Introduction to Software Engineering

(CS350)

Lecture 15

Jongmoon Baik

Review Techniques

What is A Peer Review?

- **"A type of software review in which a work product is examined by its author and/or one or more colleagues of its author, in order to evaluate its technical content and quality."** - Wikipedia

- **"The review of work products performed by peers during development of the work products to identify defects for removal."**
  - CMMI Guidelines

What Reviews Are Not

- A project summary or progress assessment
- A meeting intended solely to impart information
- A mechanism for political or personal reprisal!

Objectives of Peer Reviews

- To verify the work product meets requirements
- To detect and remove defects from the software work products early and efficiently
- To gain confidence in work products
- To prevent their leakage into field operations
- To reduce risk

Benefits of Software Peer Reviews

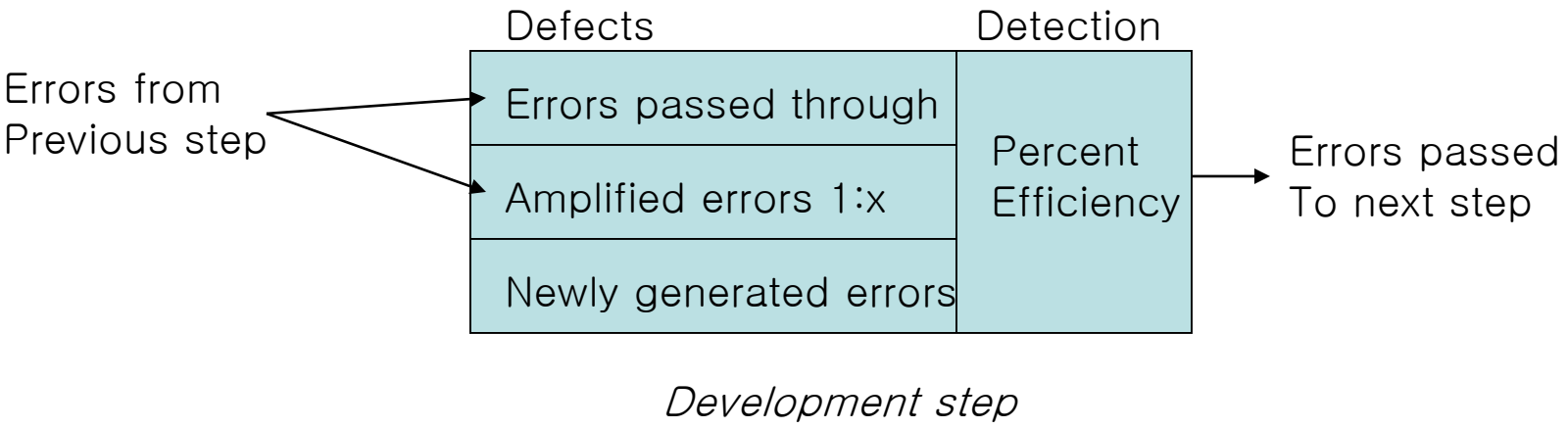
- **Get another perspective**
  - Finding defects can be easier for someone who hasn't seen the artifact before and doesn't have preconceived ideas about its correctness
- **Knowledge transfer about:**
  - software artifacts and defect detection
- **Detect and Remove errors early**
  - Dramatic cost reduction to fix them
- **Reduce rework and testing effort**
  - overall development effort reduction

