Review Techniques - Cost impact of Software Defects, Code review and statistical analysis.

[Book 2 - Chapter 20]

- Software reviews are a “filter” for the software process.
- Reviews are applied at various points during software engineering and serve to uncover errors and defects that can then be removed.
- Software reviews “purify” software engineering work products, including requirements and design models, code, and testing data.
- A review—any review—is a way of using the diversity of a group of people to:

    1. Point out needed improvements in the product of a single person or team;

    2. Confirm those parts of a product in which improvement is either not desired or not needed;

- Achieve technical work of more uniform, or at least more predictable, quality than can be achieved without reviews, in order to make technical work more manageable.

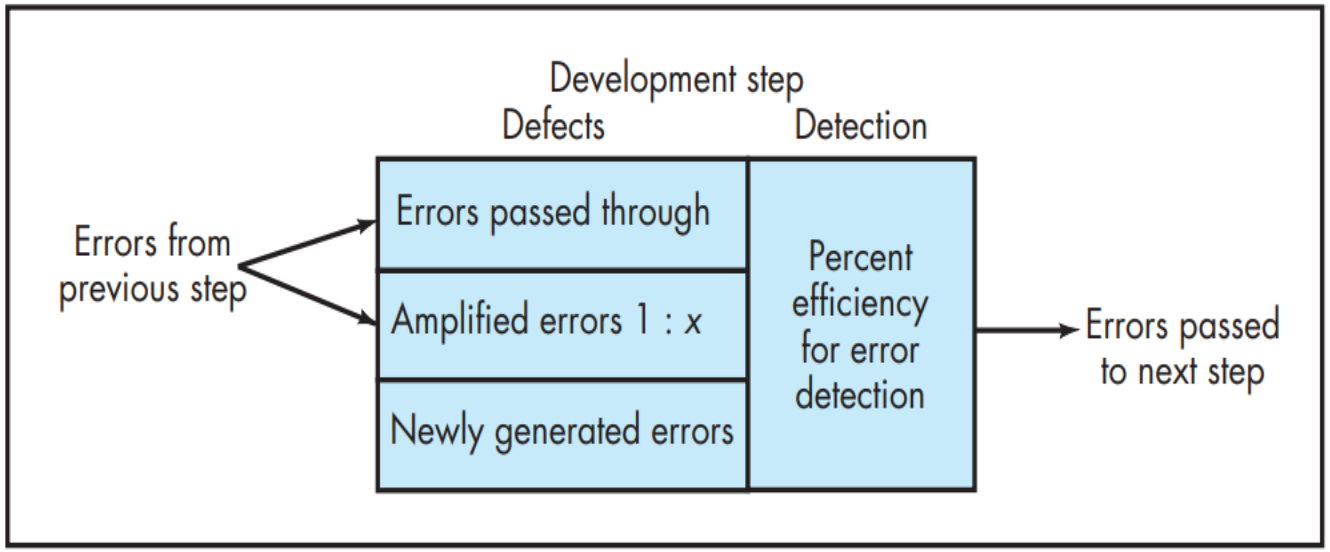
COST IMPACT OF SOFTWARE DEFECTS

- Within the context of the software process, the terms defect and fault are synonymous. Both imply a quality problem that is discovered after the software has been released to end users (or to another framework activity in the software process).
- The primary objective of technical reviews is to find errors during the process so that they do not become defects after release of the software.
- The obvious benefit of technical reviews is the early discovery of errors so that they do not propagate to the next step in the software process.
- Review techniques have been shown to be up to 75 percent effective in uncovering design flaws.
- By detecting and removing a large percentage of these errors, the review process substantially reduces the cost of subsequent activities in the software process.

