Identifying the Fundamental Drivers of Inspection Costs and Benefits

Adam Porter
University of Maryland

Overview

- **Software inspection**
- Research questions
- Experiments
- Future work

Software Inspection

- Software inspection: *An in-process technical review of any software work product conducted for the purpose of finding and eliminating defects.* [NASA-STD-2202-93]
- Software work products: e.g., requirements specs, designs, code, test plans, documentation
- Defects: e.g., *implementation errors, failures to conform to standards, failures to satisfy requirements*

Inspection Process Model

- Most organizations use a three-step inspection process
  - individual analysis
    - use Ad Hoc or Checklist techniques to search for defects
  - team analysis
    - reader paraphrases artifact
    - issues from individual and team analyses are logged
  - rework
    - Author resolves and repairs defects

Overview

- Software inspection
- **Research questions**
- Experiments
- Future work

Current Practice

- Widely-used (especially in large-scale development)
  - Few practical alternatives
  - Demonstrated cost-effectiveness
    - defects found at all stages of development
    - high cost of rework
- Substantial inefficiencies
  - 1 code inspection per 300-350 NCSL (~ 1500 / .5MNCSL)
  - 20 person-hours per inspection (not including setup and rework)
  - significant effect on interval (calendar time to complete)
  - effort per defect is high
  - many defects go undiscovered

Research Conjectures

- Several variants have been proposed
  - [Fagan76, LMW79, PW85, BL89 , Brothers90, Johnson92, SMT92, Gilb93, KM93, Hoffman94, RD94]
- Weak empirical evaluation
  - cost-benefit analyses are simplistic or missing
  - poor understanding of cost and benefit drivers
- Low-payoff areas emphasized
  - process
  - group dynamics
- High-payoff areas de-emphasized
  - individual analysis techniques
  - tool support

Inspection Costs and Benefits

- Potential drivers
  - structure (tasks, task dependencies)
  - techniques (individual and group defect detection)
  - inputs (artifact, author, reviewers)
  - technology (tool support)
  - environment (deadlines, priorities, workloads)

Overview

- Software inspection
- Research questions
- **Experiments**
- Future work

Process Structure

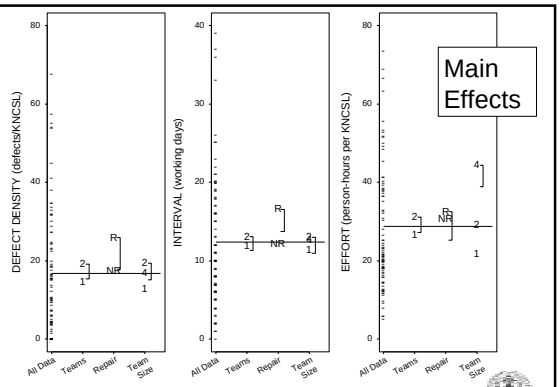
- Main structural differences
  - team size: large vs. small
  - number of teams: single vs. multiple
  - coordination of multiple teams: parallel vs. sequential
- H_0: none of these factors has any effect on effort, interval, or effectiveness
  - 6-person development team at Lucent, plus 11 outside inspectors
  - optimizing compiler (65K lines of C++)
  - Harvey Siy joined team as Inspection Quality Engineer (IQE)
  - instrumented 88 inspections over 18 months (6/94–12/95)

Experimental Design

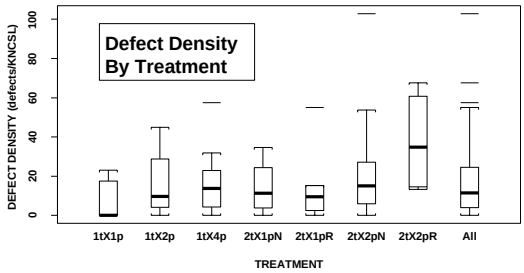
- Independent variables
  - number of inspection teams (1 or 2)
  - number of reviewers per team (1,2 or 4)
  - repair between multiple teams (required or prohibited)
- Control group: 1-team with 4-reviewers
- Dependent variables
  - inspection effort (person hours)
  - inspection interval (working days)
  - observed defect density (defects/KNCSL)
  - repair statistics

Treatment Allocation and Validity

- Treatment allocation rule
  - IQE notified via email when code unit becomes available
  - treatment assigned on a random basis
  - reviewers selected at random (without replacement)
- Internal validity
  - selection (natural ability)
  - maturation (learning)
  - instrumentation (code quality)
- External validity
  - scale (project size)
  - subject representativeness (experience)
  - team/project representativeness (application domain)