Formal/Informal Reviews

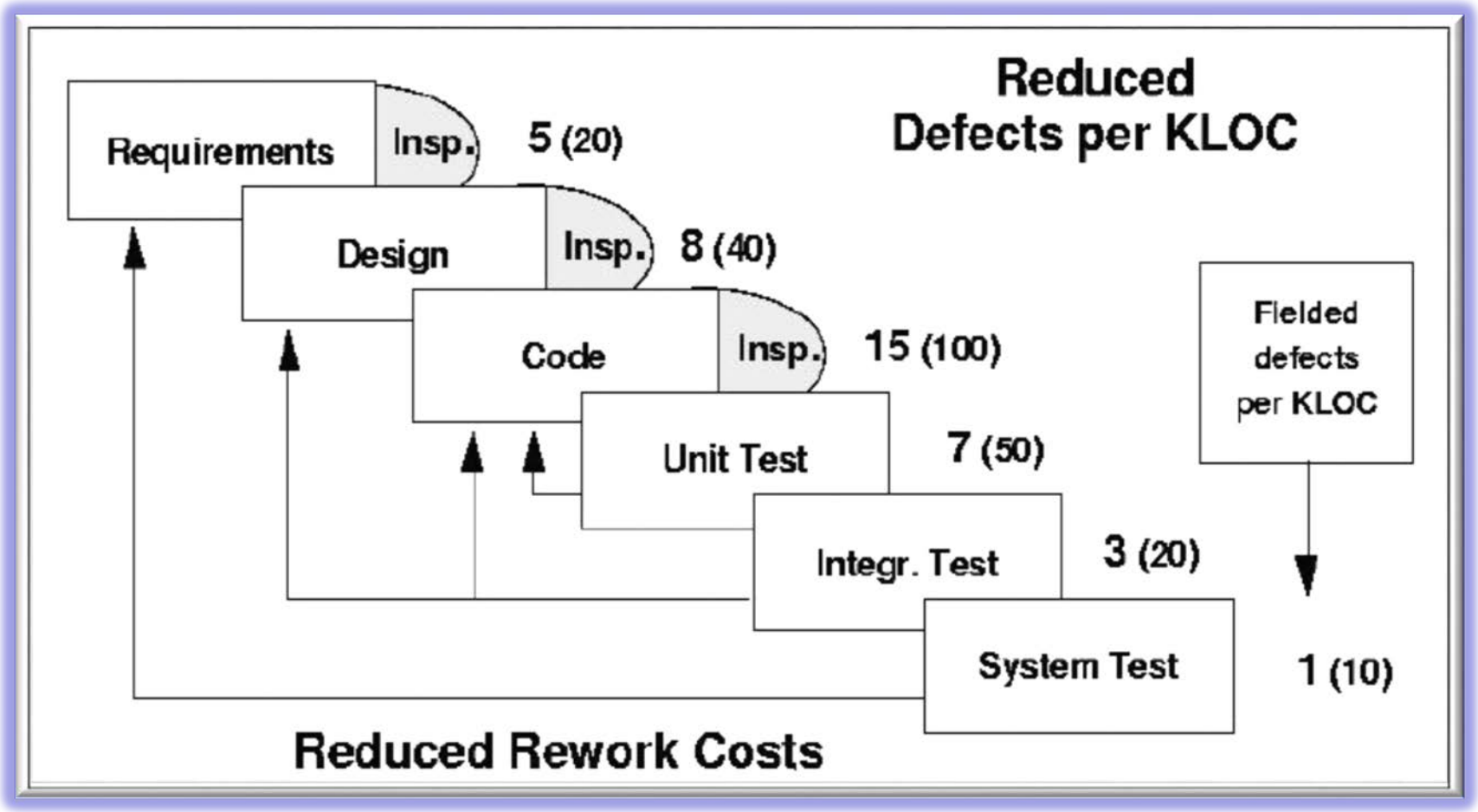
- **Formal Review:**
  - Highly structured process for preparation, meeting protocol, data collection, and post meeting activities
  - Table-top meeting oriented
  - Excellent for ensuring final product quality and standards
  - 예: Software Fagan Inspection, Formal Technical Review…
- **Informal Review:**
  - Less structure, formality, and rigidity
  - More open discussion (peer review)
  - Table-top or presentation oriented
  - Excellent for initial artifact presentation, artifact discussion, or selection of alternatives
  - 예: Walkthrough…
- **Both formal and informal inspections have their place in a project**

Defect Amplification

- A *defect amplification model* [IBM81] can be used to illustrate the generation and detection of errors during the design and code generation actions of a software process.



