Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

How Good Is the Software: A Review of Defect Prediction Techniques

Brad Clark
Dave Zubrow

Sponsored by the U.S. Department of Defense

Objectives

Awareness of defect prediction and estimation techniques

Awareness of the value to project management and process improvement activities of analyzing defect data

Recommendations for getting started

Why Analyze and Predict Defects?

Project Management

- Assess project progress
- Plan defect detection activities

Work Product Assessment

- Decide work product quality

Process Management

- Assess process performance
- Improve capability

Outline

F Background
- Definitions
- Measuring Defects
  - Tools and techniques
  - Attributes

Technique Review
- Project Management
- Work Product Assessment
- Process Improvement

Definition - Software Defect

Software Defect: any flaw or imperfection in a software work product or software process

- software work product is any artifact created as part of the software process
- software process is a set of activities, methods, practices, and transformations that people use to develop and maintain software work products

A defect is frequently referred to as a fault or bug

Focus on Predicting those Defects that affect Project and Product Performance

Defects as the Focus of Prediction

Distinguish between major and minor defects
- do not use minor or documentation defects in predictions
- minor defects will inflate estimate of latent product defects

Most defect prediction techniques used in planning rely on historical data

Defect prediction techniques vary in the types of data they require
- some require little data, others require more
- some use work product characteristics, others require defect data only

Techniques have strengths and weaknesses depending on the quality of the inputs used for prediction

Defect Attributes Available for Analysis

- Problem Status
  - Open
    - Recognized
    - Evaluated
    - Resolved
  - Closed
- Problem Type
  - Software Defect
    - Requirements Defect
    - Design Defect
    - Code Defect
    - Operational Doc. Defect
    - Test Case Defect
    - Other Product Defect
- Problem Type (con't)
  - Other Problems
    - Hardware Problem
    - Operating System Problem
    - User Mistake
    - Operations Mistake
    - New Req't / Enhancement
  - Undetermined
    - Not repeatable
    - Value not identified
- Uniqueness
  - Original
  - Duplicate
- Criticality Level
- Urgency

Additional Attributes to Consider

Recognition

- What is the problem?
- When was the problem reported?
- Who reported the problem?

Evaluation

- What work product caused the problem?
- What activity discovered the problem?
- What activity introduced the problem?

Resolution

- What work needs to be done?
- What work products will be affected by the change?
- What are the prerequisite changes?

Closure

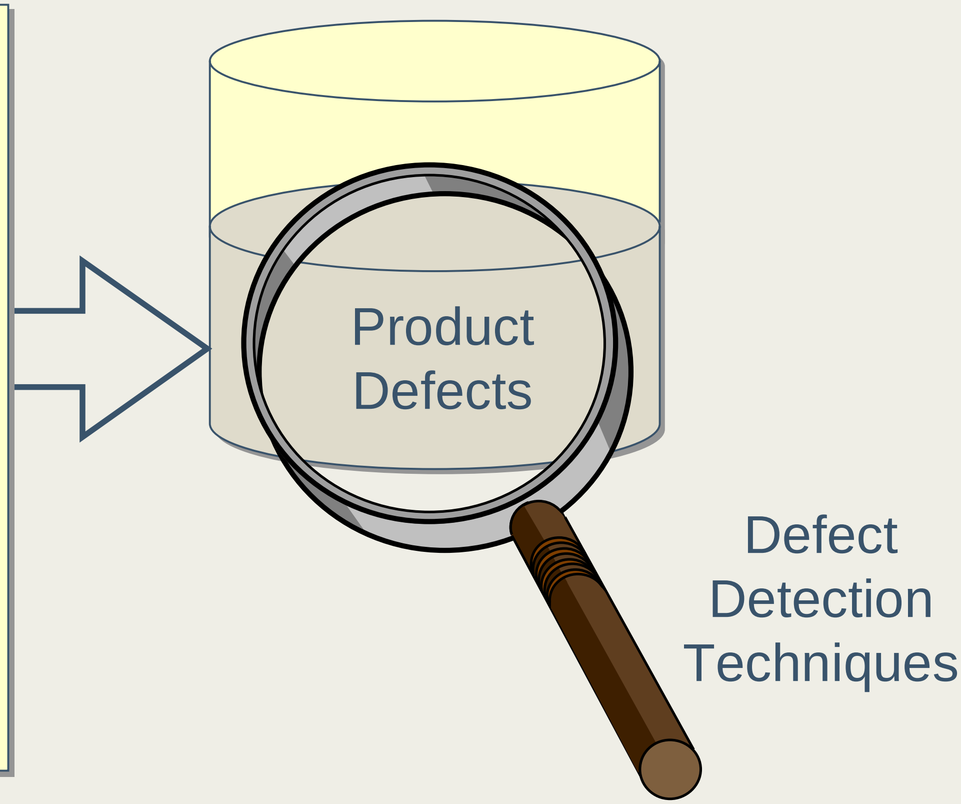
- When are the changes expected?
- What configuration contains the changes?

