

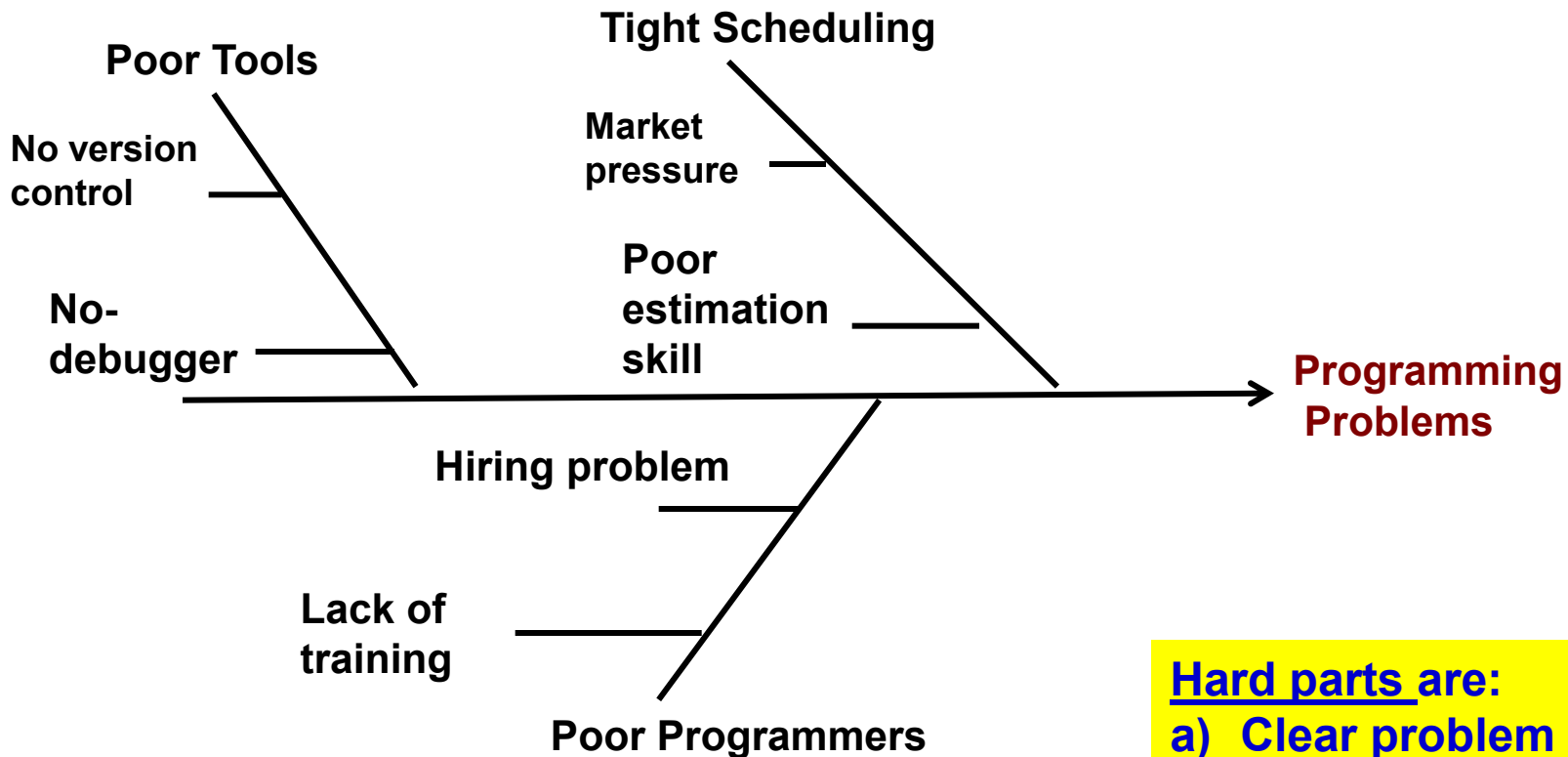
Defect Prevention & Defect Detection

- **Defect Prevention** techniques address how to prevent “error” before it is injected:
 - **Root Cause Analysis of problem**
 - **Education and training**
 - **Software process maturity and improvement (e.g. CMMI)**
- **Defect Detection** techniques address how to find the faults:
 - **Testing** (with various techniques covered in this class)
 - **Reviews and inspection**
 - **Formal verification (next lesson)**