Defect Profile w/ Review



Metrics

- The total review effort and the total number of errors discovered are defined as:
  - $E_{review} = E_p + E_a + E_r$
  - $Err_{tot} = Err_{minor} + Err_{major}$
- *Defect density* represents the errors found per unit of work product reviewed.
  - Defect density = Err_{tot} / WPS
- where …

Metrics

- *Preparation effort, E_p*—the effort (in person-hours) required to review a work product prior to the actual review meeting
- *Assessment effort, E_a*— the effort (in person-hours) that is expending during the actual review
- *Rework effort, E_r*— 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, Err_{minor}*—the number of errors found that can be categorized as minor (requiring less than some pre-specified effort to correct)
- *Major errors found, Err_{major}*— the number of errors found that can be categorized as major (requiring more than some pre-specified effort to correct)

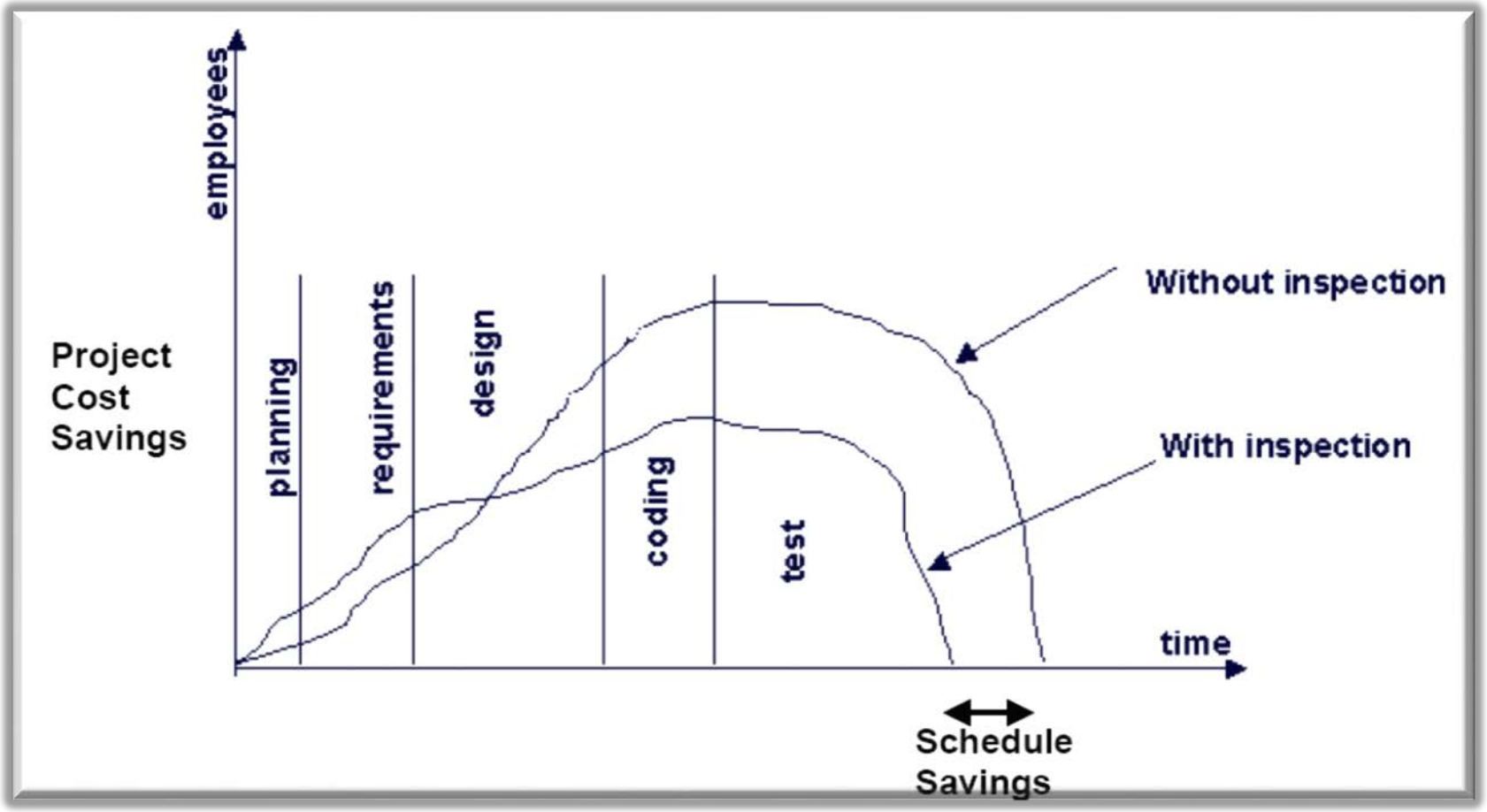
Metrics Derived from Reviews

- inspection time per page of documentation
- inspection time per KLOC or FP
- inspection effort per KLOC or FP
- errors uncovered per reviewer hour
- errors uncovered per preparation hour
- errors uncovered per SE task (e.g., design)
- number of minor errors (e.g., typos)
- number of major errors
  (e.g., nonconformance to req.)
- number of errors found during preparation

Software Inspection

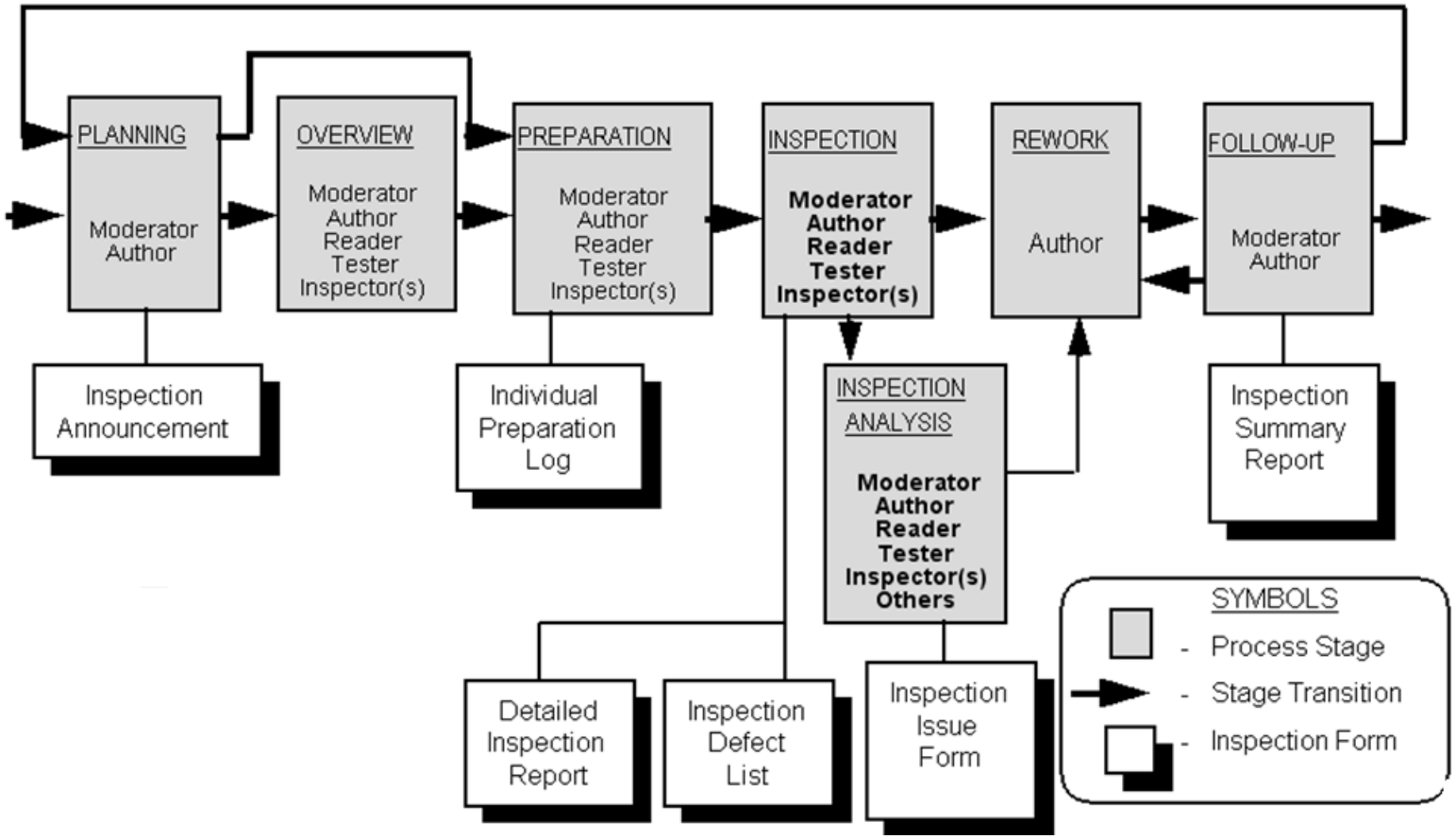
- **Defined by Michael Fagan (IBM) in 1976**
- **Consists of 7 operation stages**
- **Its objectives are to:**
  - Find all the defects in the work product that is examined
  - Find all the systemic defects in the process that created defects in the work product.
- **It also enabled:**
  - Measurement of defects
  - Management of defect rework
  - Removal of systemic defects from the development process

