Code Reviews That Work

Steven Bucksbaum, February 5, 2011

Catch 50% of Coding Errors

Code reviews can often find and remove common vulnerabilities such as format string exploits, race conditions, memory leaks and buffer overflows, and coding styles away from company set coding standards, thereby improving software security.

Capers Jones' ongoing analysis of over 12,000 software development projects showed that the latent defect discovery rate of formal inspection is in the 60-65% range. For informal inspection, the figure is less than 50%.

Lightweight code reviews or informal inspections require less-overhead than formal code inspections, and can be equally effective. Lightweight reviews are often conducted as part of the normal development process:

A Model for Periodic Lightweight Code Reviews

1. Announce to the development team that code reviews are to be conducted this week. Leave it to the developer when the review will be performed.
2. To each developer, assign him/her a different developer's code to review.
3. Expect a written report. This could in out line form.
4. Information needed is as follows:
    a. File Name and Line Number
    b. Class and Function
    c. Description of the bug or coding error found.
5. Check to make sure all code-reviewers turn in a report.
6. Check with developers to make sure changes have been based on the code review. If no changes made, state the reason for making no changes.
7. Remind the reviewer until the task is completed.
8. Next time, rotate who reviews whose code.

Report To Management:

1. Number of errors discovered.
2. Number or changes made.

Typical code review rates are about 150 lines of code per hour. Inspecting and reviewing more than a few hundred lines of code per hour for critical software (such as safety critical embedded software) may be too fast to find errors. Industry data indicate that code review can accomplish at most an 85% defect removal rate with an average rate of about 65%.

A few suggestions for the code reviews:

1. Put a summary comment at the top of your report – be positive
2. Use e-mail or an attached word document
3. Make an upfront agreement that not every question needs to be responded to.
4. Ask question rather than make statements
5. Avoid the "Why" question, tends to make people defensive.
6. Remember that there is often more than one way to approach a solution
7. Be Professional: Make sure the discussion stays focused on the code and not the coder.
8. Have good coding standards to reference.