Defect Prevention – Root Cause Analysis

- This is a structured approach to identifying the factors (causes) of problems; then follow-up with “addressing” and “fixing” these *potential causes*.
- One popular graphical tool used is the “**cause-and-effect**” diagram (developed by **Ishikawa** to study factors that affected the production of steel), which looks like a fishbone:
 - Place the **problem** at the right tip
 - Place the **causes** on the branches (out like a fish bone)

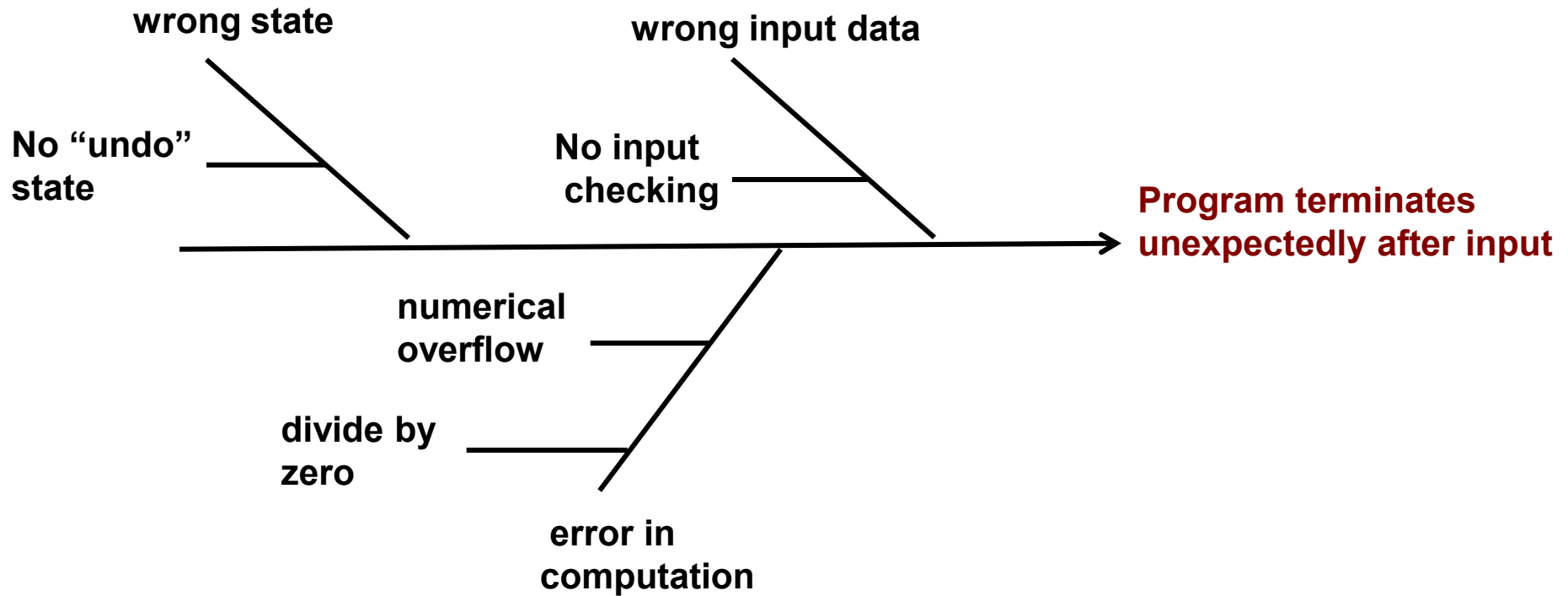
Cause-and-Effect Diagram (for error prevention)



Hard parts are:

- Clear problem def.
- Coming up with the potential "causes"

