Công Cụ & Phương Pháp Thiết Kế - Quản Lý (Phần Mềm)

TRAN KIM SANH
Instructor of DTU

Email: trankimsanh@dtu.edu.vn
Tel: 0987 409 464

Reviewing the Review Process

Contents

1. Why Do Technical Reviews?
2. Benefits of Technical Reviews
3. Technical Review Costs
4. Addressing Hurt Feelings
5. What Data Should We Collect?
6. Why Don’t More Software Teams use
7. Technical Reviews?

Question

- **All studies of Inspection have common results, the meeting will find very few errors compared to the reading code. Why are many companies still inspecting the code by meeting?**

  A. They use inspection for training

  B. Inspection can find the defect that the individual couldn't found

  C. Meetings create a schedule that people must work towards

  D. All above

Referenced from Prof.Redley of CMU

Question

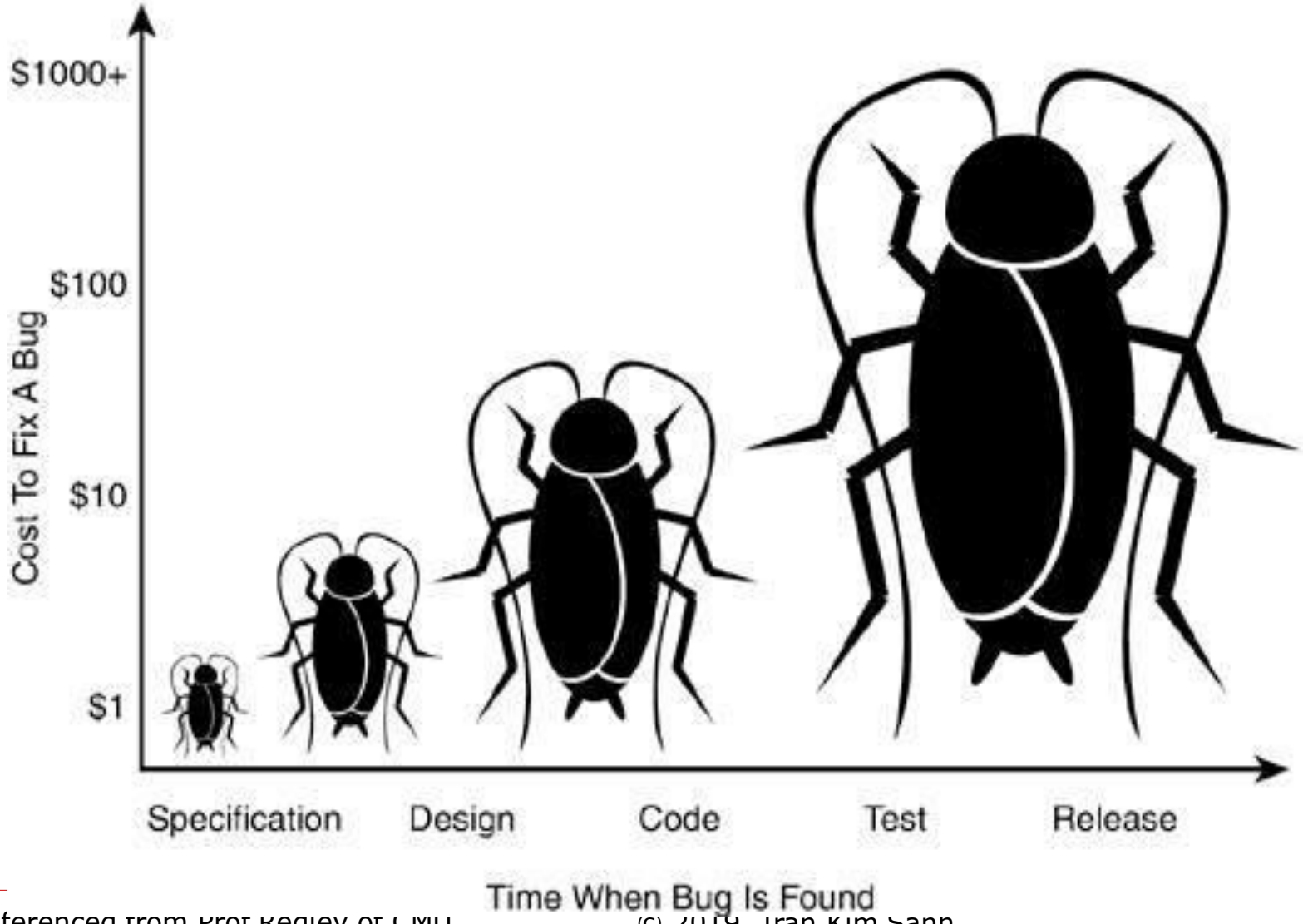
- **What is the maximum time to peer code review?**
  A. 30 minutes
  B. 60 minutes
  C. 90 minutes
  D. 120 minutes
- **What is the most successful type of Object Oriented's review?**
  A. Checklist review
  B. Systematic review
  C. Use-case review
  D. No solution is true