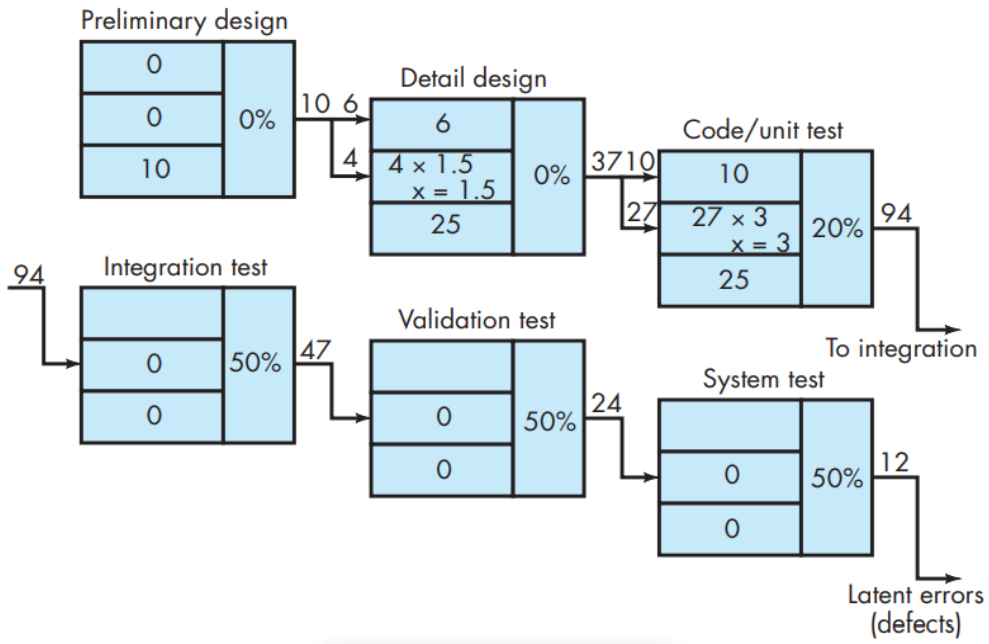
DEFECT AMPLIFICATION AND REMOVAL

- A defect amplification model can be used to illustrate the generation and detection of errors during the design and code generation actions of a software process.

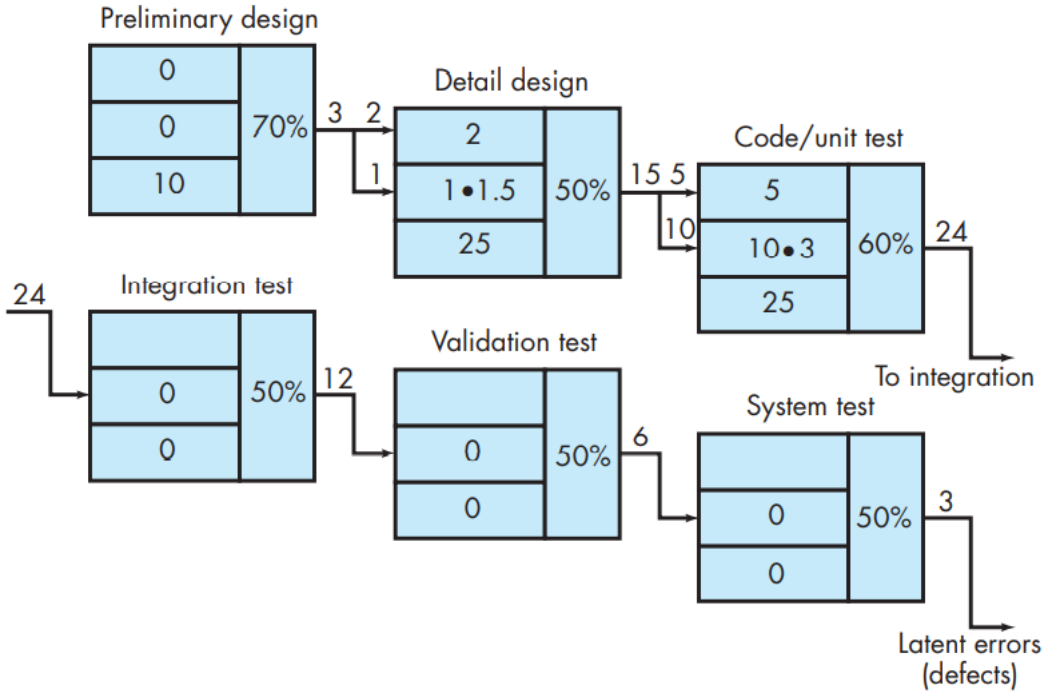
To conduct reviews, you must expend time and effort, and your development organization must spend money



