CMPT 473
Software Quality Assurance

Code Reviews

Nick Sumner - Fall 2014

Code Reviews

- *In*formally, code reviews are techniques for discussing and sharing knowledge about code

Code Reviews

- *In*formally, code reviews are techniques for discussing and sharing knowledge about code

How many of you do code reviews as part of jobs / co-ops?

Code Reviews

- *In*formally, code reviews are techniques for discussing and sharing knowledge about code

How many of you do code reviews as part of jobs / co-ops?

- Why do code reviews?

Why Do Code Reviews?

Sharing knowledge about code

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote

What might you share?

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code someone else wrote

What might you give feedback on?

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code someone else wrote
  - Correctness (does the code do what was intended?)

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code someone else wrote
  - Correctness (does the code do what was intended?)
  - Clarity

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code someone else wrote
  - Correctness (does the code do what was intended?)
  - Clarity
  - Design (are there better alternatives?)

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code someone else wrote
  - Correctness (does the code do what was intended?)
  - Clarity
  - Design (are there better alternatives?)
  - Style (is another approach *more consistent*?)

Why Do Code Reviews?

Sharing knowledge about code

- Code you wrote
  - Reasons for its design
  - How it addresses intended problems
- Code
  - Correctness (does the code do what was intended?)
  - Clarity
  - Design (are there better alternatives?)
  - Style (is another approach *more consistent*?)

Code reviews build & maintain
institutional knowledge

How To Improve Code Reviews

- Practice

How To Improve Code Reviews

- Practice
- Practice

How To Improve Code Reviews

- Practice
- Practice
- Structure

How To Improve Code Reviews

- Practice
- Practice
- Structure
- Patience

How To Improve Code Reviews

- Practice
- Practice
- Structure
- Patience
- Learning how to do code reviews is difficult, and the best way to improve is through practice.

How To Improve Code Reviews

- Practice
- Practice
- Structure
- Patience
- Learning how to do code reviews is difficult, and the best way to improve is through practice.
- But... there are patterns & structure that help

What Are Code Reviews?

Many different approaches

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback
- Walkthroughs-
  - Developer guides the team or reviewers through changes to explain the code and design

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback
- Walkthroughs-
  - Developer guides the team or reviewers through changes to explain the code and design
- Inspections-
  - Reviewer (& sometimes mediator) examines the code and focuses on core objectives (security, API, …)

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback
- Walkthroughs-
  - Developer guides the team or reviewers through changes to explain the code and design
- Inspections-
  - Reviewer (& sometimes mediator) examines the code and focuses on core objectives (security, API, …)
- Audits-
  - Third party inspections of conformance & quality

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback
- Walkthroughs-
  - Developer guides the team or reviewers through changes to explain the code and design
- Inspections-
  - Review[illegible] the code and focuses on core objectives (security, API, …)
- Audits-
  - Third party inspections of conformance & quality

Why / when might you do audits?

What Are Code Reviews?

Many different approaches

- Passarounds-
  - Developer submits code to many teammates for feedback
- Walkthroughs-
  - Deve[illegible]gh chang
- Inspections-
  - Reviewer (& sometimes mediator) examines the code and focuses on core objectives (security, API, …)
- Audits-
  - Third party inspections of conformance & quality

Also less formal approaches:
ad hoc reviews, pair programming, etc.

What Are The Outcomes?

- Depends on the approach

What Are The Outcomes?

- Depends on the approach
  - e.g. Walkthroughs- informal feedback

What Are The Outcomes?

- Depends on the approach
  - e.g. Walkthroughs- informal feedback
  - e.g. Inspections- (Accept, accept w. change, redo)

What Are The Outcomes?

- Depends on the approach
  - e.g. Walkthroughs- informal feedback
  - e.g. Inspections- (Accept, accept w. change, redo)
  - e.g. Audits- Full reports on overall quality

What Are The Outcomes?

- Depends
  - e.g. Wall
  - e.g. Insp
  - e.g. Audi

MY REVIEW OF TOYOTA'S SOURCE CODE

Access to Toyota's "electronic throttle" source code
- In a secure room in Maryland
- Subject to confidentiality agreements
- For vehicle models with ETCS spanning ~2002-2010 model years
  Camry, Lexus ES, Tacoma, and others

Approximately 18 months of calendar time with code
- By a very experienced team of embedded systems experts
  Including 3 other engineers from Barr Group
- Building upon NASA's earlier source code review; digging deeper

5

NASA must reach a clear-cut conclusion by the end of August.
So they are under a fair amount of pressure.