Referenced from Prof.Redley of CMU

Why Do Technical Reviews?

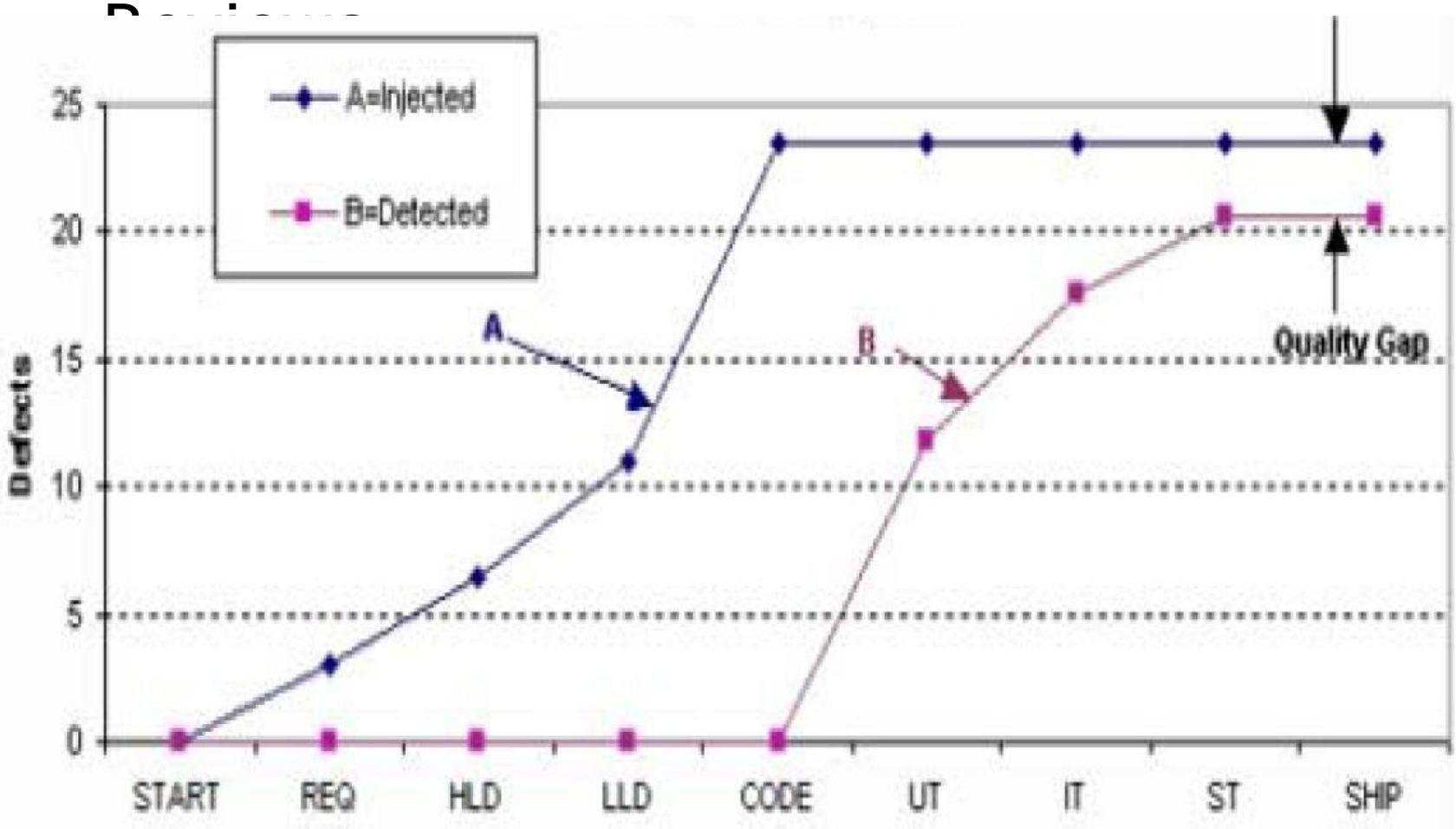
- Software Development Goal:
  - To Remove Software Defects at a Reduced Cost
- Technical Reviews remove defects early in the Life Cycle, and it is always cheaper to remove defects earlier than later
  - Note that Technical Reviews help remove defects, and prevent future defects

