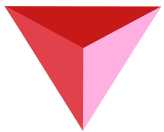


Quality Management

Chapter 15

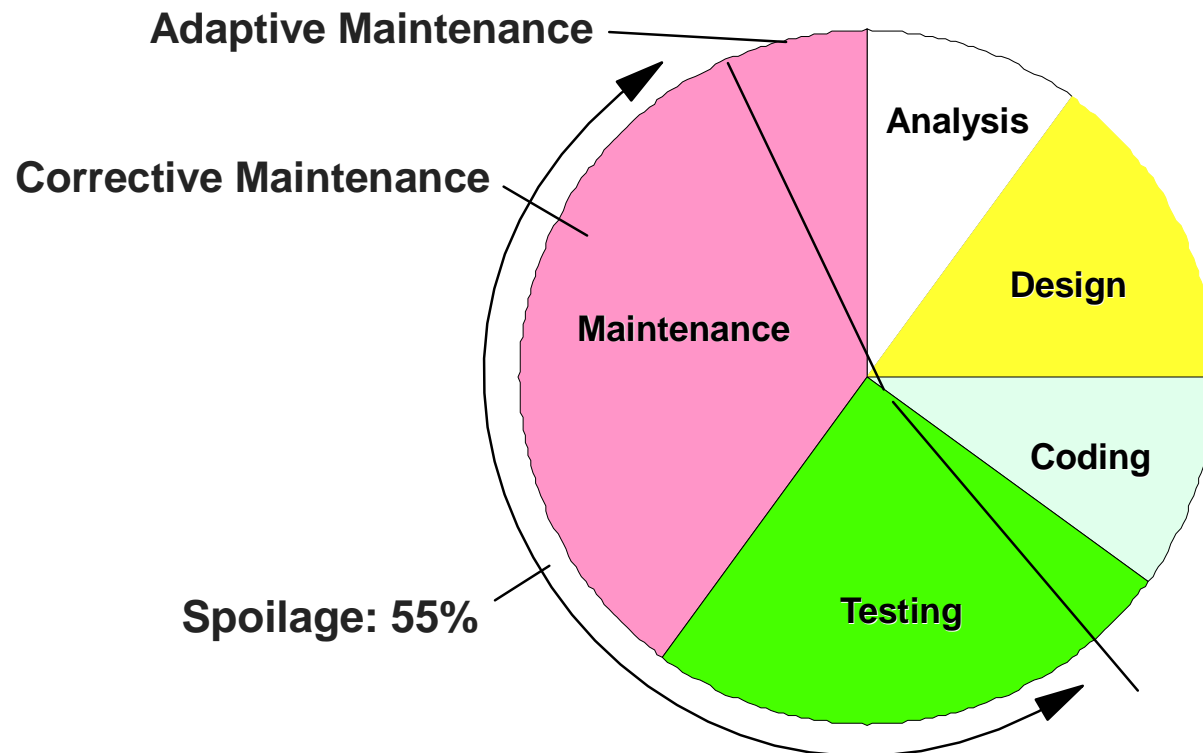
Managing Information
Technology Projects





Software Spoilage

Software Spoilage



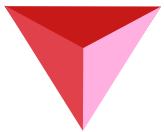
Source: Tom DeMarco, *Managing and Controlling Software Projects*

Figure 15.1



State of the Nation

- The new Denver airport opening was delayed for 18 months, at a cost of some \$1 Million per day due to problems in completion and reliability of an automated luggage handling system.
- In 1986, a Therac machine administered allegedly fatal doses of radiation to two patients after a software problem caused the machine to ignore calibration data. Paraphrased from Datamation, May 1987
- A software error was partly to blame for the information leading to the decision of a US aircraft carrier captain to fire upon a civilian airliner in the Gulf, causing the deaths of all aboard. *Newsweek story*



Ability to Deliver New Functionality

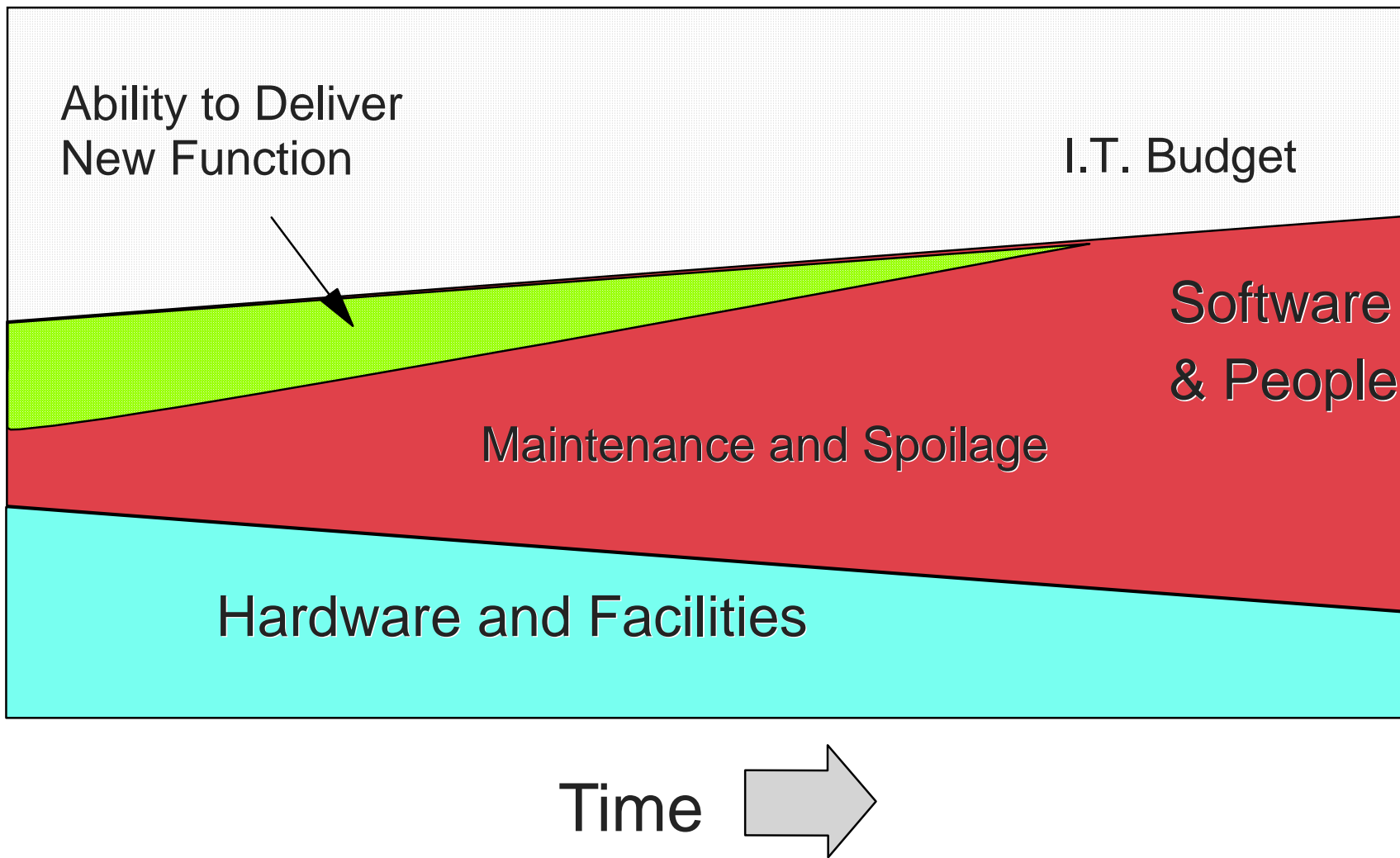
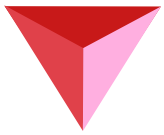