Effect of Inspection



출처: Michael Fagan

Inspection Process



Inspection Operation Stages

PROCESS OPERTION

1 PLANNING

2 OVERVIEW

3 PREPARATION

4 INSPECTION MEETING

5 PROCESS IMPROVEMENT

6 REWORK

7 FOLLOW-UP

OBJECTIVES

- Material, Inspectors, schedule.
- Present Overview.
- Learn material, prepare role, DO NOT focus on finding defects.
- **FIND DEFECTS!**
- Learn from last 4 operations to improve next inspection.
- Identify SYSTEMIC DEFECTS in the process and recommend fixes.
- Rework all defects.
- Verify all fixes.

Role of Inspectors- I

- Moderator
  - Organizes Inspection
    - Keeps discussion on track
    - Ensures follow-up happens
- Reader
  - Presents material (different from author)
    - Provides point of comparison for author and other team members
      - Differences in interpretation provoke discussion
      - Reveals ambiguities vs. if author were to present

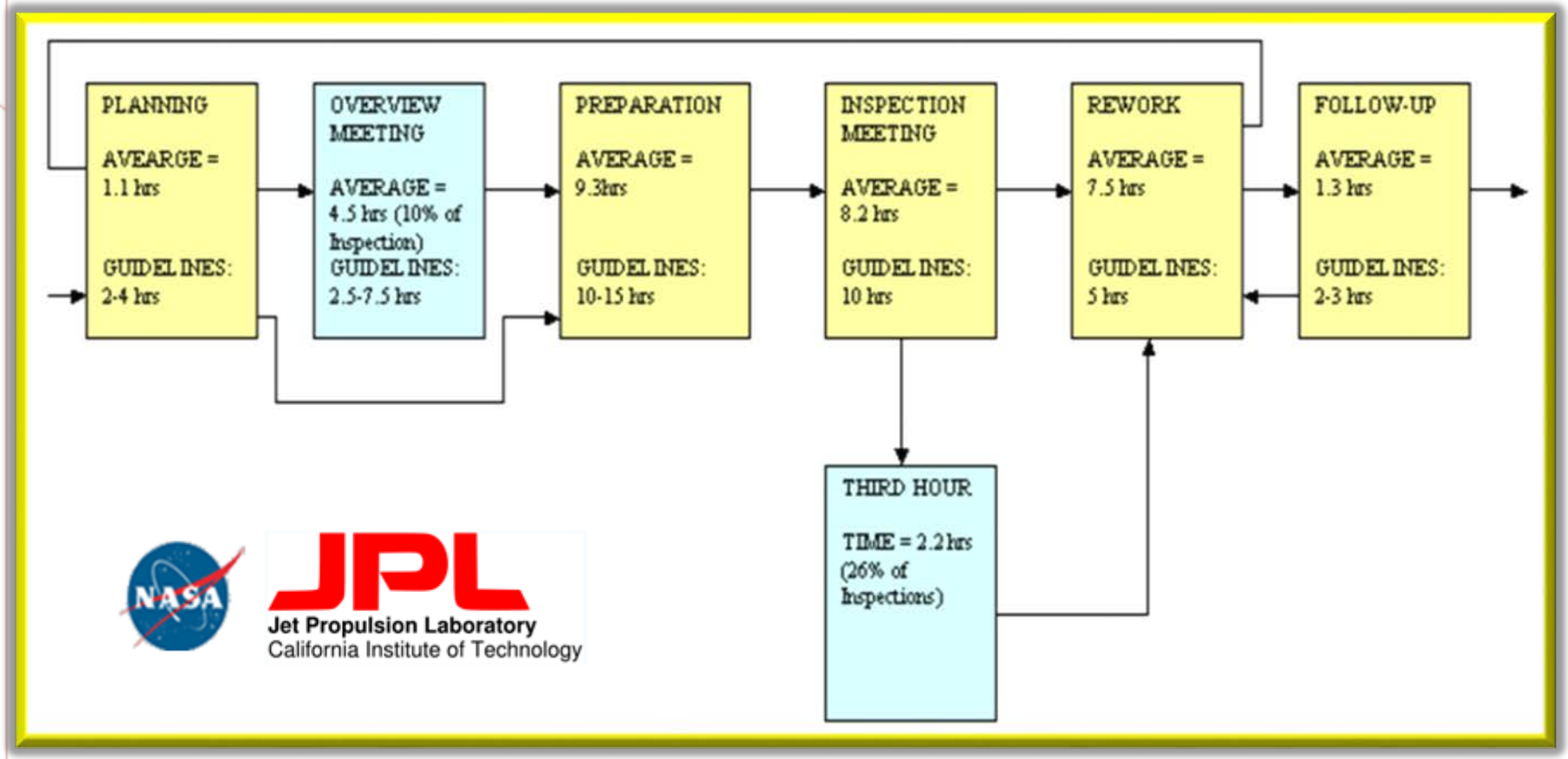
Role of Inspectors- II

- Recorder
  - Writes down identified defects with explanations
- Author
  - Not moderator or reader
    - Hard for author to be objective when presenting work or moderating discussion
    - Other inspectors can raise issues more comfortably
  - Not recorder
    - Temptation to not write down issues the author disagrees with
  - Significant benefits to attending
    - Gain insight from others' perspectives
    - Can answer questions
    - Can contribute to discussion based on knowledge of artifact; others' discussion may stimulate ideas
    - Only downside: meeting is more confrontational