Cost to fix a bug



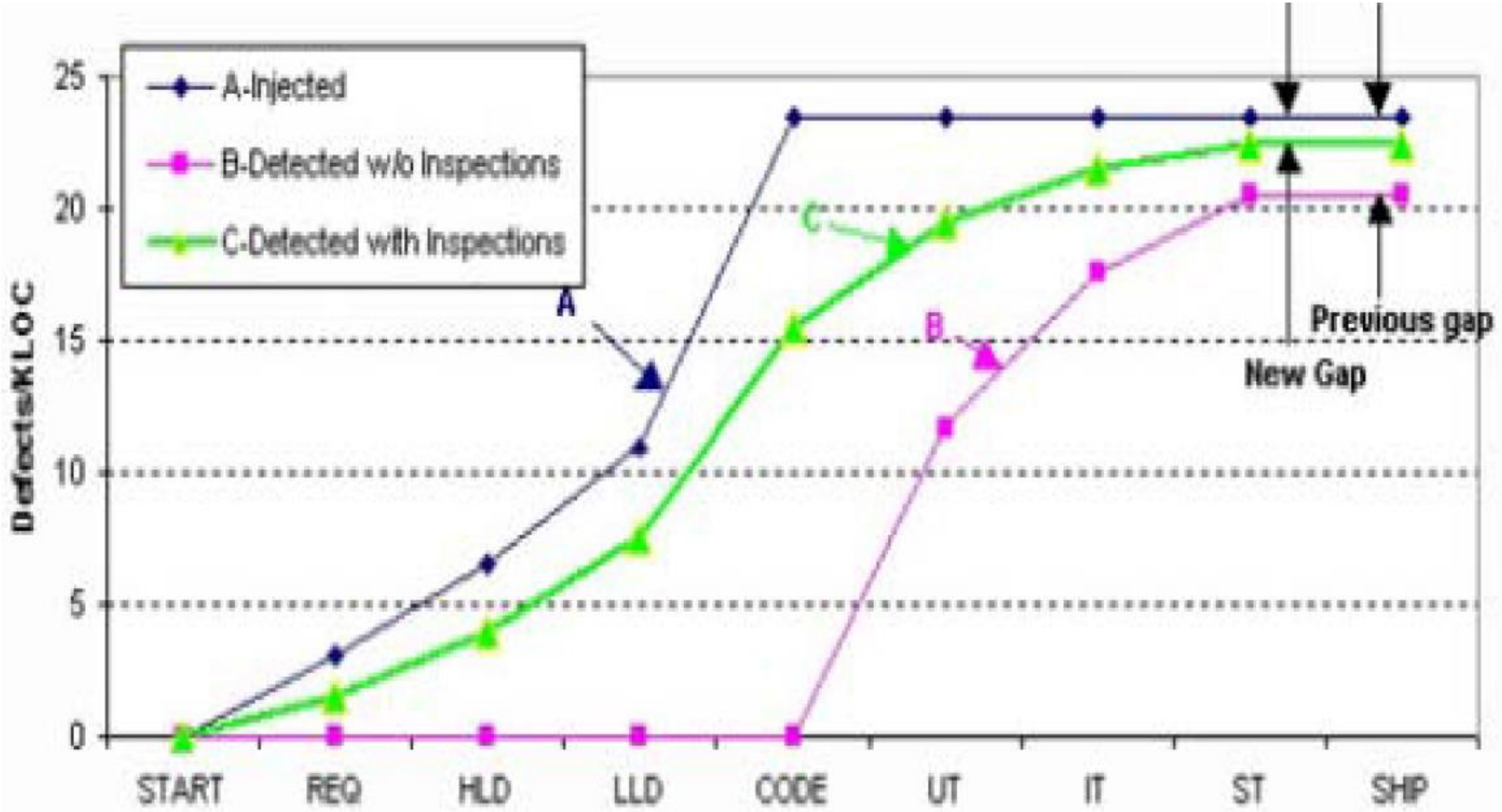
Technical Review Theory

- Software Defect Removal Without



Technical Review Theory

- Software Defect Removal With Reviews



Technical Review Benefits

- Early removal of defects
- Teams find faults that no individual reviewer would be able to find
- Less experienced developers and reviewers learn from their more experienced peers
- Meetings create a schedule that people must work towards
- Personal incentive to contribute & improve
- Significant knowledge sharing