Defect amplification— no reviews



Defect amplification— reviews conducted



REVIEW METRICS AND THEIR USE

- Technical reviews are one of many actions that are required as part of good software engineering practice.
- Each action requires dedicated human effort

Review metrics:

- **Preparation effort, Ep**—the effort (in person-hours) required to review a work product prior to the actual review meeting
- **Assessment effort, Ea**— the effort (in person-hours) that is expended during the actual review
- **Rework effort, Er** — the effort (in person-hours) that is dedicated to the correction of those errors uncovered during the review
- **Work product size, WPS**—a measure of the size of the work product that has been reviewed (e.g., the number of UML models, or the number of document pages, or the number of lines of code)
- **Minor errors found, Errminor**—the number of errors found that can be categorized as minor (requiring less than some prespecified effort to correct)
- **Major errors found, Errmajor**—the number of errors found that can be categorized as major (requiring more than some prespecifi ed effort to correct

Analyzing Metrics

- The total review effort and the total number of errors discovered are defi ned as:

  Ereview = Ep + Ea + Er

  Errtot = Errminor + Errmajor

- Error density represents the errors found per unit of work product reviewed.

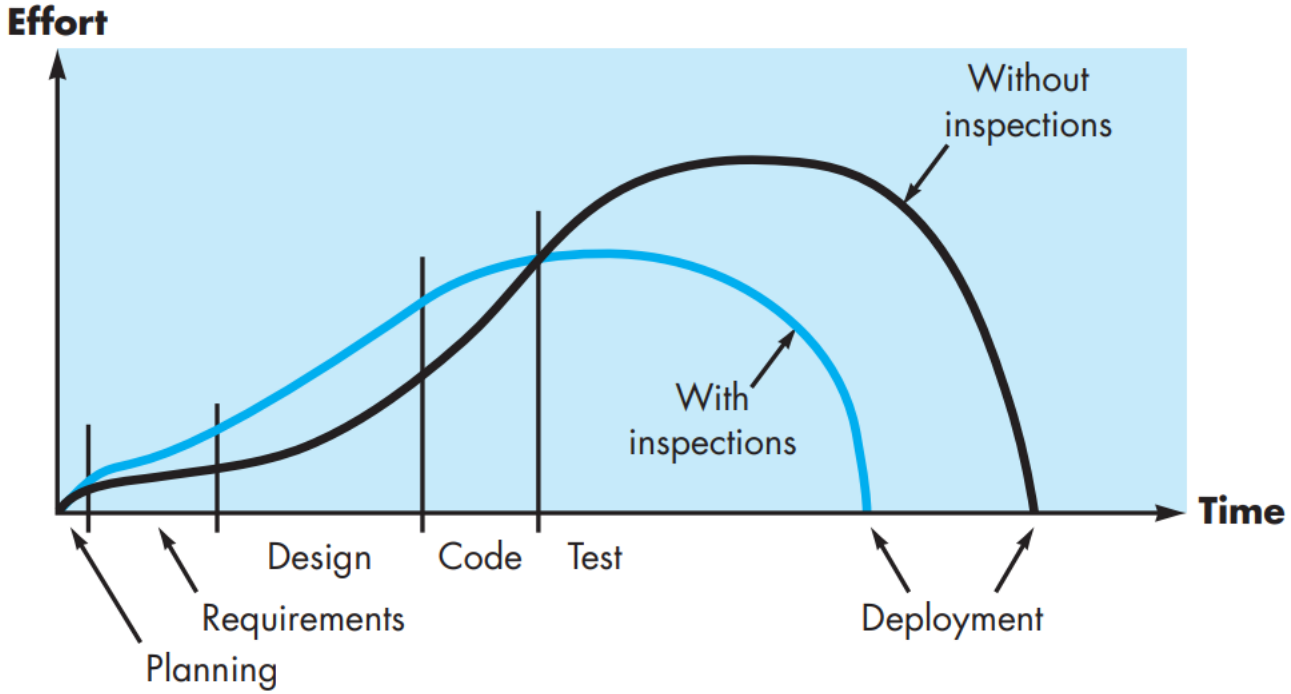
  Error density = Errtot/ WPS

Cost-Effectiveness of Reviews

- It is difficult to measure the cost-effectiveness of any technical review in real time.
- A software engineering organization can assess the effectiveness of reviews and their cost benefit only after reviews have been completed, review metrics have been collected, average data have been computed, and then the downstream quality of the software is measured
- Effort saved per error = Etesting - Ereviews