TOY-MDL05951378
TOY-MDL05951378P-0001

BARR group

From Barr's audit of Toyota code:
http://www.safetyresearch.net/Library/BarrSlides_FINAL_SCRUBBED.pdf

What Are Code Reviews?

- Most often, people think of “passarounds”:

  e.g. http://llvm-reviews.chandlerc.com/D3009

What Are Code Reviews?

- Most often, people think of “passarounds”:

  e.g. http://llvm-reviews.chandlerc.com/D3009

- Patches with explanations & feedback with {accept, accept w/changes, change & resubmit}

What Are Code Reviews?

- Most often, people think of “passarounds”:

  e.g. http://llvm-reviews.chandlerc.com/D3009

- Patches with explanations & feedback with {accept, accept w/changes, change & resubmit}
- More rigorous approaches (inspections,audits,...) are more likely to find bugs

When To Review

What approaches have you used?

When To Review

What approaches have you used?

- Can vary with institutional process

When To Review

What approaches have you used?

- Can vary with institutional process
- Good approach:
  - *Informal review before every commit!*

When To Review

What approaches have you used?

- Can vary with institutional process
- Good approach:
  - *Informal review before every commit!*
- Increasingly in depth reviews as necessary

When To Review

What approaches have you used?

- Can vary with institutional process
- Good approach:
  - *Informal review before every commit!*
- Increasingly in depth reviews as necessary
- Regularly scheduled walkthroughs/inspections.

Why?

Passarounds

Not a formal term

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?
  - Why might larger be a problem?

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?
  - Why might larger be a problem?

Why might reviewing smaller sections of code be reasonable?

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?
  - Why might larger be a problem?
- Review in short periods – Why?!

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?
  - Why might larger be a problem?
- Review in short periods – Why?!
  - If your attention wanes, it is useless

Passarounds

- Review cohesive but small changes
  - No more than 400 lines of code at a time is reasonable
  - Why might smaller be a problem?
  - Why might larger be a problem?
- Review in short periods – Why?!
  - If your attention wanes, it is useless
  - Fast/prompt feedback is crucial to progress

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality
  - Range from looking at code on a projector to discussions in front of a whiteboard

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality
  - Range from looking at code on a projector to discussions in front of a whiteboard
- Everyone should read the code in advance to *look for* issues

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality
  - Range from looking at code on a projector to discussions in front of a whiteboard
- Everyone should read the code in advance to *look for* issues

Why don't we want to fix the issues now?

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality
  - Range from looking at code on a projector to discussions in front of a whiteboard
- Everyone should read the code in advance to look for issues
- Knowledge & design decisions are explicitly disseminated throughout the team

Walkthroughs

- Guide the team / reviewers through changes or through the code and its design
  - A middle ground of formality
  - Range from looking at code on a projector to discussions in front of a whiteboard
- Everyone should read the code in advance to look for issues
- Knowledge & design decisions are explicitly disseminated throughout the team
- Shouldn't last more than an hour

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria

Often, these are *required*,
although we won't use them in our exercises

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria
- Driven by 4 roles:
  - Moderator, Author, Reviewer, Scribe

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria
- Driven by 4 roles:
  - Moderator, Author, Reviewer, Scribe

Keeps meeting moving.
Makes sure that reported items are acted on.

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria
- Driven by 4 roles:
  - Moderator, Author, Reviewer, Scribe

Explains the code.
Answers questions of the Reviewer.

Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria
- Driven by 4 roles:
  - Moderator, Author, Reviewer, Scribe

Searches for issues in the code.
Prepares in advance for the inspection meeting.

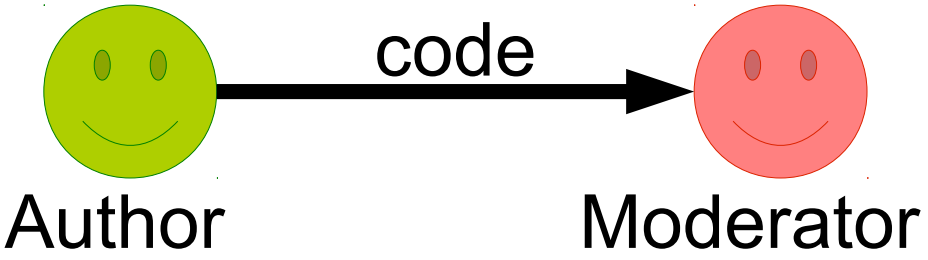
Inspections

- Reviewer (& sometimes mediator) examines the code and focuses on core objectives
- Thorough inspections can eliminate 70-85% of bugs
- Best with preparation & a focused checklist of criteria
- Driven by 4 roles:
  - Moderator, Author, Reviewer, Scribe

Records the issues and proposed actions.