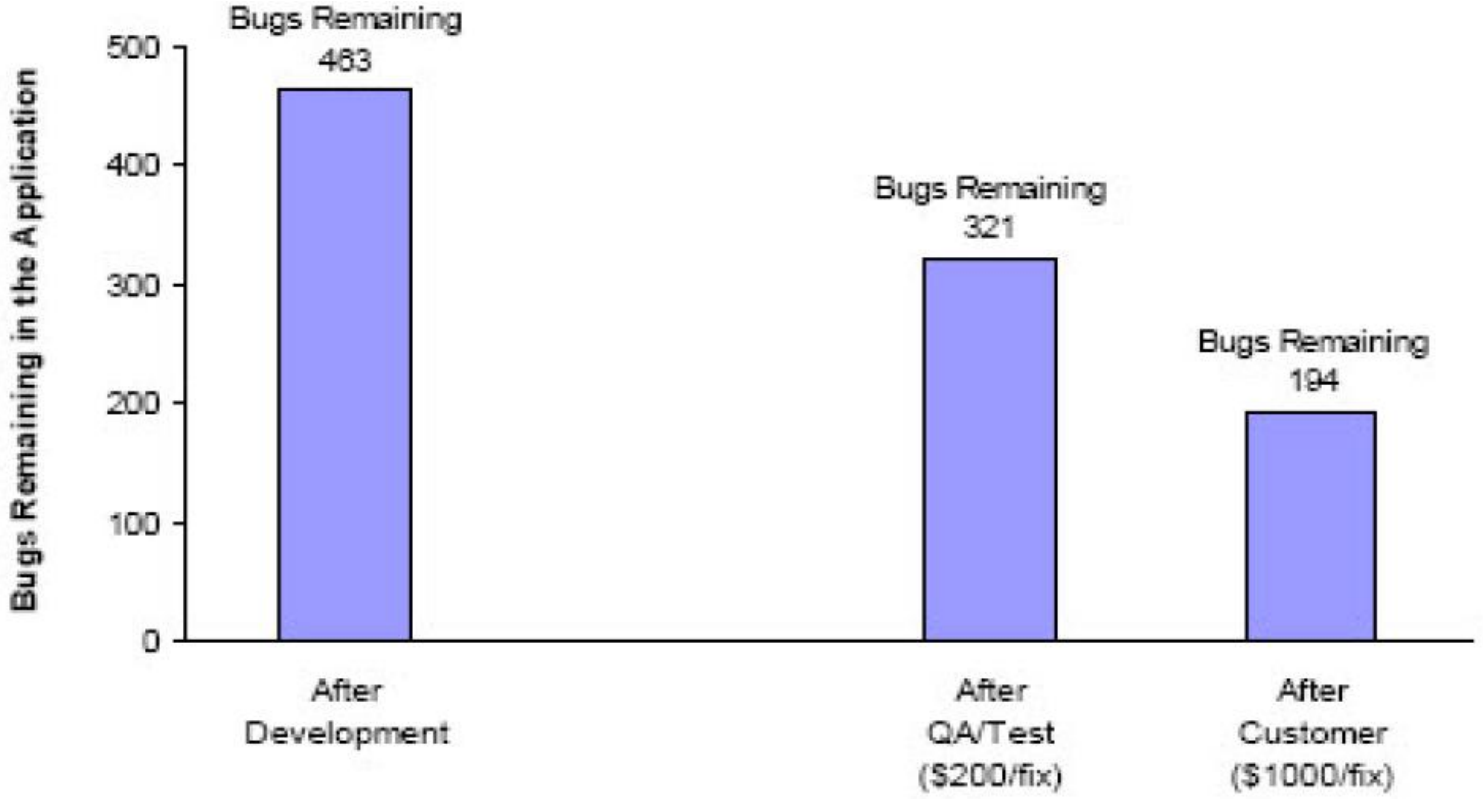
Ex. 1

- Three-month, 10KLOC project with 10 developers. How much $ would the company have saved if they had used technical reviews?
- The result:
  - Code review would have saved *half the cost of fixing the* bugs
  - Plus they would have found 162 additional bugs