Inspection guidelines - I

- Build reviews into your schedule
  - Otherwise, viewed as intrusion
  - Recognize that reviews can accelerate schedule by reducing other V&V activities
- Keep review team small
  - General guidelines: 3-6 participants
    - 3 is minimum for formal process to work
    - Below 3 you don't get enough perspectives besides author
    - Above 7, work goes more slowly, scheduling is difficult, moderating is hard
  - Smaller groups for code, larger groups for other documents
- Find problems, but don't try to solve them and don’t blame author
  - Typically less expensive to address 1-on-1
  - Guideline: halt solution discussion after 1 minute
- Limit meetings to 2 hours maximum
  - Attention span gets lost beyond this
- No manager involved in the meeting
- Require advance preparation
  - Provides much of the value of a (formal) review

Inspection Guideline ~ II



E.g: Requirement Checklist

1. Do requirements exhibit distinction between functions and data?
2. Do requirements define all the information to be displayed to users?
3. Do requirements address system and user response to error condition?
4. Is each requirement stated clearly, concisely, and unambiguously?
5. Is each requirement testable?
6. Are there ambiguous or implied requirements?
7. Are there conflicting requirements?
8. Are there areas not addressed in the Software Requirement Specification (SRS) that need to be?
9. Are performance requirements (such as response time, data storage requirements) stated?