- Effectiveness: no significant effects

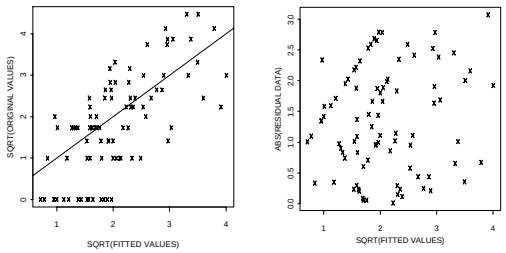


- Team size: 1tX1p < (1tX2p ≡ 1tX4p)
- Repair: 2tXR ≡ 2tXN
- Teams: 2t X1p > 1tX1p, 2tX2p ≡ 1tX2p
- Teams: 2t ≡ 1t (total # of rev's held constant)

Process Inputs

- Independent vars insignificant, but variation is high
  - are the effects of unknown factors obscuring the effects of process structure?
  - are the effects of unknown factors greater than the effect of process structure?
- Process inputs are likely source of variation
- Develop statistical models
  - generalized linear models (Poisson family with logarithmic link)
  - model variables reflect process structure and process inputs
  - remove insignificant factors

Defect Density



- Model: *Defects ~ Functionality +* log*(Size) +* $R_B + R_F$
  - explains ≈ 50% of variation using 10 of 88 degrees of freedom
- Process input is more influential than process structure
  - structure: ≈ 2%, inputs: ≈ 50%

Summary

- Structural factors had no significant effect on effectiveness
  - more reviewers didn't always find more defects
- Process inputs were far more influential than process structure
- Best explanation of inspection effectiveness (so far)
  - not process structure
  - reviewer expertise

Analysis Techniques: Groups vs. Individuals

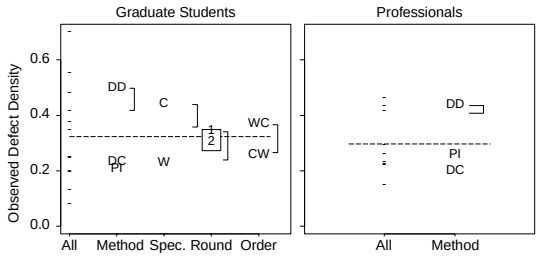
- Traditional view: meetings are essential
  - many defects or classes of defects are found during meetings
  - these defects would not have been found otherwise
- Research hypotheses:
  - inspections with meetings are no more effective than those without
  - inspections with meetings do not find specific classes of faults more often than those without
  - benefit of additional individual analysis is greater than or equal to the benefit of meeting

Candidate Inspection Methods

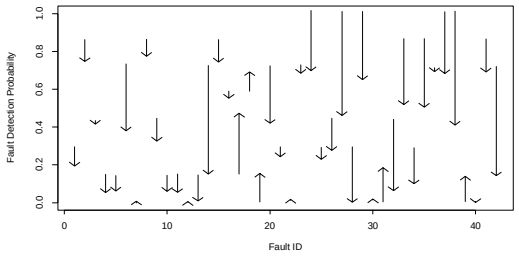
- Preparation -- Inspection (PI)
  - individuals become familiar with artifact
  - team meets to identify defects
- Detection -- Collection (DC)
  - individuals identify issues
  - team meets to classify issues and identify defects
- Detection -- Detection (DD)
  - individuals identify issues
  - individuals identify more issues

Experimental Design

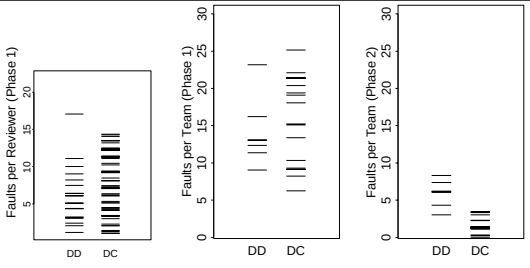
- Subjects:
  - 21 UMD CS graduate students (Spring '95)
  - 27 professional software developers (Fall '96)
- Artifacts
  - software requirements specs (WLMS and CRUISE)
- Independent Variables
  - inspection method (PI, DC, or DD)
  - inspection round (R1 or R2)
  - specification to be inspected (W or C)
  - presentation order (WC or CW)
- Dependent Variables
  - individual and team defect detection ratios
  - meeting gain and loss rates



- H_1: Inspections with meetings find more defects than those without
  - DD method found more faults than any other method
  - PI method was indistinguishable from DC method