Figure 15.2



Turning Productivity Around

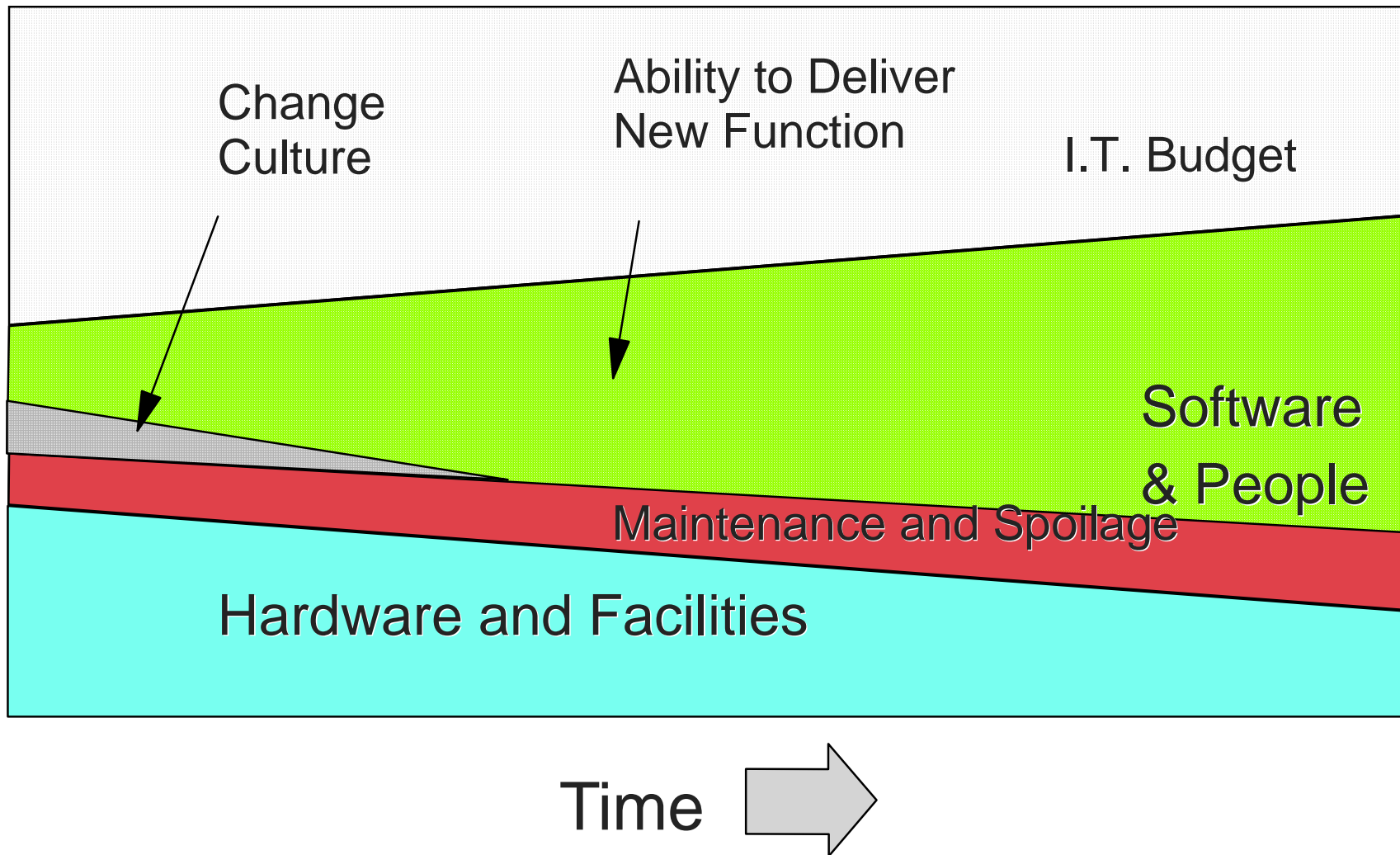
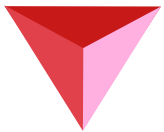


