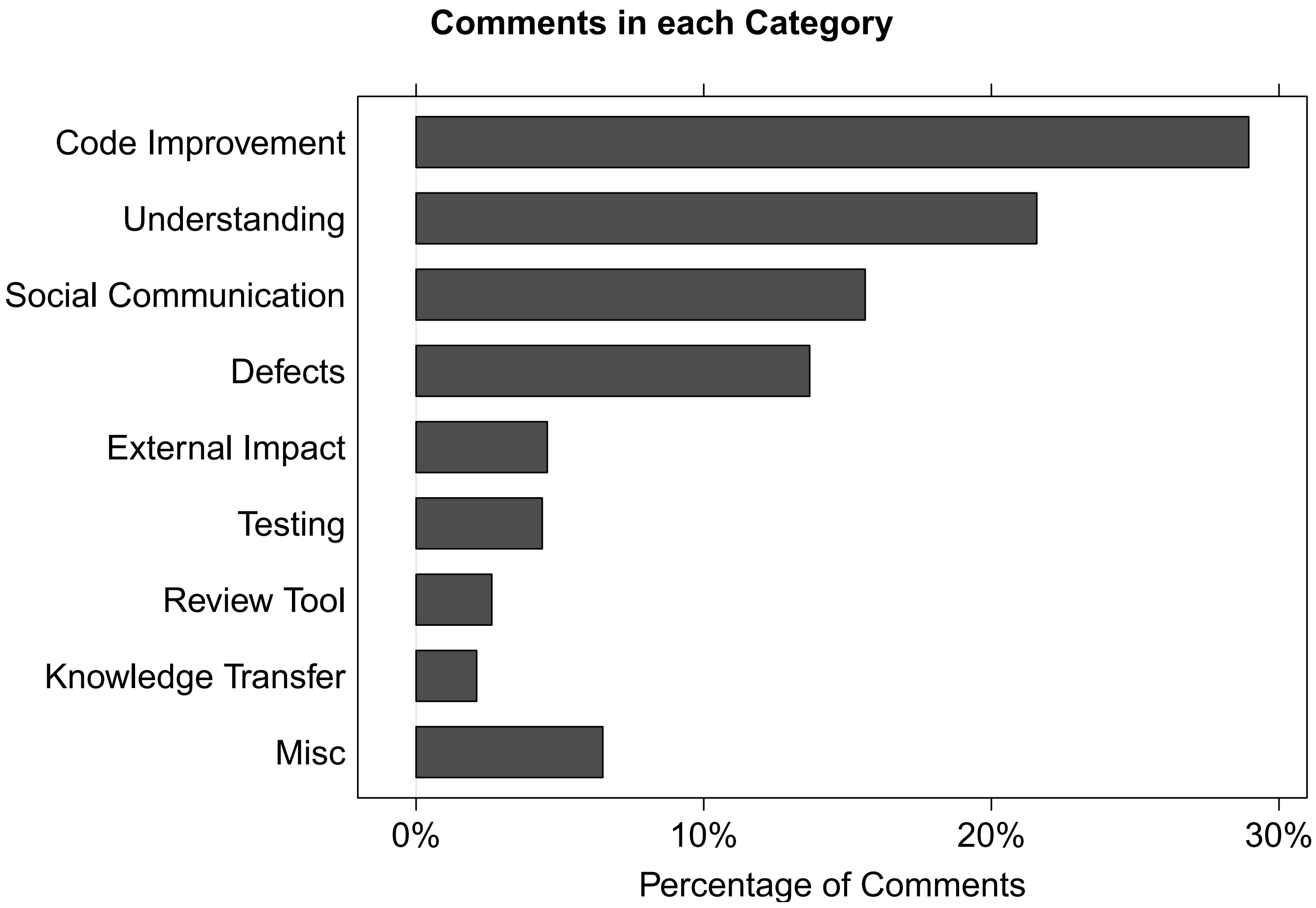
Motivation for Code Review

- ❑ Why do programmers do code reviews?



Outcomes of Code Reviews

- Sample of 570 code review comments



Summary of the results

- Motivation for code reviews
  - Finding defects
  - Code improvement
- Other benefits
  - Knowledge transfer
  - Team awareness
  - Understanding

Challenges

- ❑ Understanding
  - ➢ Code review submissions must include accurate summary of the changes
  - ➢ The changes must correspond to a cohesive change (small, independent, and complete)
  - ➢ Understanding needs change with the expected outcome of code review: The most difficult tasks from the understanding perspective are finding defects and alternative solutions
  - ➢ Top-down versus bottom-up approach for understanding the changes depending on whether the reviewer is familiar with the code or not
  - ➢ Tool limitation: although code review tools provide diffing capabilities, inline commenting, or syntax highlighting, often times reviewers and authors see the need to talk in person

Code review using pull requests

- Most code review tools require you to setup a server

- An alternative way to perform code reviews is to use pull requests, e.g.:
  - Create a branch for the feature/bug you are working on
  - Once you are done, commit your work and push it to your repository
  - Create a pull request from your branch
    - ✓ Describe what the work is about
    - ✓ Select reviewers
  - Address the reviewers comments until the code is accepted
  - The reviewer/verifier can then merge the code to the master

Tasks for today

- ❑ Sit next to your project teammates
- ❑ One of you must commit **the buggy version** of the Calendar example in a repository and share it with the rest of the team members
- ❑ Decide on one task for each team member (you can use tasks that already performed)
- ❑ Each team member should create a branch and work on the task. Once completed, the task must be submitted for review to the other team members.
- ❑ Once all comments of the reviewers are addressed, the branches must be merged to the master. (Each team member must do one merge)