Project and Process Factors Correlated with Defect Insertion

Requirements adequacy
Application Size
Application Complexity
COTS and Reused Code
Development Team Capability
- Problem Solving
- Designing
- Coding

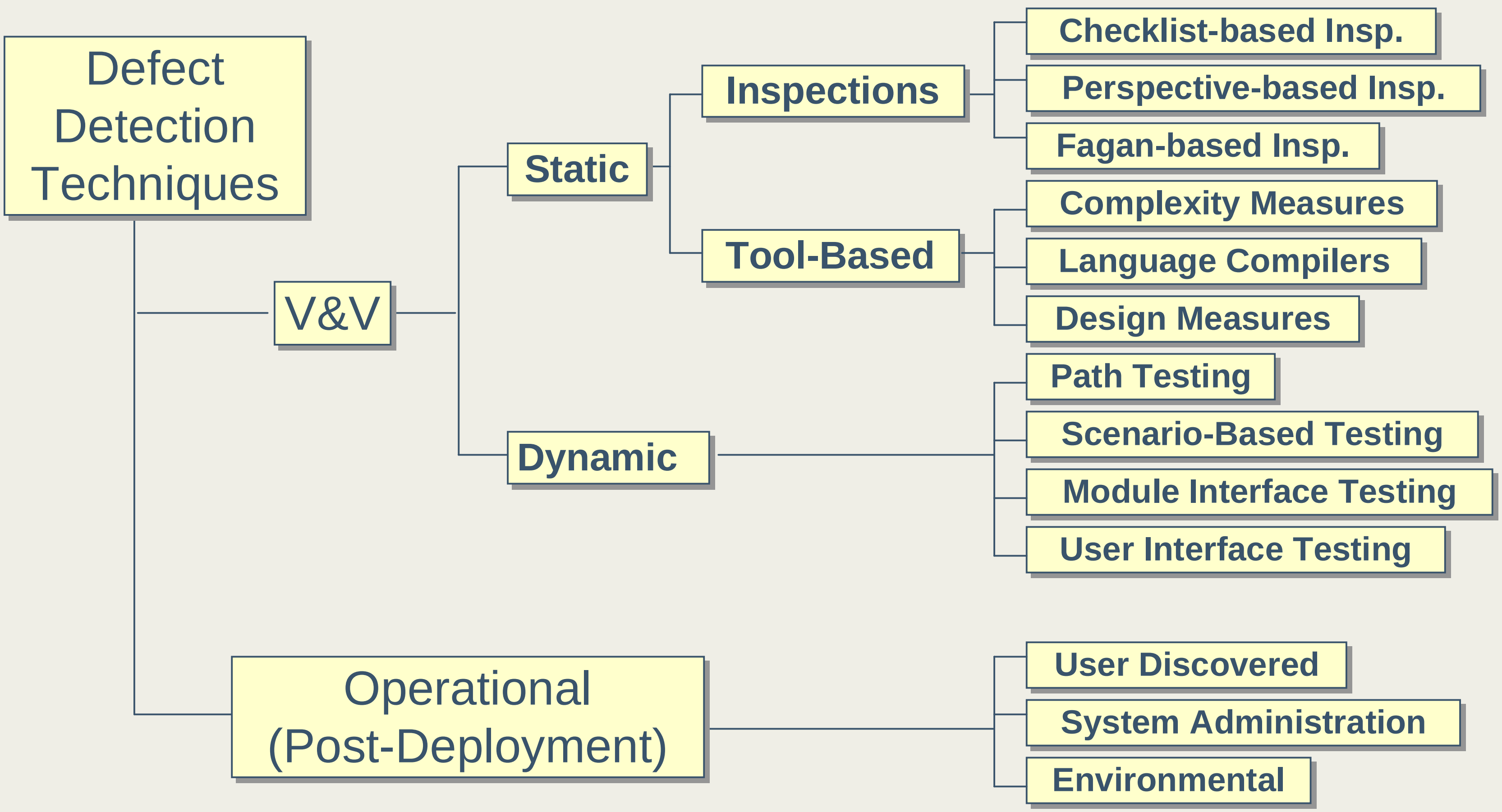
Development Team Experience
- Application Domain
- Language & Tools
- Platform