Figure 15.3



Quality-Team Effort

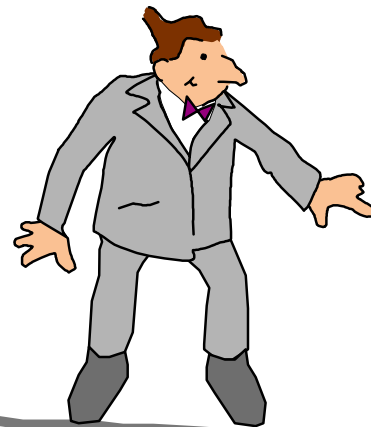
Quality Management

- Training time
- Best Equipment
- Equipment Maintenance
- Culture



Quality Assurance

- Balance bar
- Right shoes
- Personal training
- Fitness



Quality Control

- Prevents a mess if we slip

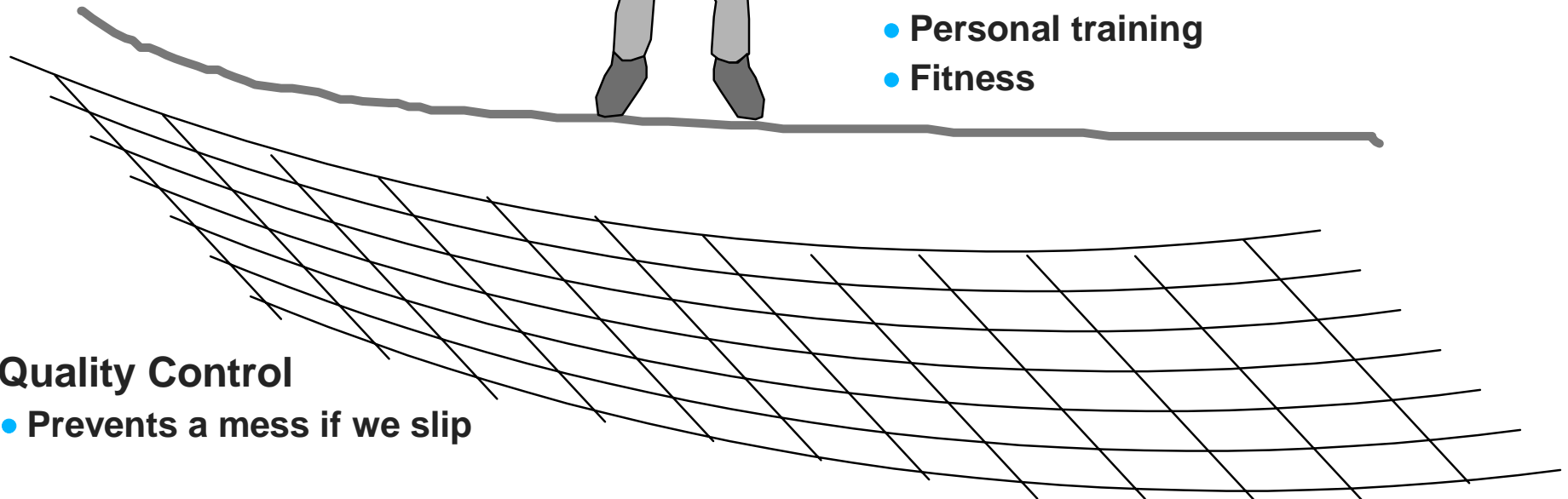
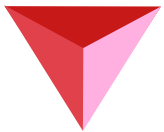