- H_2: Inspections with meetings find specific classes of defects more often than those without
  - 5 of 42 defects are found more often by inspections with meetings than by those without
  - only 1 difference is statistically significant



- H_3: Benefit of additional individual analysis is less than or equal to the benefit of meeting
  - no differences in 1st phase team performance
  - significant differences in 2nd phase team performance

Summary

- Meetingless inspections identified the most defects
  - also, generated the most issues and false positives
- Few "meeting-sensitive" faults
- Additional data
  - similar study at the University of Hawaii shows same results (Johnson97, Porter and Johnson97)
  - industrial case study of 3000 inspections showed that meetingless inspections were as effective as those with meetings (Perpich, Perry, Porter, Votta, and Wade97)
- Best explanation of inspection effectiveness (so far)
  - not process structure nor group dynamics
  - reviewer expertise

Improved Individual Analysis

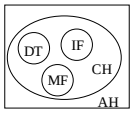
- Develop an improved individual analysis
- Measure effect on overall inspection effectiveness
- Classification of individual analysis methods
  - analysis techniques: strategies for detecting defects
    - prescriptiveness: nonsystematic - systematic
  - reviewer responsibility: population of defects to be found
    - scope: specific - general
  - coordination policy: assignment of responsibilities to reviewers
    - overlap: distinct - identical

Systematic Inspection Hypothesis

- Current Practice: Ad Hoc or Checklist methods
  - nonsystematic techniques with general and identical responsibilities
- Alternative approach
  - systematic techniques with specific and distinct responsibilities
- Research Hypothesis
  - H_0: Inspections using non-systematic techniques with general and identical responsibilities find more defects than those using systematic techniques with specific and distinct responsibilities

Defect-based Scenarios

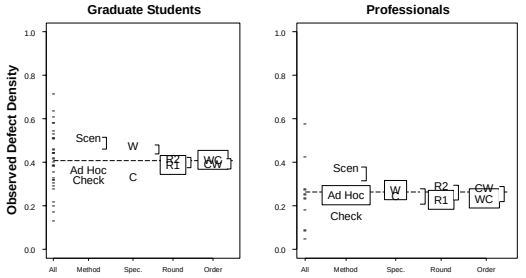
- Ad Hoc method based on defect taxonomy [BW]
- Checklist method based on taxonomy plus items taken from industrial checklists.
- Scenario method refined Checklist items into procedures for detecting a specific class of defects
- Three groups of scenarios
  - data type inconsistencies
  - incorrect functionality
  - ambiguity/missing functionality



Reviewer Responsibility

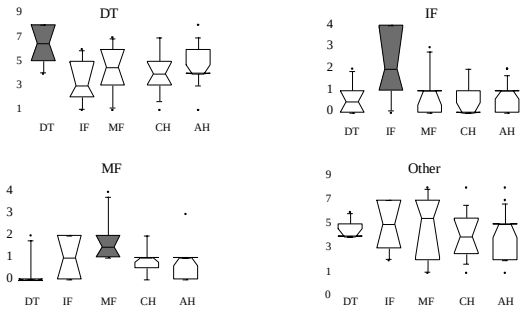
Experimental Design

- Subjects
  - 48 UMD CS graduate students (Spring and Fall '93)
  - 21 professional software developers (Fall '95)
- Software requirements specs (WLMS and CRUISE)
- Independent variables
  - replication (E1, E2)
  - round (R1, R2)
  - analysis method (Ad Hoc, Checklist, or Scenario)
  - specification (W or C)
  - order (CW, WC)
- Dependent variables
  - individual & team defect detection rates
  - meeting gain & loss rates



- Scenarios outperform all methods
- Checklist performance no better than Ad Hoc

Individual Inspection Performance: WLMS



- Scenario reviewers found more targeted detects
- Scenarios reviewers found as many untargeted defects

Summary

- Current models may be unfounded
  - meetings not necessarily cost-effective
  - more complex structures did not improve effectiveness
- Reviewer expertise appears to be dominant factor in inspection effectiveness
  - structure had little effect
  - inputs more influential than structure
  - individual effects more influential than group effects
  - improved individual analysis methods significantly improved performance

Field Testing

- Goal: reduce interval without reducing effectiveness
- Solution approach: remove coordination
  - private vs. shared individual analysis
  - meetings vs. meetingless
  - sequential vs. parallel tasks
- Developed web-based inspection tool (HyperCode)
  - Event monitor for distributed development groups
- Have deployed the tool
  - Naperville, IL and Whippany, NJ
  - multi-phase experiment