Process Maturity



Defect Discovery Sources

(how are the data generated)



Carnegie Mellon
Software Engineering Institute

Outline

Background
- Definitions
- Measuring Defects
  - Tools and techniques
  - Attributes

F Technique Review
- Project Management
- Work Product Assessment
- Process Improvement

Defect Prediction Techniques

Project Management

- Empirical Defect Prediction
- Defect Discovery Profile
- COQUALMO
- Orthogonal Defect Classification

Work Product Assessment

- Fault Proneness Evaluation (Size, Complexity, Prior History)
- Capture/Recapture Analysis

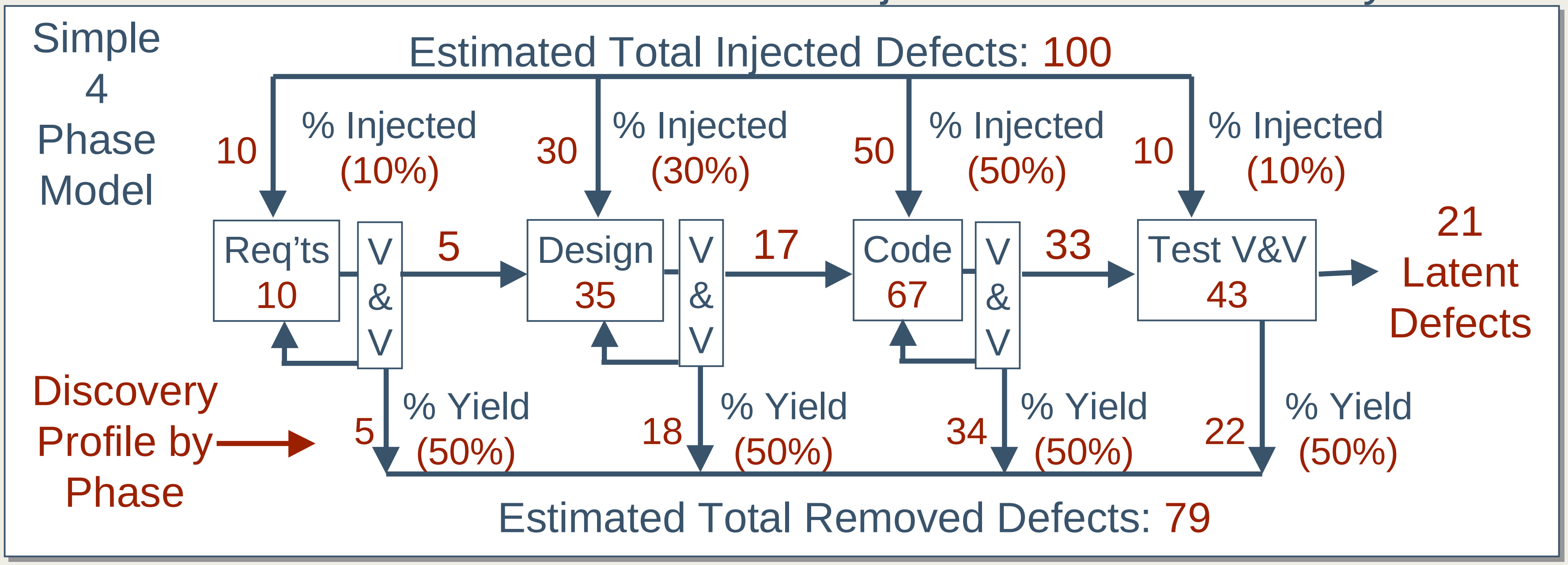
Process Improvement

- Defect Prevention Program
- Statistical Process Control

Empirical Defect Prediction Technique Review

Description - number of Defects per Size (Defect Density)

- defect density (Number of Defects / Thousands Lines of Code) based on historical data
- enhanced with historical data on injection distribution and yield



Empirical Defect Prediction - 2

When to use - use for planning (total defects) and in-process monitoring of defect discovery numbers (latent defects)