Figure 15.4



Quality-Appraisal Philosophy

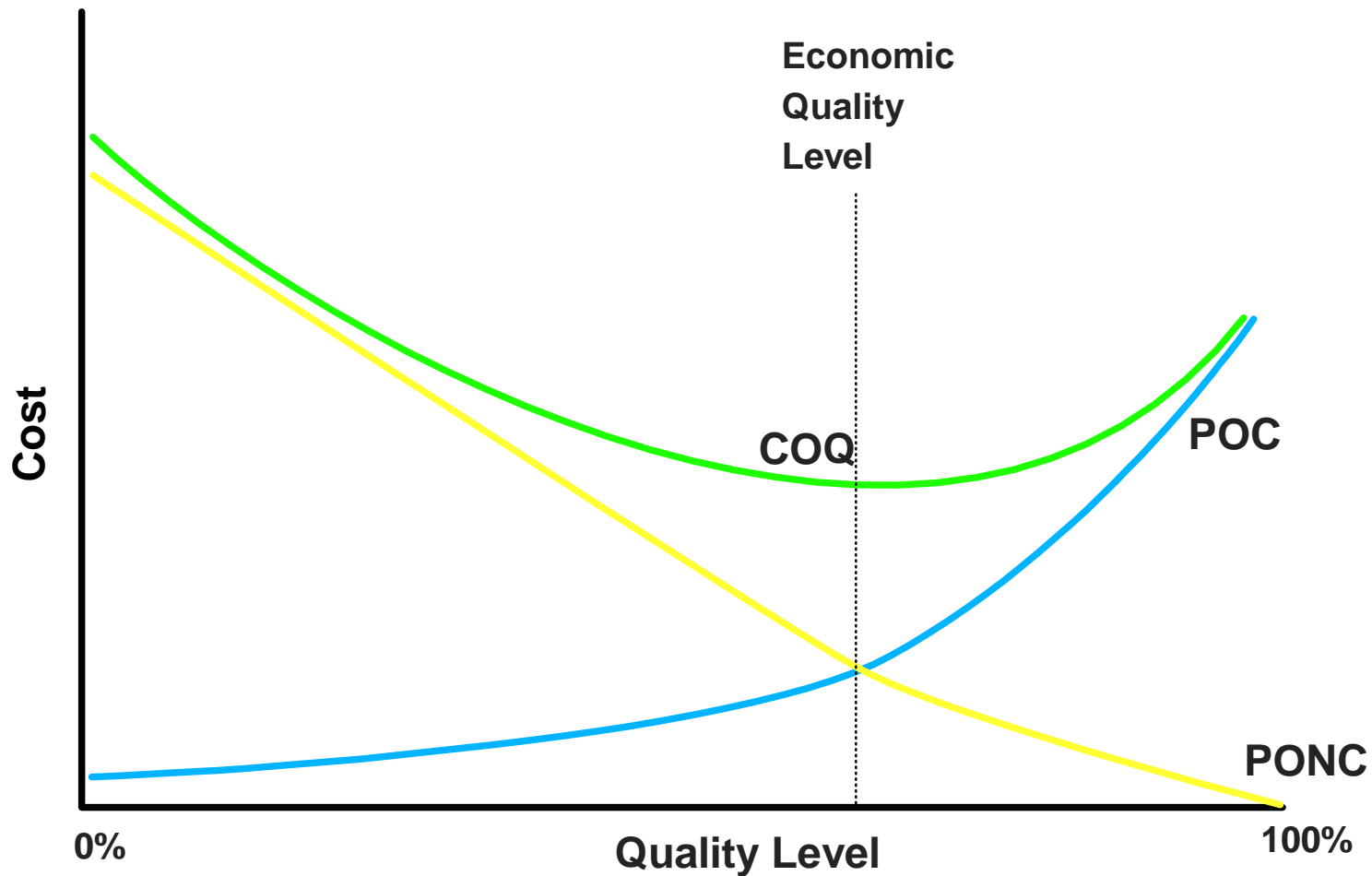


Figure 15.5

Quality-Prevention Philosophy

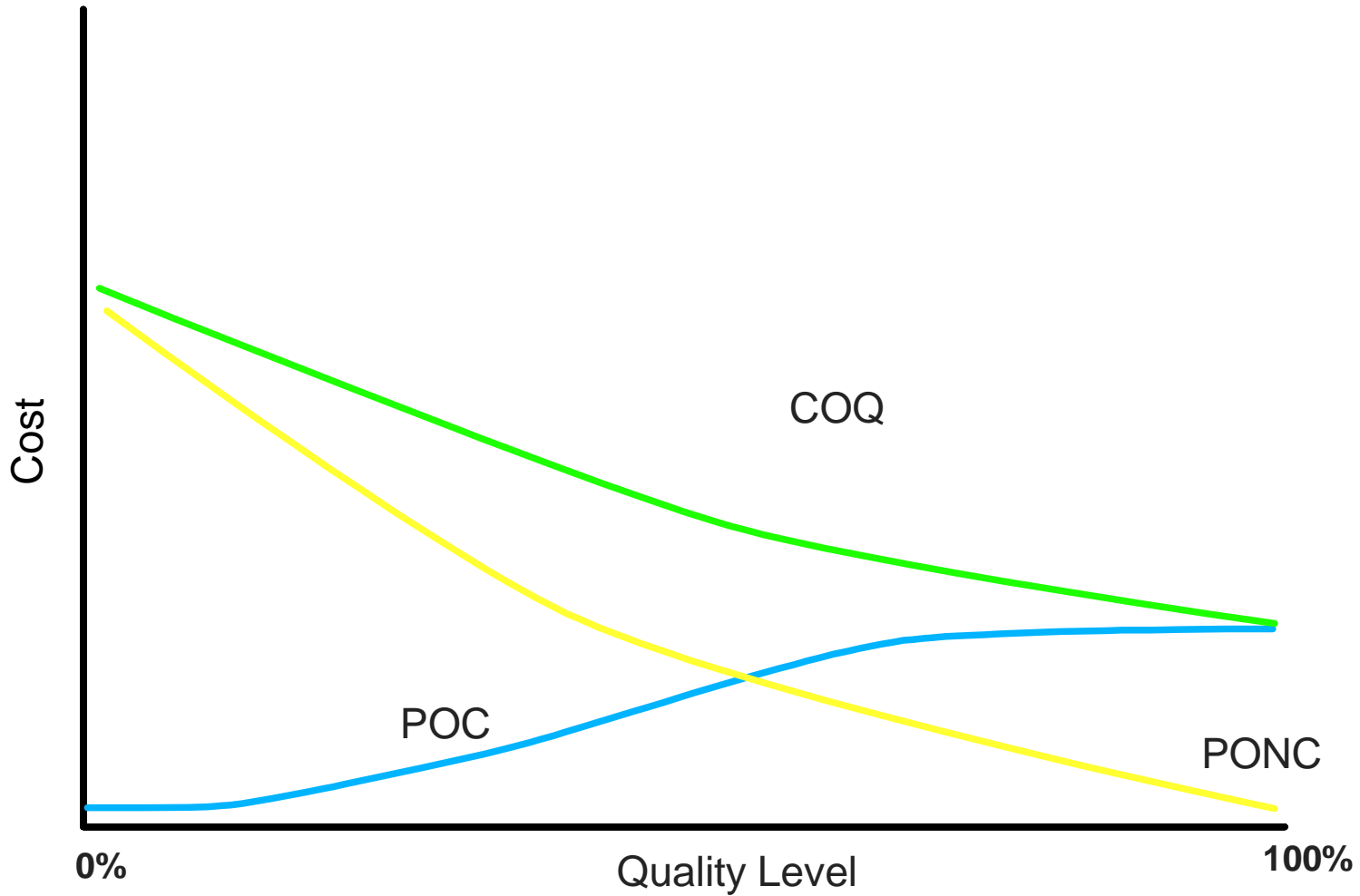
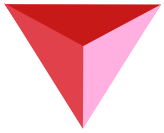


Figure 15.6



Prevention Scenario

- During integration testing, we find a Query program written in a 4GL which works correctly, but runs very inefficiently
- What do we do?
- Tune the program and continue testing
 - ✓ Determine why we found it so late - amend the unit test procedure
 - ✓ Look for other programs with a similar problem
 - we find some
 - ✓ Determine why they were written that way
 - turns out, that was how programmers were taught to do it on a recent vendor course
 - ✓ Fix the standards, communicate new way of working to all programmers
 - ✓ Liaise with vendor to get training corrected
 - ✓ Monitor and follow up until resolved