.....

E.g: Code Checklist

1. Data (DA)
   - ✓Is each variable correctly typed?
   - ✓Is each variable initialized before use?
   - ✓Is the initialization appropriate for the type?
   - ✓Can global variables be made local?
   - ✓Are buffers appropriately sized?
   - ✓Are buffer overflows checked?
   - ✓Is dynamically allocated memory freed?
2. Interface (IF)
   - ✓Are appropriate values returned from functions?
   - ✓Do function calls have correct parameter types/values?
   - ✓Are return values tested?
   - ✓Are parameters passed by reference modified correctly?
   - ✓Are parameters passed by value not modified?
3. Functionality (FN)
   - ✓Do comparisons use the correct logic?
   - ✓Do loops terminate?
   - ✓Do all loops iterate the correct number of times (no off-by-one errors)?
   - ✓Is behavior correct if a loop is never entered?
   - ✓Is there dead (unreachable) code?
   - ✓Do all switch statements have a default case?
   - ✓Do all switch arms have break statements? If not, is the ``fall through'' correct?
4. Input/Output (IO)
   - ✓Are files opened before use?
   - ✓Are files closed after use?
   - ✓Are buffers flushed at correct times?
   - ✓Are error conditions checked?
5. Other (OT)
   - ✓Any defect discovered that does not fall into one of the above categories?

E.g.: Inspection Record

Page ___ of ___ Date __/__/__

Code Inspection Log

Team _____

Module to be inspected _____

Role	Name	Prep time	Meeting time
Moderator	_____	_____	_____
Author	_____	_____	_____
Reader	_____	_____	_____
Inspector	_____	_____	_____
	Total Time	_____	_____

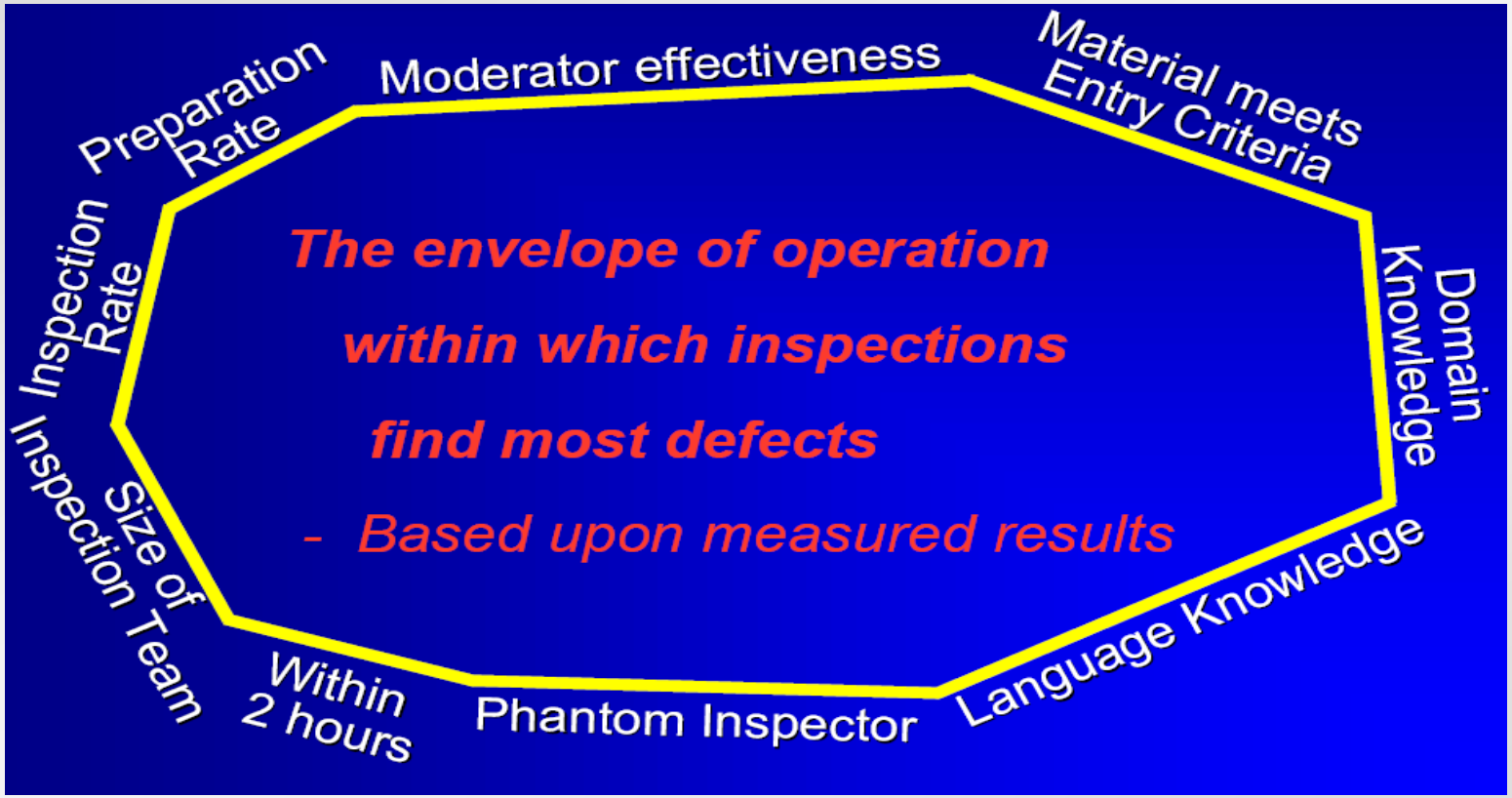
Total Engr effort _____ # of lines _____