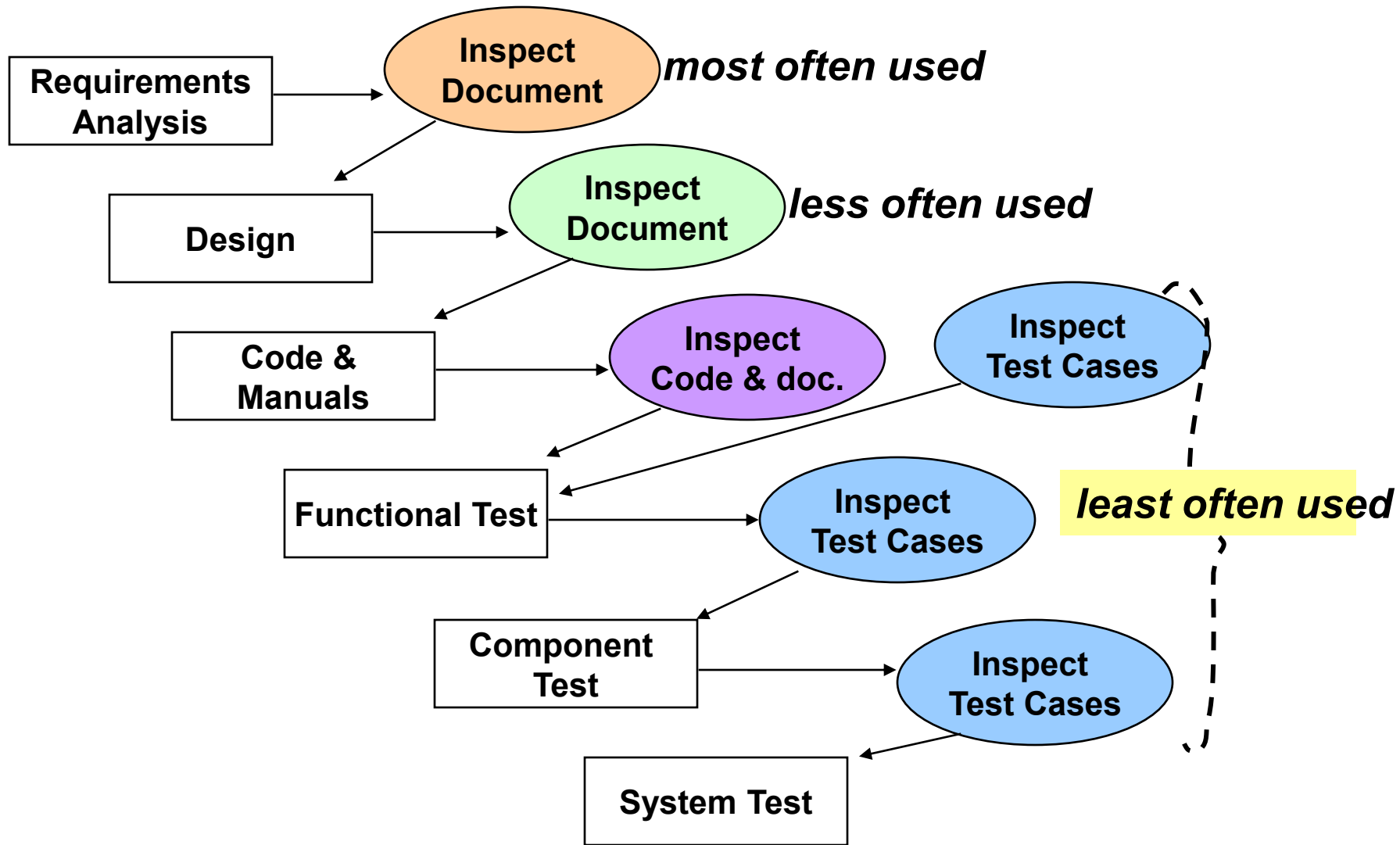
Cause-and-Effect Diagram (a more “specific” example)