Jason Cohen. Best Kept Secrets of Peer Code Reviews

Ex. 2 - Before

- Before Code Reviews

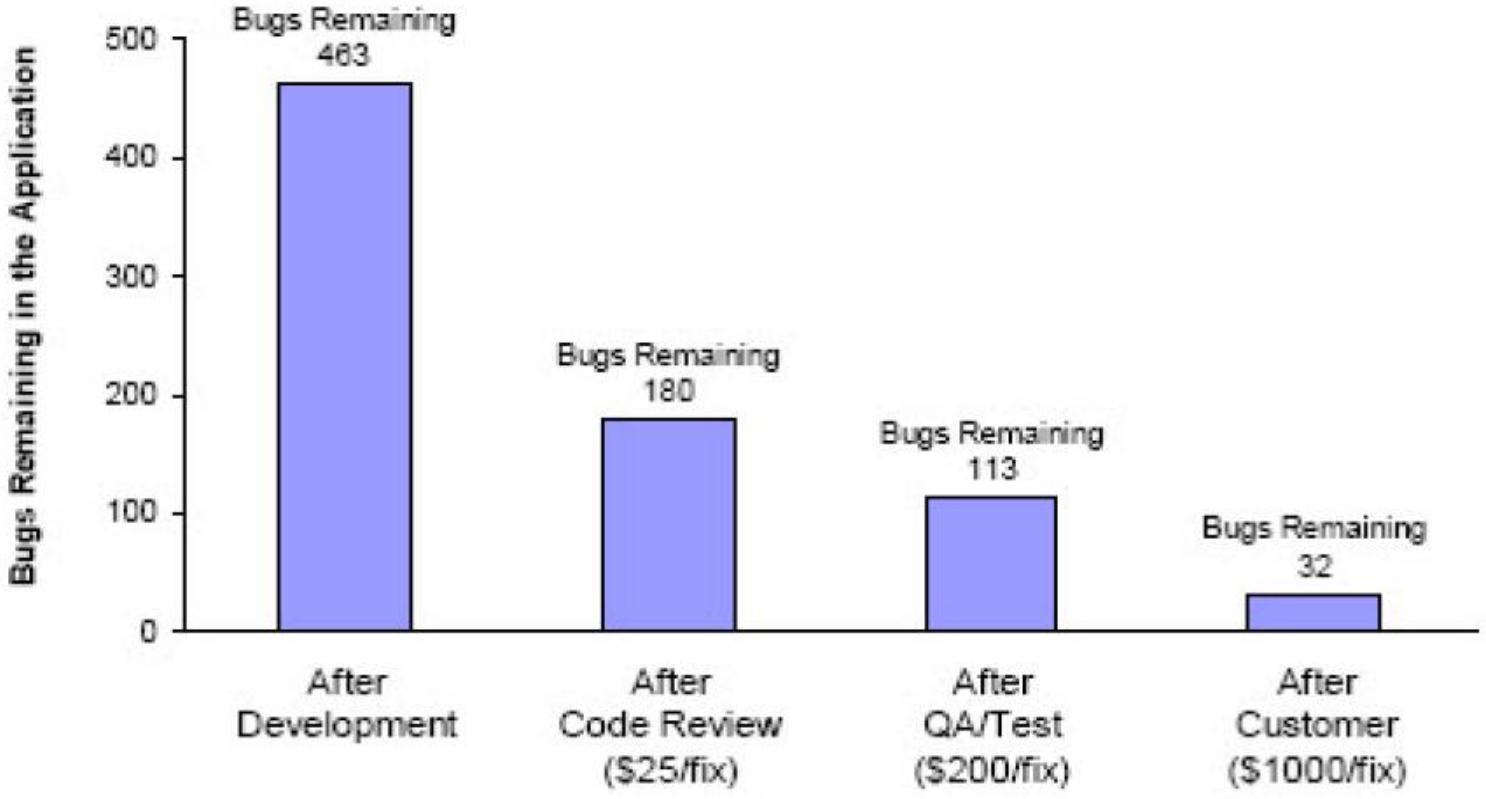


Jason Cohen. Best Kept Secrets of Peer Code Reviews

Cost of fixing bugs: $174k
\+ Cost of 194 latent bugs: $194k
Total Cost: $368k

Ex. 2 - After

- After Code Reviews



Jason Cohen. Best Kept Secrets of Peer Code Reviews

Cost of fixing bugs: $120k
\+ Cost of 32 latent bugs: $ 32k
Total Cost: $152k

Benefits from Other Studies

- Design & Code Inspections usually remove 70 – 85% of product defects
  - Capers Jones. *Software Defect Removal Efficiency, IEEE Computer. April 1996*
- Inspections increase productivity ~ 20%
  - Multiple citings

Technical Review Costs

- ~ 10-15% of project budget
  - If designs and code are inspected
- Vary Widely Depending on review type
  - Effort associated with preparing, doing and documenting the review
  - Cost of training
  - Cost of tools

- Hurt Feelings
  - Any criticism is an opportunity both for growth and for embarrassment
  - Criticism can feel like a personal attack
  - Will review data affect annual reviews?

Addressing Hurt Feelings

- Be consistent in pointing out that:
  - Finding defects is good, not evil
  - Defect density is not correlated with developer ability