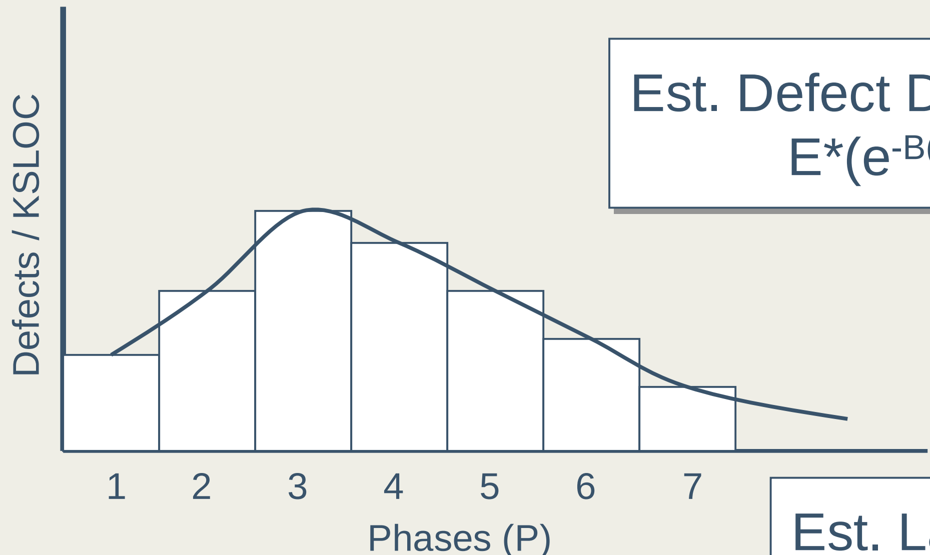
Required Data - historical defect density data required for planning; in-process data required for monitoring

Strengths - easy to use and understand; can be implemented with minimal data

Weaknesses - requires stable processes and standardized life cycle; does not account for changes in the project, personnel, platform, or project

Defect Discovery Profile Technique Review

Description - projection, based on time or phases, of defect density (or number of defects) found "in-process" onto a theoretical discovery curve (Rayleigh). Found in the SWEEP and STEER models



Est. Defect Density in Phase =
$E*(e^{-B(P-1)^2} - e^{-BP^2})$

E = total Defects/KSLOC
B = efficiency of discovery process
P = phase number

Est. Latent Defect Density Remaining =
$E * e^{-(BP^2)}$

Defect Discovery Profile - 2

When to use - as early in the life cycle as defect data collection permits

Required Data - historical defect density data, estimated and actual size, and consistently tracked defect counts

Strengths - predicts defect density by time period enabling the estimation of defects to be found in test

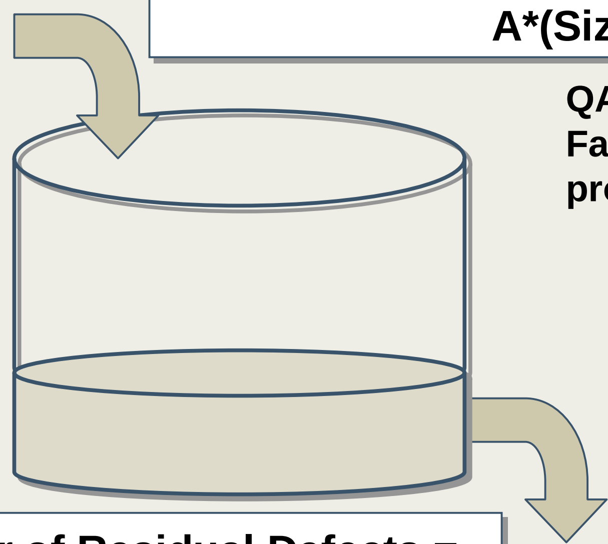
Weaknesses - no insight or adjustment mechanism for B to account for changes in the product, personnel, platform, or project will impact defect predictions.

COQUALMO Technique Review

Description - a defect prediction model for the requirements, design, and coding phases based on sources of introduction and discovery techniques used

Est. Number of Defects Introduced by Phase =
$A*(Size)^{B}*QAF$

<u>Phases</u>
Requirements Defects
Design Defects
Code Defects



QAF are 21 Quality Adjustment Factors characterizing the people, product, platform and project.