Inspection and Review

- The main objective of an Inspection or a Review is to detect defects.
- This activity and procedure was first formalized by **Mike Fagan** of IBM during a period when CPU was still much more costly than people, and system complexity increased to the point where software quality became a huge issue.
- Inspections and reviews are testing of software artifacts without the actual execution of code and is especially suited for :
 1. Requirements - documents
 2. Design - documents
 3. Plans and Tests Cases – documents
 4. Customer education material - documents

Inspection as *Natural* Part of Software Development



Introduction to (Inspection/Review) Activity

- **Some Resistance**

1. What's the value?
2. Why so formal?
3. How to include into an already tight schedule?
4. Who should be involved?

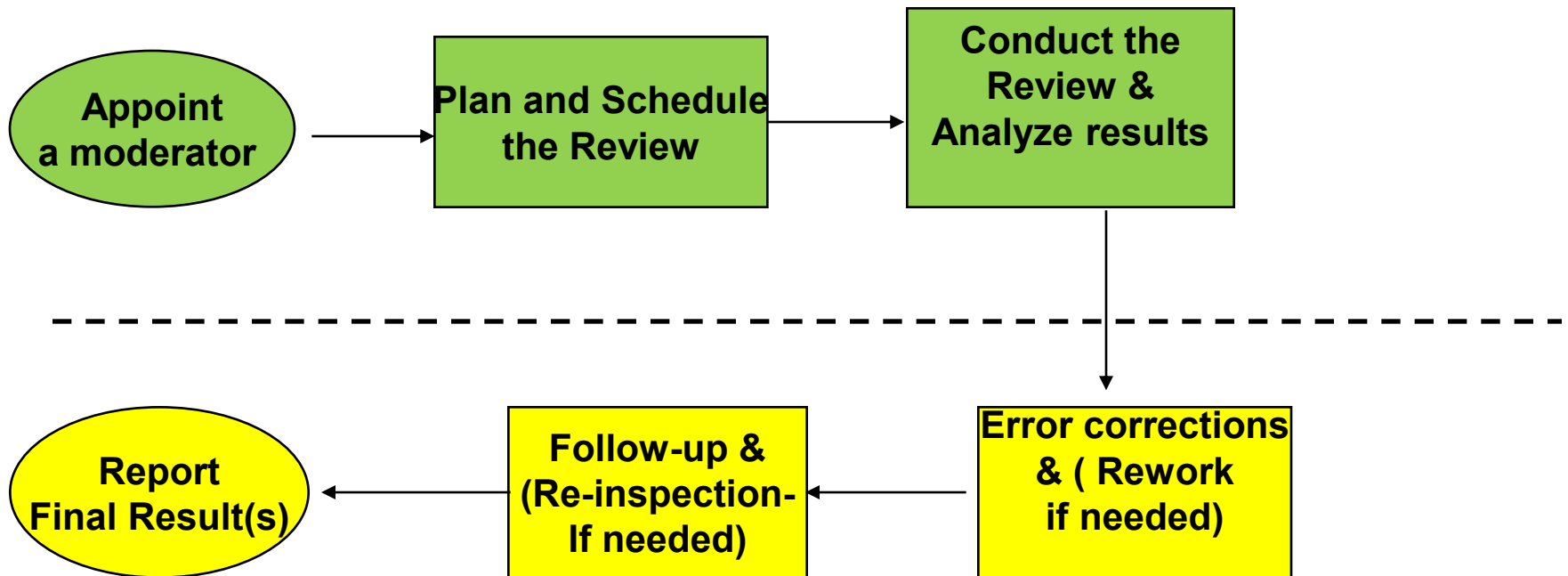
- **What do you think ?**

1. Compare the cost of bug found by customer ?
2. Would people take this seriously, otherwise?
3. Would code test time decrease enough to make up the difference?
4. How reliable are your "friends?" in telling you the truth?