Quality Model

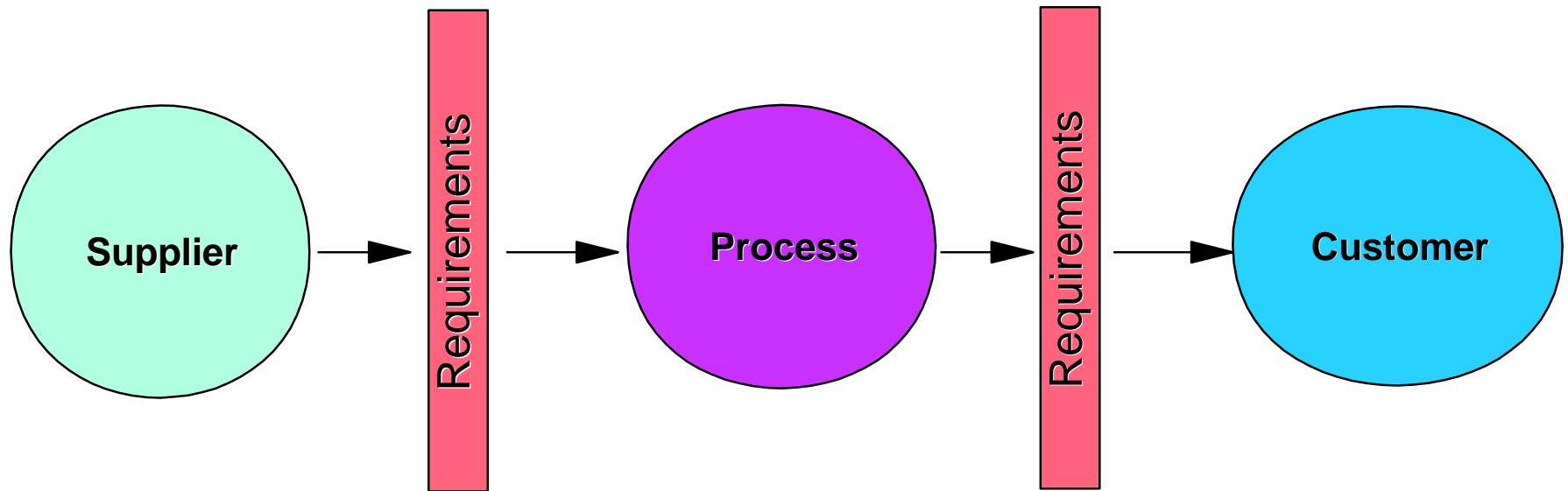


Figure 15.7

Quality Model for File Design

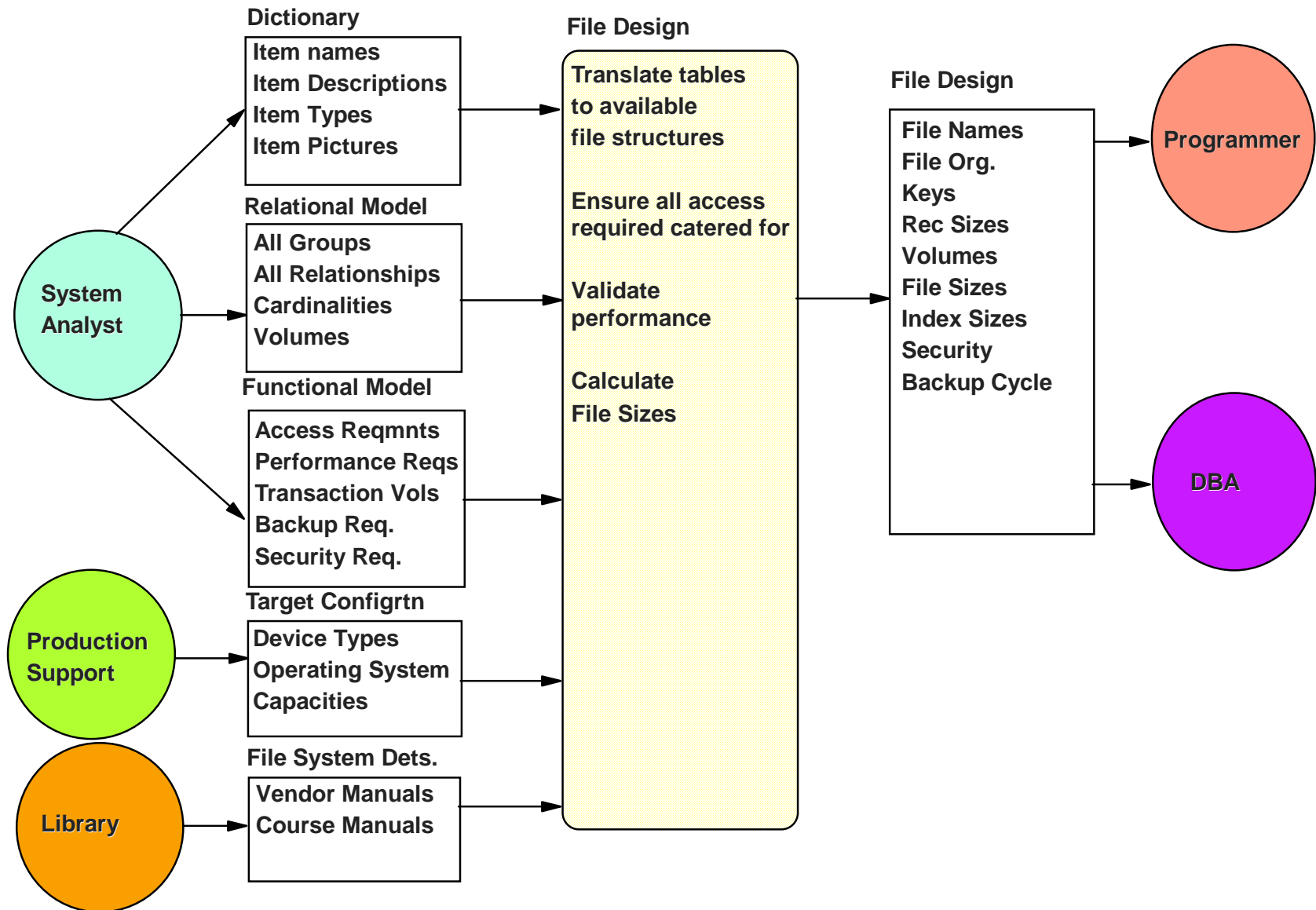
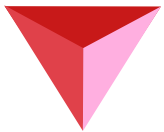