Detection and Removal (DRF) Factors are Automated Analysis, People Reviews and Execution Testing / Tools

Est. Number of Residual Defects =
C_j* (Defects Introduced)$_j$ * $\prod$ (1-DRF$_j$)

COQUALMO - 2

When to use – used in the planning phase of a project

Required Data - size of the product and ratings for 21 Quality Adjustment Factors

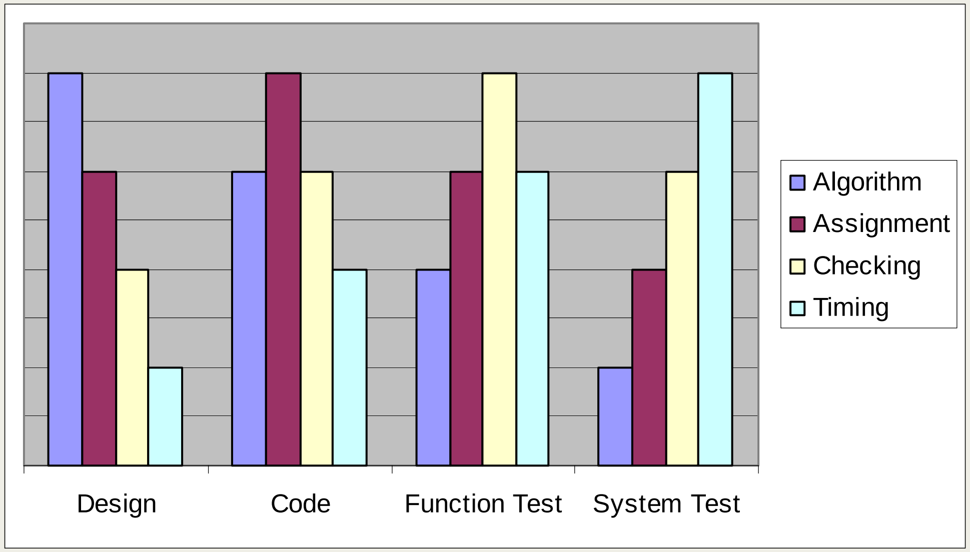
Strengths - predicts defects for three phases; quantifies the effect of different discovery techniques on the detection and removal of defects. Considers the effects of attributes such as product, personnel, project, and platform

Weaknesses - covers a small number of phases; does not predict test or post-deployment defects

Orthogonal Defect Classification Technique Review

Description – classification and analysis of defects to identify project status based on comparison of current defects with historical patterns; identify areas for process improvement based on analysis of defect types, "triggers," impact, and source

- Types are what was required for the fix, not the cause of the defect (e.g. function, assignment, interface)
- Triggers are catalysts that cause defects to surface (e.g. testing, inspection, conditions of operational use)



Orthogonal Defect Classification - 2

When to use – ongoing throughout project

Required Data – orthogonal defect classification scheme mapped to development process; historical defect profiles

Strengths – classifications linked to process provide valuable insight; classification takes little time

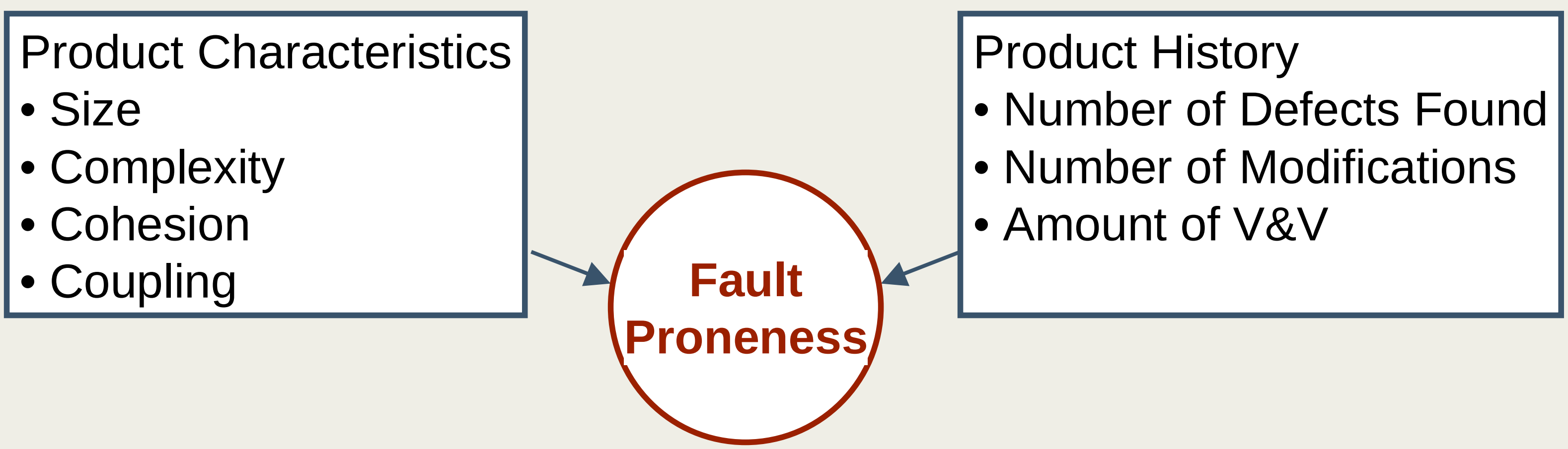
Weaknesses – requires development of classification scheme; reliable classification of defects; ongoing data collection and analysis; does not account for changes in people, process, or product