  - We want to find defects so we can honestly evaluate our own behavior and productivity
  - Reviewers are doing a good job if they find lots of defects
  - Defect density will never be used for performance evaluations
  - Negative attitudes will not be tolerated

What Review Data Should We Collect?

- Depends on what our goals are
  - Goal: Cost to find defects using reviews
    - ✓ # defects found
    - ✓ Effort (hours) to find each defect
    - ✓ Other review costs
  - Goal: # defects found by reviews that unit test won't find (should we do unit test & reviews?)
    - ✓ # defects found
    - ✓ For each defect found, indicator as to whether it would be found by unit test (author's assessment)

What is a Defect?

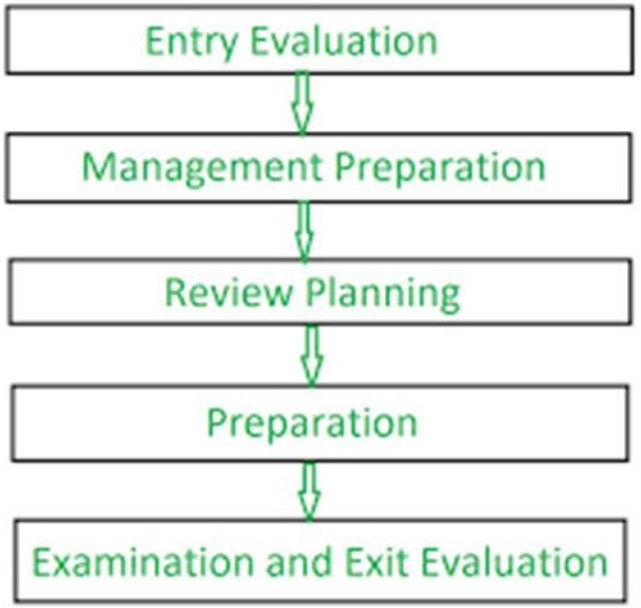
- *When a reviewer or consensus of reviewers determines that code must be changed before it is acceptable, it is a "defect"*
  - If the algorithm is wrong, it's a defect
  - If the code is right but unintelligible due to poor documentation, it's a defect
  - If the code is right but there's a better way to do it, it's a defect
- For the purposes of reviews:
  - A defect is an improvement to the code that would not have occurred without review