Number of critical problems

…

Inspection Effectiveness Factors

* The effectiveness of inspection in finding defects that are present



http://www.sdm.de/download/sdm-konf2001/f_7_fagan.pdf

Phantom Inspector

- **Team synergy (due to cooperative team interaction) that cause identification of additional defects over and above the sum of defects found by individual inspectors.**

	Code	Requirements
Additional Defects found by the PHANTOM INSPECTOR	55 %	28 %

What is Walkthoruhg?

- **"A form of software peer review in which a designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems" [IEEE Std. 1028-1997, *IEEE Standard for Software Reviews*]**

- **An informal way of presenting a technical document in a meeting.**
  - No advance preparation
- **Unlike other kinds of reviews, the author runs the walkthrough:**
  - calling the meeting, inviting the reviewers, soliciting comments
  - ensuring that everyone present understands the work product.
- **No rigid process and formal follow-up**
- **Work products that are commonly reviewed using a walkthrough**
  - Requirement specification, design specifications and use cases…
- **Low cost , Valuable for education**

Walkthrough

- **An informal way of presenting a technical document in a meeting.**
  - No advance preparation
- **Unlike other kinds of reviews, the author runs the walkthrough:**
  - calling the meeting, inviting the reviewers, soliciting comments
  - ensuring that everyone present understands the work product.
- **No rigid process and formal follow-up**
- **Work products that are commonly reviewed using a walkthrough**
  - Design document, source code, and Use cases…
- **Low cost , Valuable for education**
- **Possible to review incomplete work products unlike Inspections**

Objectives of Walkthorugh

- Find anomalies
- Improve the software product
- Consider alternative implementation
- Evaluate conformance to standard and specifications

- Focusing on demonstrating how work product meets all the requirements
- To gain feedback about the technical quality or content of the document
- To familiarize the audience with the content

Roles of Walkthrough

- **Leader**
  - Conducts the walkthrough, handles administrative tasks, and ensures orderly conduct (and who is often the Author)
- **Recorder**
  - Notes all anomalies (potential defects), decisions, and action items identified during the walkthrough meeting
  - Generate minutes of meeting at the end of walkthrough session
- **Author**
  - Presents the software product in step-by-step manner at the walk-through meeting, and is probably responsible for completing most action items

IEEE Std. 1028-1997: *IEEE Standard for Software Reviews*, clause 38

Q & A