Fault Proneness Technique Review

Description – analysis of work product attributes to plan for allocation of defect detection resources (inspection and testing)

A variety of models and heuristics

- comparing cyclomatic complexity against a threshold
- various parametric models (e.g., discriminant analysis)
- reviewing module or component defect histories



Fault Proneness - 2

When to use – test planning, during coding and testing,

Required Data – size, complexity, coupling, historical defect data, etc.

Strengths – efficient and effective focus of defect detection activities

Weaknesses – “in-process” fault density by module or component may not predict operational fault density, effort may be misdirected; models and assumptions not likely to hold from one system to the next

Capture Recapture Technique Review

Description – analysis of pattern of defects detected within an artifact by independent defect detection activities (inspectors or inspection versus test)

Number of remaining defects is estimated from the overlap in defects identified independently by individual inspectors according to the following formula:

N (estimated) in work product = n(inspector 1) * n(inspector 2) / m (number defects found by both inspectors)

N(estimated) – N (unique discovered) = Remaining defects (est.)

Capture Recapture - 2

When to use – determining whether a work product should undergo re-inspection

Required Data – detailed defect descriptions from each inspector

Strengths – can be used as soon as data are available

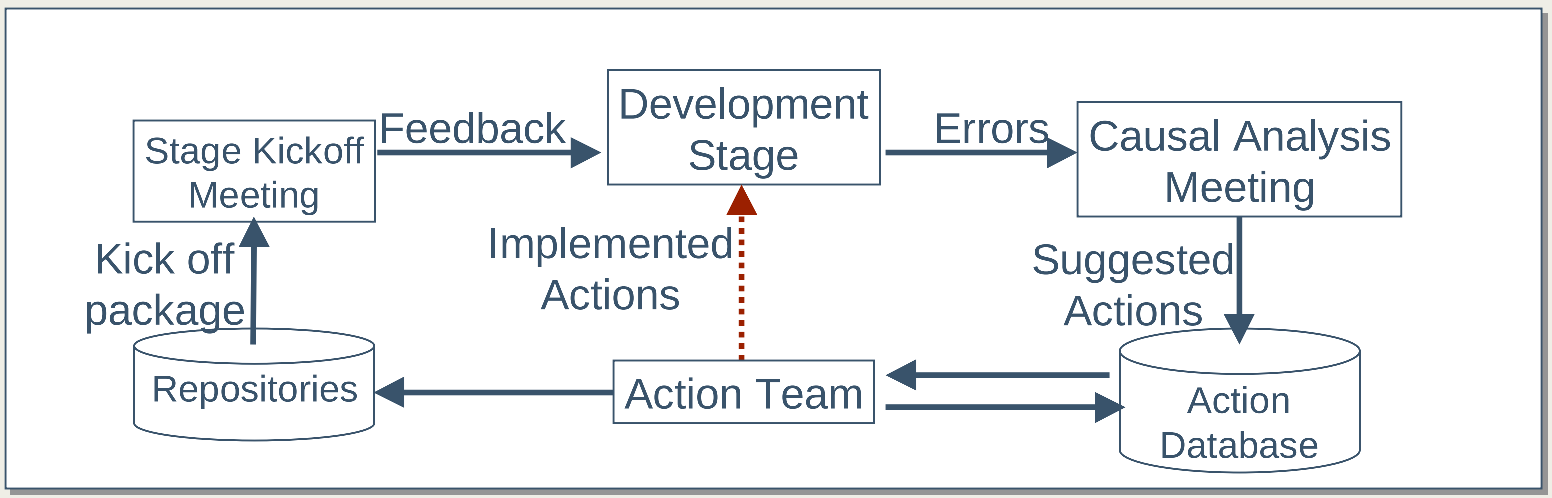
Weaknesses – estimates of number of remaining defects best when stringent assumptions are met. Relaxing assumptions requires more complicated estimation. More robust when simply used to predict whether a criterion for re-inspection has been exceeded