Effort expended with and without reviews

- Effort expended when reviews are used does increase early in the development of a software increment
- testing and corrective effort is reduced.
- The deployment date for development with reviews is sooner than the deployment date without reviews.
- Reviews don’t take time, they save it



REVIEWS : A FORMALITY SPECTRUM

- Technical reviews should be applied with a level of formality that is appropriate for the product to be built, the project time line, and the people who are doing the work.
- The formality of a review increases when

  (1) Distinct roles are explicitly defined for the reviewers,

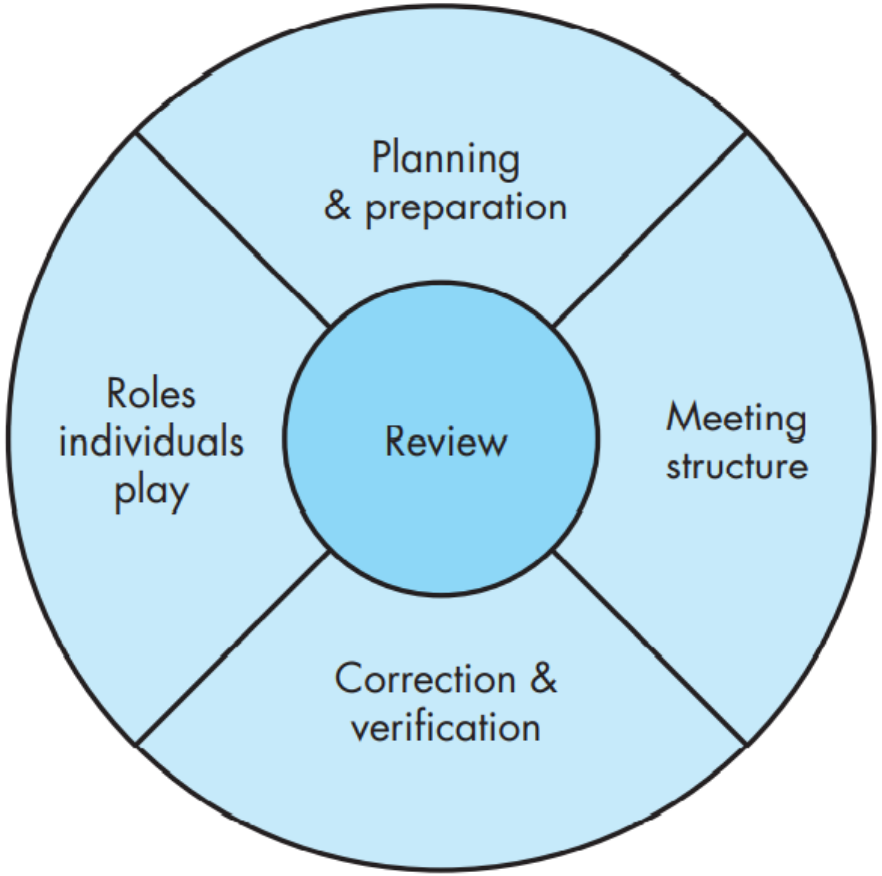
  (2) There is a sufficient amount of planning and preparation for the review,

  (3) A distinct structure for the review (including tasks and internal work products) is defined,

A set of specific tasks would be conducted based on an agenda that was developed before the review occurred.

The results of the review would be formally recorded, and the team would decide on the status of the work product based on the outcome of the review.

Members of the review team might also verify that the corrections made were done properly



Technical reviews: informal reviews and more formal technical reviews.

INFORMAL REVIEWS

- Informal reviews include a simple desk check of a software engineering work product with a colleague, a casual meeting (involving more than two people) for the purpose of reviewing a work product, or the review-oriented aspects of pair programming
- One way to improve the efficacy of a desk check review is to develop a set of simple review checklists for each major work product produced by the software team.
- The questions posed within the checklist are generic, but they will serve to guide the reviewers as they check the work product

Checklist for interfaces:

- Is the layout designed using standard conventions? Left to right? Top to bottom?
- Does the presentation need to be scrolled?
- Are color and placement, typeface, and size used effectively?
- Are all navigation options or functions represented at the same level of abstraction?
- Are all navigation choices clearly labeled?