Figure 15.8



Work Breakdown Model

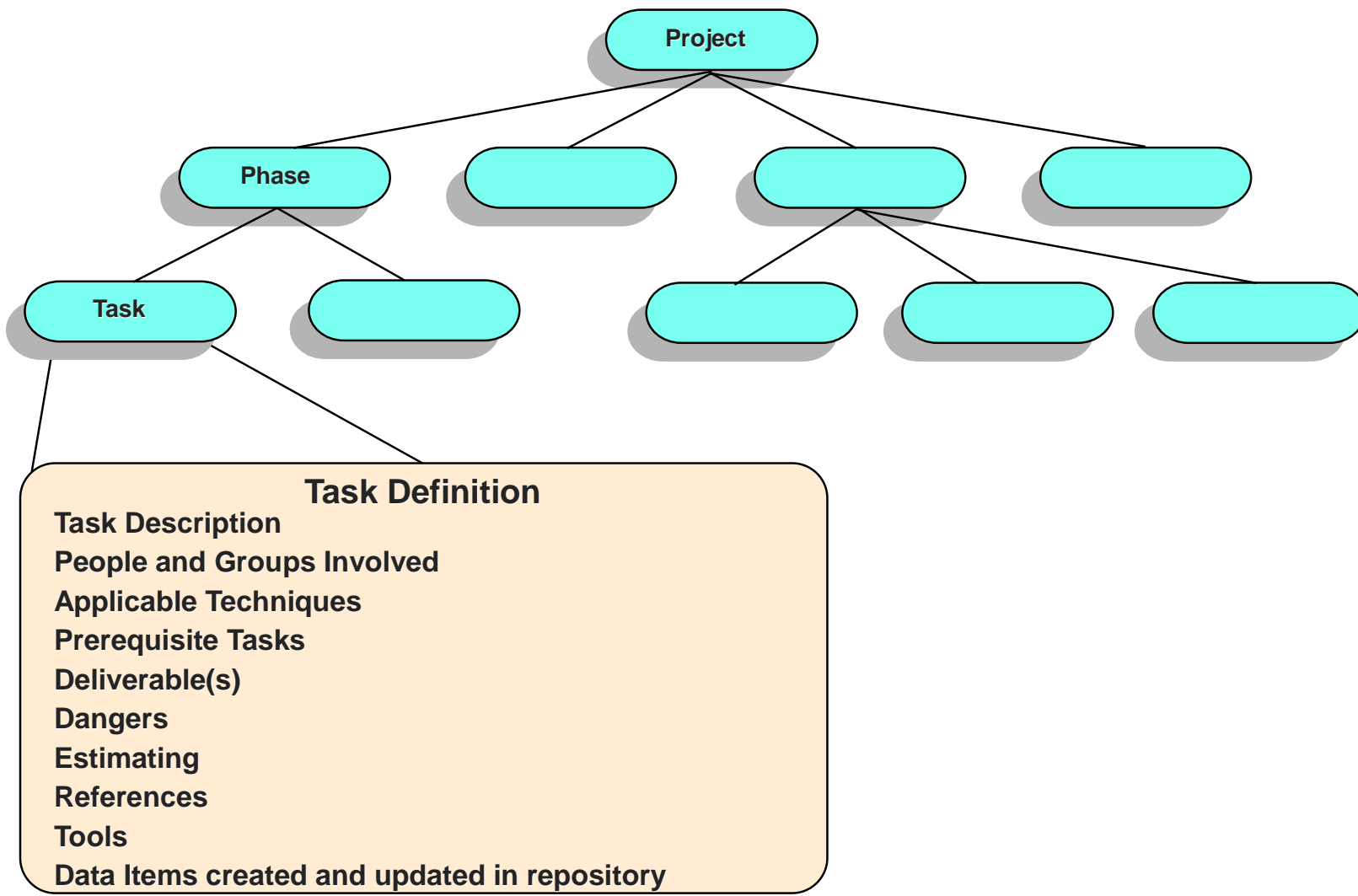


Figure 3.2

Differences Between Developers

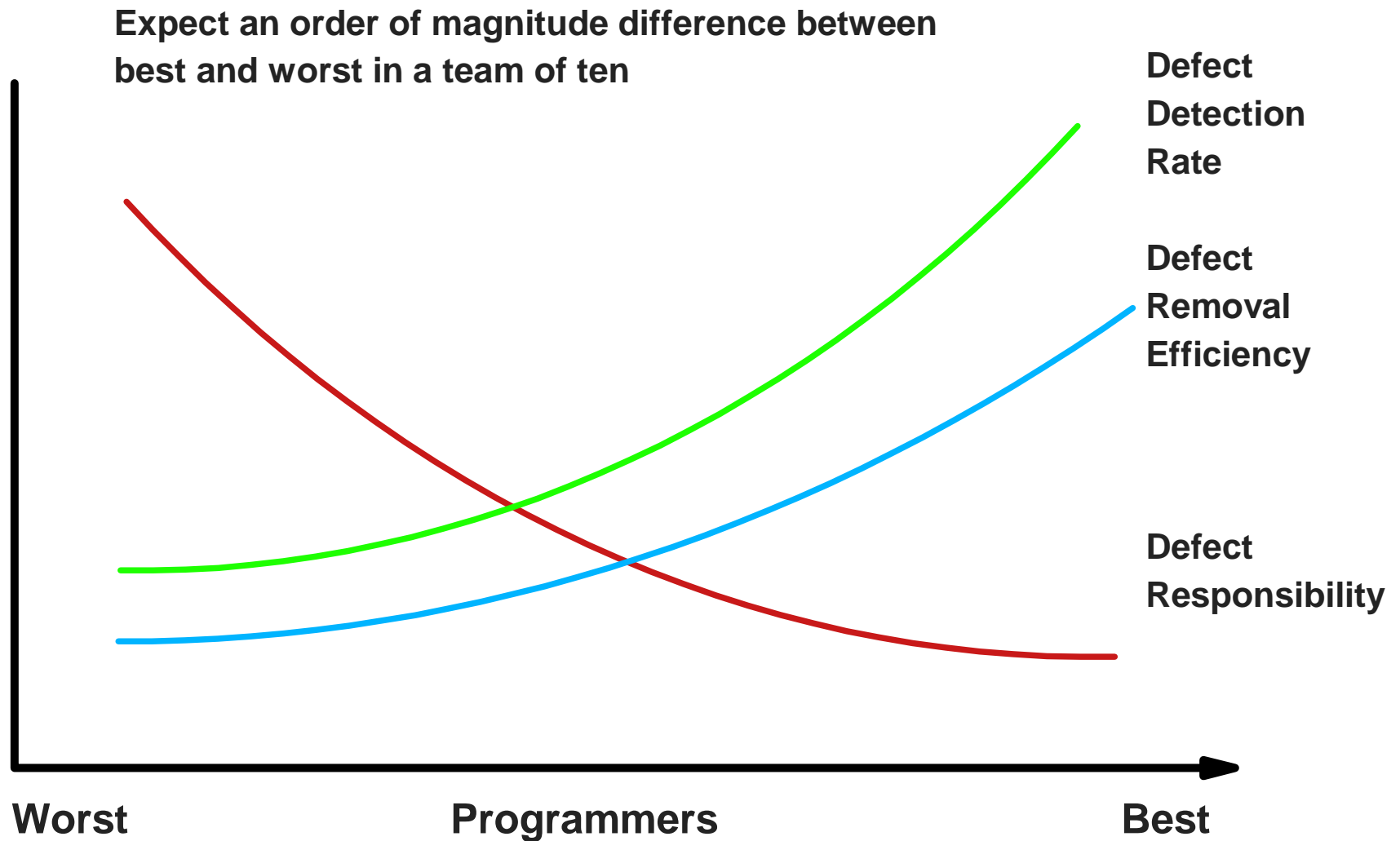
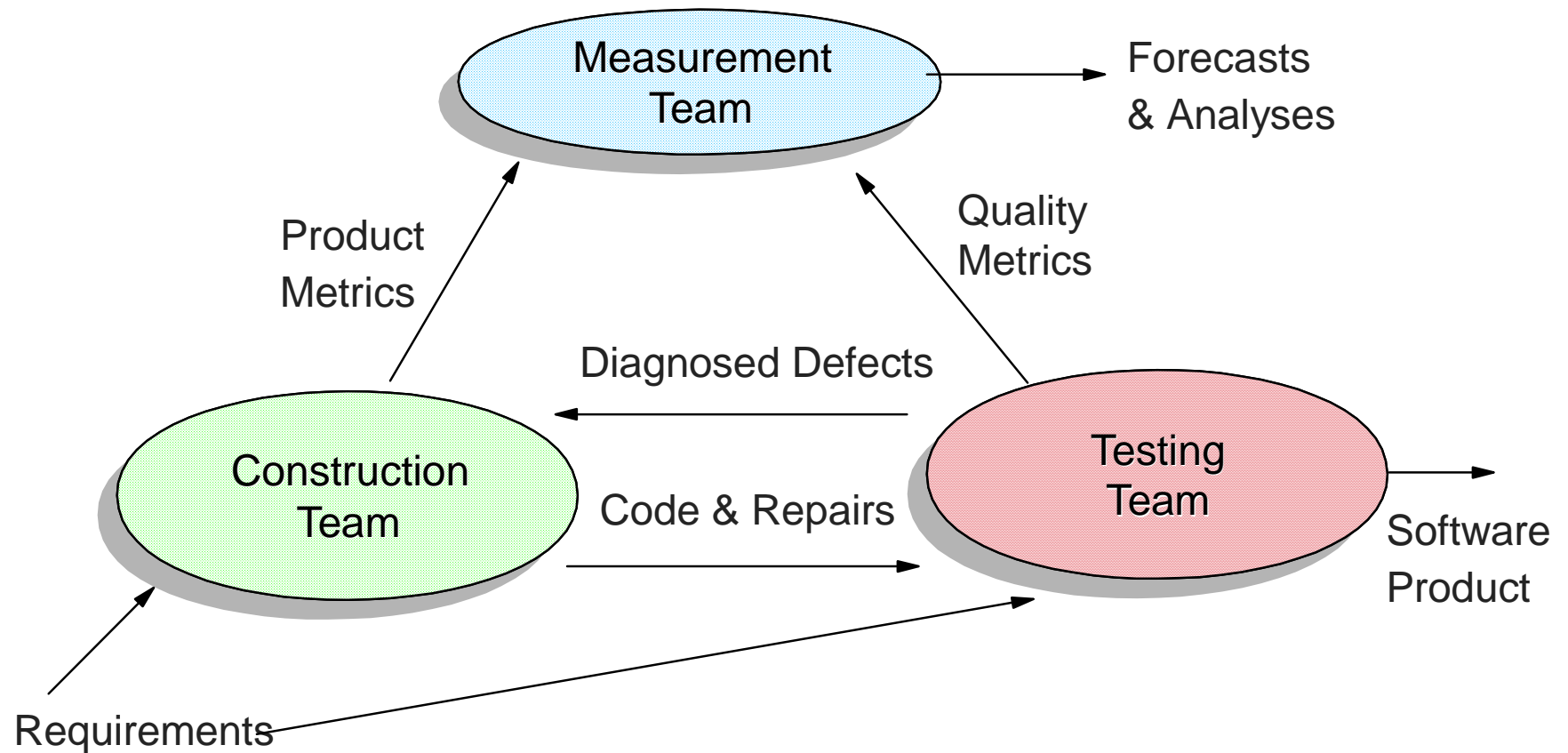