Defect Prevention Technique Review

Description – root cause analysis of most frequently occurring defects

Sample of defect reports selected for in-depth causal analysis

Actions taken to make process changes or improve training to eliminate the root cause of defects and prevent their recurrence



Defect Prevention- 2

When to use – prior to launching new projects or beginning new phases of a project

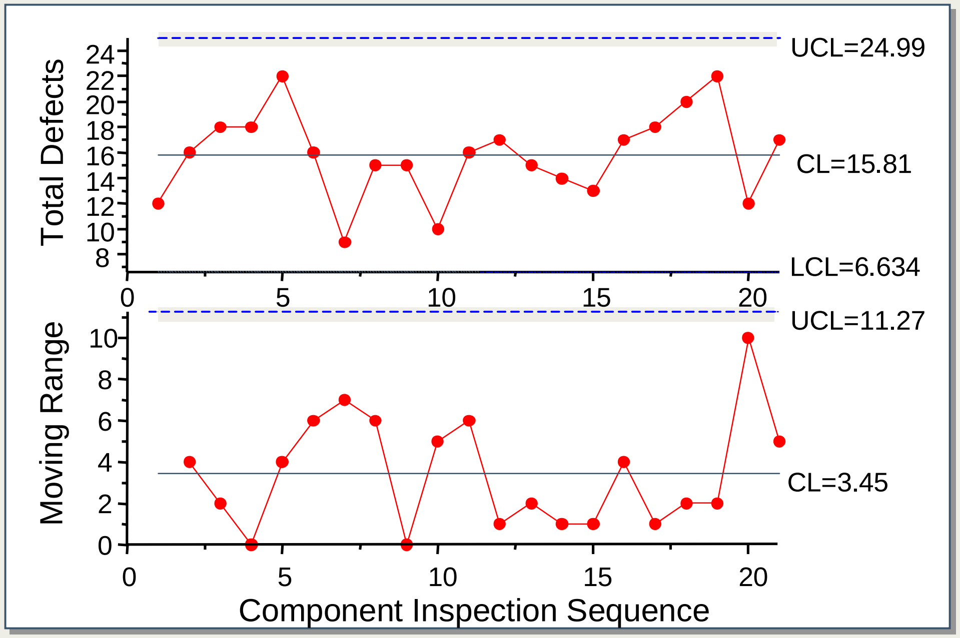
Required Data – historical defect data

Strengths – allows for comparison of defect trends over time to assess impact and ROI for defect prevention activities

Weaknesses – requires sampling of defects and in-depth analysis and participation by engineers to identify root causes

Statistical Process Control Technique Review

Description – use of control charts to determine whether inspection performance was consistent with prior process performance. Process capability depicts expected range of performance in terms of selected attribute.



Statistical Process Control - 2

When to use – when inspections are being conducted

Required Data – current measures of inspection process (e.g., defects found, prep time, review rate); historical measures to develop control chart

Strengths – gives indication of inspection and development process performance. Signals provide rapid feedback suggesting re-inspection or process anomaly.

Weaknesses – requires stable process and “real time” data collection and analysis

Observations

For Project Management

- models predict total defects in a product and latent defects from “in-process” measures
- models use estimated and actual software size as a parameter
- models use additional factors to adjust defect estimates

For Product Quality

- predictions can be developed from inspection or product characteristics data

For Process Improvement

- expected process behavior can be used to gauge performance and identify opportunities for improvement

Observations -2

Prediction models are useful for planning and establishing expectations.

Tracking against expectations

- when deviations occur - some action is taken such as reallocation of resources towards defect detection, specific modules, or re-inspection.
- most analysis techniques are not very explicit on the threshold that triggers investigation. The exception is control limits in SPC.