Introduction of Review/Inspection Activities

- Must have complete “buy-in” --- (how do you do this? - use cost analysis?)
 - Upper management
 - Project leaders
 - Team members
- Must provide training for inspection/review:
 - **Not natural for people to look for defects** ---- **we like to show that things work**
 - **Delivering negative comments in a positive fashion**
 - **Focus on defects** and not just “better” solution
 - **Prioritize the problems** and not dwell on minor problems
 - **Agree on defect types, definitions, and priorities** at the end
- Reviewers must be **“prepared”** --- read and understand the material
- Must keep **good records** and **act on the result**

General Review Sub-Process Framework





rework and re-inspections are optional and needed only if the review results did not meet some prior criteria

Inspection/Review Sub-Process (more details)

Assuming no Re-review

Responsible	Activity (almost sequential)	“output”
Project mgr	<div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; display: inline-block;">Appoint moderator</div> ↓	“responsible/trained” moderator
Moderator	<div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; display: inline-block;">Plan, schedule, organize the review</div> ↓	review date set; review material disseminated; review team organized, review “rules” set; review-checklist (if applicable) established
Reviewers	<div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; display: inline-block;">Prepare for review</div> ↓	materials read; organized set of defects found while preparing for the actual review; priorities of found defects set; written down questions for the review
Moderator, Reviewers & Review material owner	<div style="border: 1px solid black; background-color: #e0ffe0; padding: 5px; display: inline-block;">Conduct the review</div> ↓	<u>Pre-review:</u> review rules re-explained, <u>Post review:</u> list of “agreed upon” prioritized problems; list of resolved problems; list of questions to be resolved (all lists must be “trackable”) <u>Decision:</u> re-review needed decision, problem resolution schedule

Inspection/Review Sub-Process (more details)

Responsible	Activity	“output”
review material owner	<p>Fix the defects in the material and provide them to the moderator</p> 	Fixed material
Moderator & reviewers	<p>Ensures the fixes are indeed correct by having the reviewer “sign off”</p> 	“Signed off” Fixed material
Moderator	<p>Prepare the final report and disseminate the final review report</p>	Final review report which contains the “previously agreed upon” statistics from the review.

Trained Participants

- **Moderator :**

- schedule and ensure the inspectors are ready for inspection
- keep the group to the major task of defect finding
- moderate the discussions and keep the activity moving
- arbitrate and decide on the defect severity level
- record, analyze, and report on the review results (may involve an assistant)
- follow up on the review
- report on the final result

- **Inspectors:**

- look for defects first versus look for “best” alternatives
- *establish what works* (not promoted by all practitioners of inspections)
- critical with the product, not the person (very hard to do, but must)
- participate in deciding on the severity of the defects found
- take the inspection result to their own “downstream” domain of activities

buggy areas deserve more attention and preparation

Classification of what's important (problems)

Example: Defect Severity levels by “problem impact”

- **Level 1 : Catastrophic error which causes the whole system to be down and possible loss of life.**
- **Level 2 : An error that will cause a major functional failure and there is no work around.**
- **Level 3 : An error that will cause a major functional error but there is a manual work around.**
- **Level 4 : An error that will cause a minor error or system degradation.**
- **Level 5 : A cosmetic level error that may be fixed at the next most convenient time**

How do these apply to Requirements Review?

Rework and Re-Inspection Guidelines

1. These guidelines must be set ahead of time.
2. What should rework guidelines be ?
 - All defects found in an inspection should be reworked and fixed?
 - All severity level 1 through level 3 must be reworked and fixed before the inspected artifact may be declared complete.
 - Should requirements and design artifacts be under a stricter rule ?
3. What should Re-Inspection guideline be ?
 - Re-inspect the complete artifact if there is “x” number of severity level 1 problems ?
 - Re-inspect only the rework and fixes for severity 1 and 2 problems ?

What Have We Learned in the last 40 years ?

- Inspections and reviews are expensive and time consuming
- Inspections and reviews **do bring down defect rates and also :**
 - force **some discipline** into an organization
 - brings **awareness of quality focus**
 - helps in **setting milestone marks** (inspection exit)
- Inspection and Reviews should be used selectively
 - mostly on non-executables
 - error prone and complex areas
 - new areas (new functions, new employees, new domain)