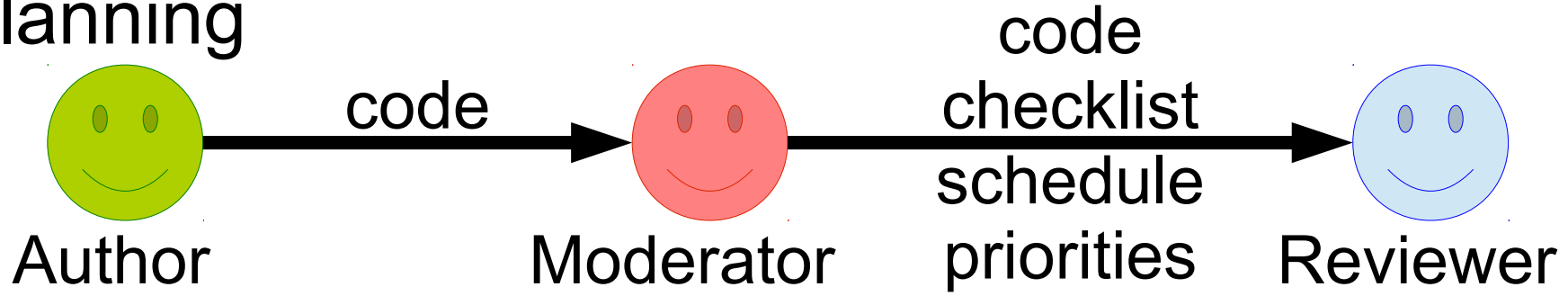
Inspection

- Planning



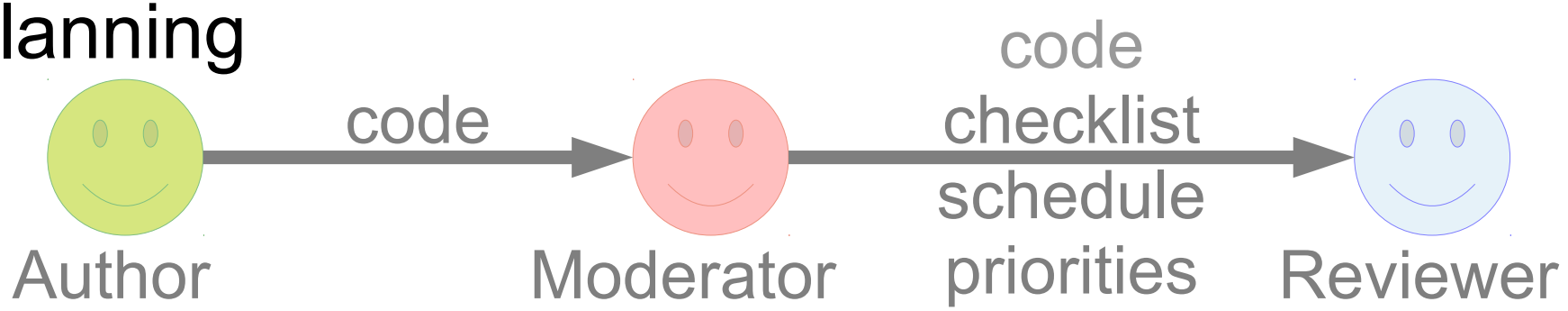
Inspection

- Planning



Inspection

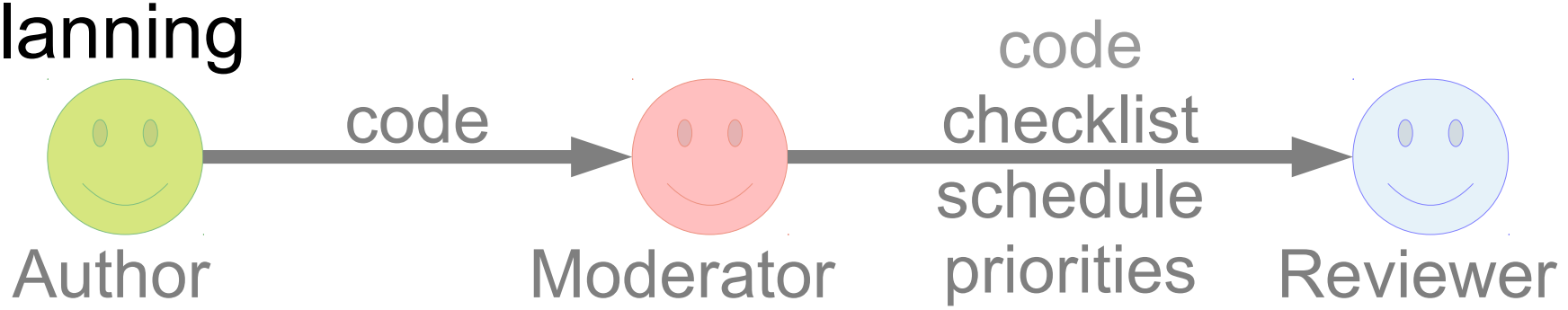
- Planning



- Overview? → A risky option

Inspection

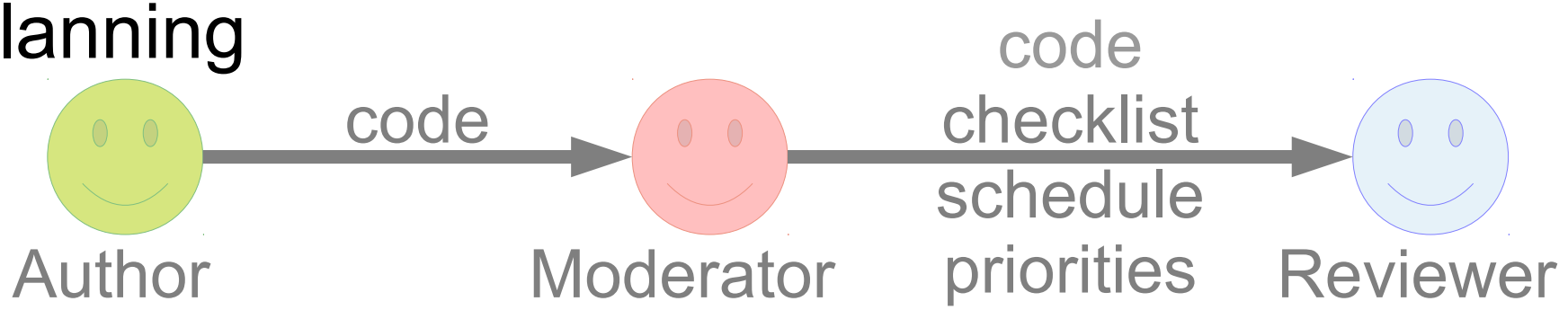
- Planning



- Overview? → A risky option
- Preparation

Inspection

- Planning



- Overview? → A risky option
- Preparation

Inspection

- Inspection Meeting
  - Moderator selects a nonauthor to lead through and explain each line & all logic in the code
  - Scribe records all errors

Inspection

- Inspection Meeting
  - Moderator selects a nonauthor to lead through and explain each line & all logic in the code
  - Scribe records all errors
  - Discussion of an error stops once it is detected

Why isn't it fixed?

Inspection

- Inspection Meeting
  - Moderator selects a nonauthor to lead through and explain each line & all logic in the code
  - Scribe records all errors
  - Discussion of an error stops once it is detected
- Report
  - Each defect along with checklist violation & severity is disseminated

Inspection

- Inspection Meeting
  - Moderator selects a nonauthor to lead through and explain each line & all logic in the code
  - Scribe records all errors
  - Discussion of an error stops once it is detected
- Report
  - Each defect along with checklist violation & severity is disseminated
- Fixing & Followups
  - Fixes are assigned & ensured by the moderator

Ego

- Code reviews are socially troublesome in the same way as bug reporting

Ego

- Code reviews are socially troublesome in the same way as bug reporting
- Effective reviews ***must be egoless***

Exercises

- Let's walk through some code.

Exercises

- Let's walk through some code.
- Let's walk through *your* code.
  - Have one author and one scribe for each session.
  - The moderator shouldn't be the scribe, but for us....
  - Turn in your notes at the end.
  - Record the names of the author, reviewer, & scribe for each.