Estimates are often inaccurate but suggestive and value-added for decision making and planning

Recommendations for Getting Started

Get started even with simple techniques
- the data available will help determine the technique
- availability of historical data will drive model selection
- analyze for patterns across defects, don’t just fix the defects

Measure product defects early in the life cycle
- Post-release defect tracking is the least helpful
- Pre-release defect tracking by phase and by type is most helpful but also more burdensome

Defect definition should meet the intended use of the data
- Track project progress
- Determine product quality
- Assess process performance

Changes in the product, personnel, platform, or project must be measured and accounted for in predictions

Call for Collaboration

If you are interested in using these techniques or studying their effectiveness, please contact:

Dave Zubrow
Software Engineering Measurement and Analysis
4500 Fifth Ave
Pittsburgh, Pa 15213

412-268-5243
dz@sei.cmu.edu

I look forward to hearing from you.

Resources and References –1

General References

Fenton, N. and Neil, M. “ A critique of software defect prediction models,” IEEE Transactions on Software Engineering, May/June 1999.

Frederick, M. “Using defect tracking and analysis to improve software quality,” University of Maryland, June 1999.

Florac, W. A. Software Quality Measurement: A Framework for Counting Problems and Defects (CMU/SEI-92-TR-22). Pittsburgh Pa.: Software Engineering Institute, Carnegie Mellon University, September 1992. Available online:
www.sei.cmu.edu/publications/documents/92.reports/92.tr.022.html

Peng, W.W and Wallace, D.R. “Software error analysis,” NIST, Special Publication 500-200, 1993.

Empirical Defect Prediction

Humphrey, W. Introduction to the Team Software Process. Reading, MA: Addison Wesley, 2000.

Weller, E.F. “Managing software projects using inspection data,” IEEE Software, 1994.

Defect Profile Prediction Technique

Gaffney, J., Roberts, W., and DiLorio, R. A Process and Tool for Improved Software Defect Analysis and Quality Management. Software Technology Conference Proceedings, May 1997.

COQUALMO Prediction Technique

Chulani, S. Modeling Software Defect Introduction and Removal: COQUALMO. University of Southern California Center for Software Engineering Technical Report USC-CSE-99-510, 1999.

Resources and References - 2

Orthogonal Defect Classification Defect Prediction Technique

Chillarege, R., Bhandari, I., Chaar, H., Halliday, D Ray, B. and Wong, M. Orthogonal Defect Classification - A Concept for In-Process Measurements. IEEE Transactions on Software Engineering, Vol 18, No 11, Nov 1992

Bridge, N., Miller, C. Orthogonal Defect Classification: Using defect data to improve software development. http://www.chillarege.com.

El Emam, K, Wieczorek, I. The repeatability of code defect classifications. IEEE: Proceedings of the Ninth International Symposium on Software Reliability Engineering, 1998.

Fault Proneness

Selby, R. and Basili, V. Analyzing error-prone system structure. IEEE Transactions on Software Engineering, vol 17, no. 2, Feb 1991.

Briand, L.C., Melo, W.L., and Wust, J. “Assessing the applicability of falut-proneness models across object-oriented software projects,” ISERN Report No. ISERN-00-06, 2000.

El Emam, K. A Primer on Object Oriented Measurement. National Research Council of Canada, 2000.

Fenton, N.E. and Ohlsson, N. “Quantitative analysis of faults and failures in a complex software system,” IEEE Transactions on Software Engineering, vol 26, no 8, August 2000, 797-814.

Ohlsson, M.C. and Wohlin, C. Identification of green, yellow, and red legacy components. Lund University, Sweden.

Resources and References -3

Capture/Recapture Analysis

Briand, L. and Freimut, B.G. “A comprehensive evaluation of capture-recapture models for estimating software defect content,” IEEE Transactions on Software Engineering, vol 26., No. 6, June 2000, 518-540.

Humphrey, W. Introduction to the Team Software Process. Reading, MA: Addison Wesley, 2000.

Petersson, H. and Wohlin, C. “An empirical study of experience-based software defect content estimation methods” Lund University.

Defect Prevention Program

Mays, R.G., Jone, C.L., Holloway, G.J., and Sudinski, D.P., “Experiences with defect prevention,” IBM Systems Journal, vol. 29, no. 1, 1990, 4-32.

Grady, R.B., “Software failure analysis for high-return process improvement decisions,” Hewlett Packard Journal, August, 1996.

Gale, J.L., Tirso, J.R., and Burchfield, C.A., “Implement the defect prevention process in the MVS interactive programming organization,” IBM Systems Journal, vol. 29, no. 1, 1990, 33-43.

Statistical Process Control

Florac, W.A. and Carleton, A.D. Measuring the Software Process: Statistical process control for software process improvement. Reading, MA : Addison Wesley, 1999.