Figure 15.9

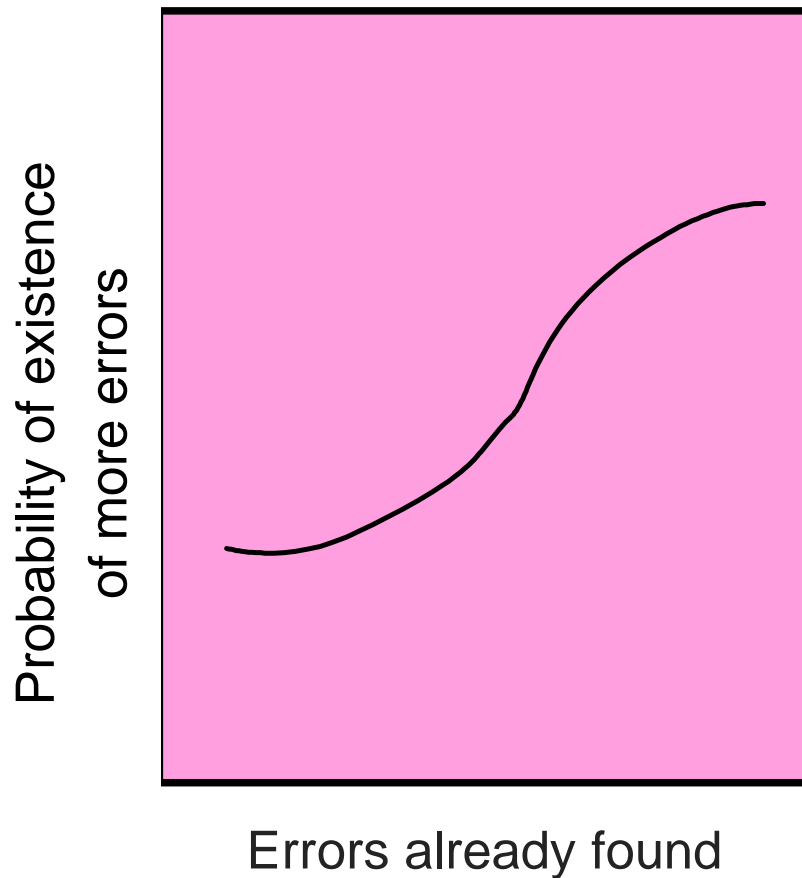
Adversary Teams



Example: The IBM "Black" team

Figure 15.10

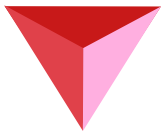
Myer's Findings



"The probability of the existence of more errors in a section of a program is proportional to the number of errors already found in that section"

- IMS - 57% defects in 7.3% of modules
- Defects' presence is not consistent
- Cockroach theory
 - Kill the nest