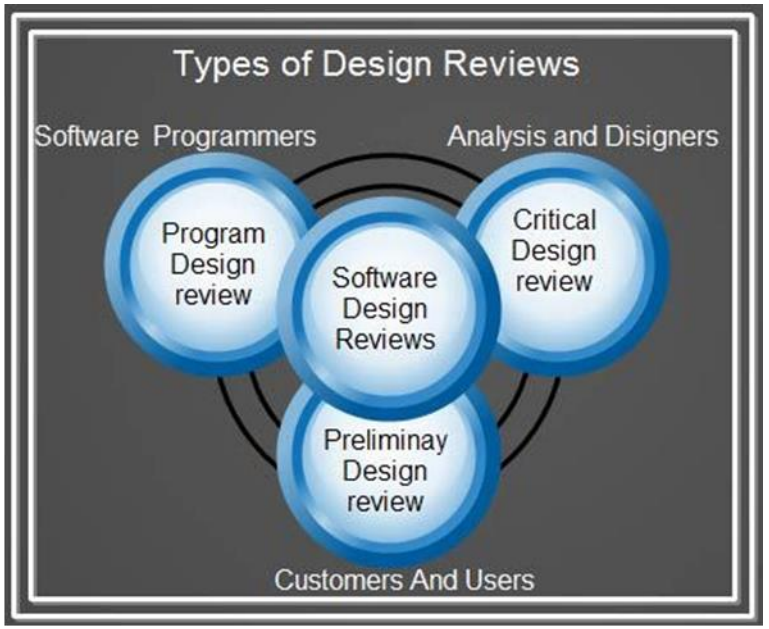
Group discussion

- Compare **defect**, **error, bug**
- (2 students – 5 minutes)

Why Are Technical Reviews Not More Widely Used?

- Significant data supporting the effectiveness of technical reviews (for the past 30 years), so why aren't technical reviews more widely used?





Why Are Technical Reviews Not More Widely Used?

- Perception that there is just one way to do reviews (inspections), and that they are not easy to do
  - Multiple types of reviews
- Viewed as an added cost
  - True, but if done correctly they will save $
- Viewed as taking too long
  - True, but if done correctly they will save time

Ron Radice. *Software Inspections. Methods & Tools. Summer, 2002*

Why Are Technical Reviews Not More Widely Used?

- Not needed with modern languages and development techniques
  - Modern developers still create defects
- "Tried it and it doesn't work"
  - Don't confuse a good idea with a poor implementation
  - The process is often not well implemented or is changed without supporting data
- Developers don't like to do them
  - They can become tedious, but they can also be fun

Ron Radice. *Software Inspections. Methods & Tools. Summer, 2002*

Why Are Technical Reviews Not More Widely Used?

- Our developers are very good
  - Even very good developers create defects
  - Most developers want to learn from their mistakes
- Developers don't take the feedback well
  - Most developers want to learn from their mistakes
- Unit testing is just as effective
  - Unit testing doesn't catch everything
  - Unit testing can give a false sense of quality

Ron Radice. *Software Inspections. Methods & Tools. Summer, 2002*

Why Are Technical Reviews Not More Widely Used?

- They are a competitive advantage!
  - No one wants to give away the secret of how to release fewer defects efficiently

Jason Cohen. Best Kept Secrets of Peer Code Reviews

Summary

- If implemented properly, reviews are a proven method for:
  - Significantly reducing the number of delivered bugs
  - Keeping code maintainable
  - Getting new hires productive quickly and safely
- Methods and tools can be misapplied, treated as a failure, and then dismissed as a bad experience by users who were not enabled for success
- Quality techniques such as reviews and testing are not mutually exclusive
- Cost, benefit and category data must be collected to verify and improve your review process!

Ron Radice. *Software Inspections. Methods & Tools. Summer, 2002*

Video link

- https://www.youtube.com/watch?v=FTN_93Px-Qc
- https://www.youtube.com/watch?v=5KB5KAak6tM

References

- Capers Jones. *Software Defect Removal Efficiency, IEEE Computer.*
- Jason Cohen, Steven Teleki, Eric Brown. *Best Kept Secrets of Peer Code Review*