❑ Any errors or issues noted by the reviewers are recorded by the designer for resolution at a later time.

- Pair programming can be characterized as a continuous desk check. Rather than scheduling a review at some point in time
- Pair programming encourages continuous review as a work product (design or code) is created.
- The benefit is immediate discovery of errors and better work product quality as a consequence.

FORMAL TECHNICAL REVIEWS

- A formal technical review (FTR) is a software quality control activity performed by software engineers (and others).
- The objectives of an FTR are:

  (1) to uncover errors in function, logic, or implementation for any representation of the software;

  (2) to verify that the software under review meets its requirements;

  (3) to ensure that the software has been represented according to predefined standards;

  (4) to achieve software that is developed in a uniform manner; and

  (5) to make projects more manageable.

- FTR serves as a training ground, enabling junior engineers to observe different approaches to software analysis, design, and implementation.
- The FTR also serves to promote backup and continuity because a number of people become familiar with parts of the software that they may not have otherwise seen.
- The FTR is actually a class of reviews that includes walkthroughs and inspections.
- Each FTR is conducted as a meeting and will be successful only if it is properly planned, controlled, and attended.

1.The Review Meeting

Constraints:

- Between three and five people (typically) should be involved in the review.
- Advance preparation should occur but should require no more than two hours of work for each person.
- The duration of the review meeting should be less than two hours.

- The focus of the FTR is on a work product (e.g., a portion of a requirements model, a detailed component design, source code for a component).
- The individual who has developed the work product—the producer—informs the project leader that the work product is complete and that a review is required.
- The project leader contacts a review leader, who evaluates the product for readiness, generates copies of product materials, and distributes them to two or three reviewers for advance preparation.
- Each reviewer is expected to spend between one and two hours reviewing the product, making notes..

- The review meeting is attended by the review leader, all reviewers, and the producer.
- One of the reviewers takes on the role of a recorder, that is, the individual who records (in writing) all important issues raised during the review.
- The FTR begins with an introduction of the agenda and a brief introduction by the producer.
- The producer then proceeds to “walk through” the work product, explaining the material, while reviewers raise issues based on their advance preparation. When valid problems or errors are discovered, the recorder notes each.

At the end of the review, all attendees of the FTR must decide whether to:

(1) accept the product without further modification,

(2) reject the product due to severe errors (once corrected, another review must be performed), or

(3) accept the product provisionally (minor errors have been encountered and must be corrected, but no additional review will be required).

After the decision is made, all FTR attendees complete a sign-off, indicating their participation in the review and their concurrence with the review team’s findings.

2.Review Reporting and Record Keeping

- During the FTR, a reviewer (the recorder) actively records all issues that have been raised.
- These are summarized at the end of the review meeting, and a review issues list is produced.
- In addition, a formal technical review summary report is completed.
- A review summary report answers three questions:
  - 1. What was reviewed?
  - 2. Who reviewed it?
  - 3. What were the findings and conclusions?

- The review issues list serves two purposes:

  (1) to identify problem areas within the product and

  (2) to serve as an action item checklist that guides the producer as corrections are made.

- An issues list is normally attached to the summary report.

3. Review Guidelines

Minimum set of guidelines for formal technical reviews:

1. Review the product, not the producer
2. Set an agenda and maintain it.
3. Limit debate and rebuttal.
4. Enunciate problem areas, but don't attempt to solve every problem noted..
5. Take written notes.
6. Limit the number of participants and insist upon advance preparation.
7. Develop a checklist for each product that is likely to be reviewed.
8. Allocate resources and schedule time for FTRs..
9. Conduct meaningful training for all reviewers.
10. Review your early reviews

POST-MORTEM EVALUATIONS

- Many lessons can be learned if a software team takes the time to evaluate the results of a software project after the software has been delivered to end users.
- Unlike an FTR that focuses on a specifi c work product, a PME examines the entire software project, focusing on both “ excellences and challenges” a PME is attended by members of the software team and stakeholders.
- The intent is to identify excellences and challenges and to extract lessons learned from both.
- The objective is to suggest improvements to both process and practice going forward.