Figure 15.11



Monitoring Defect Removal

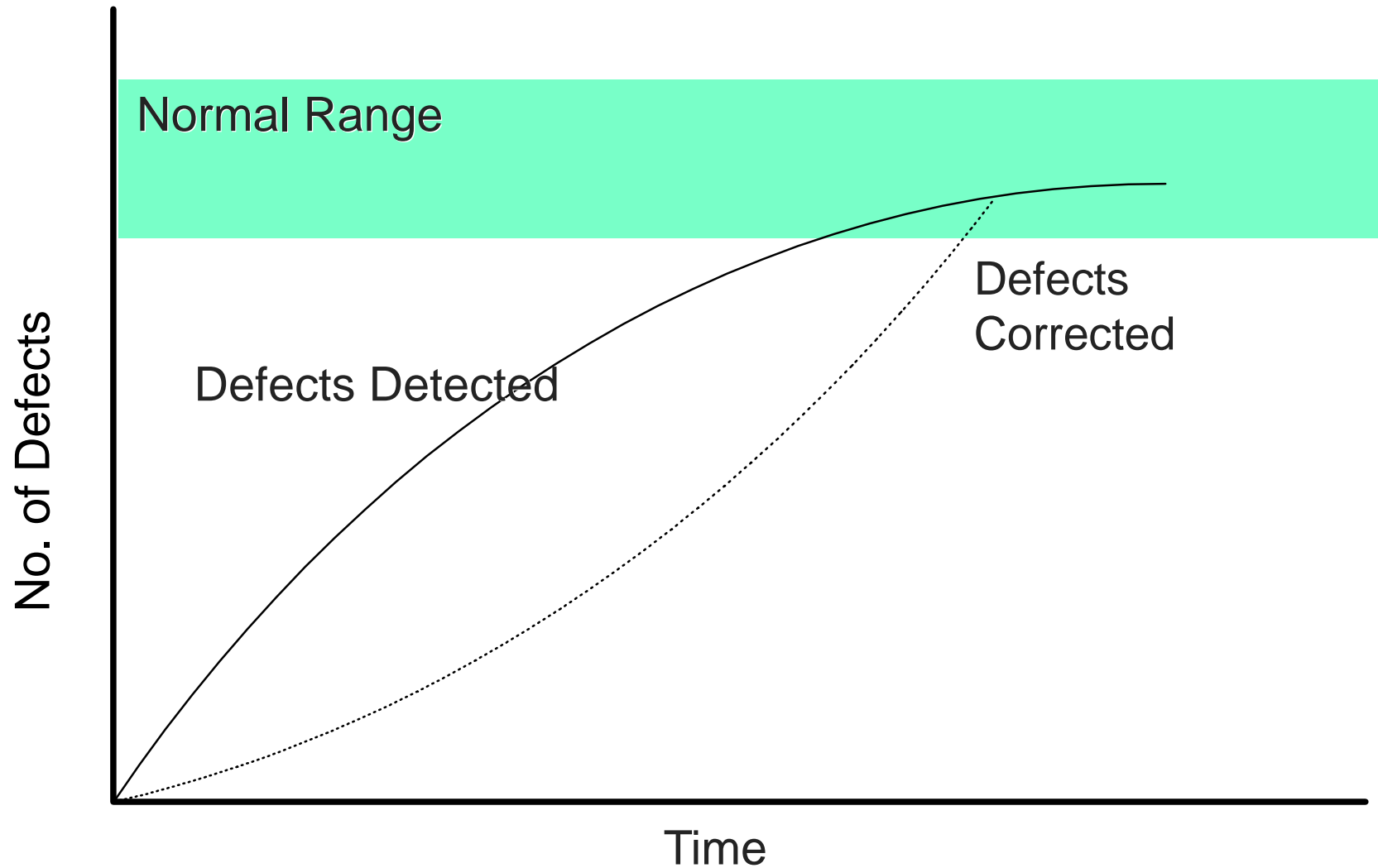
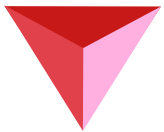


Figure 15.12



Problem Incidence as Reliability Indicator

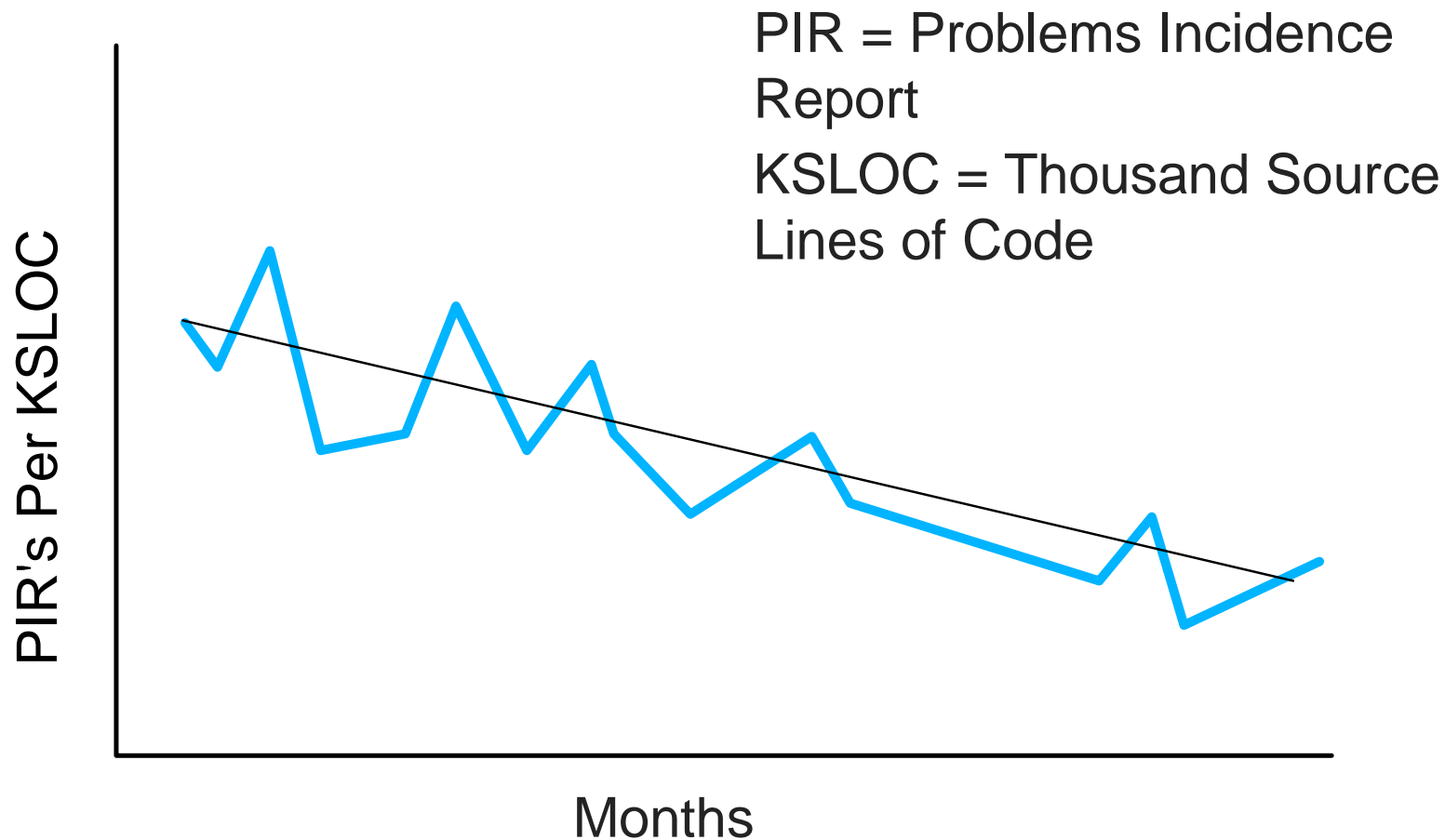


Figure 15.13

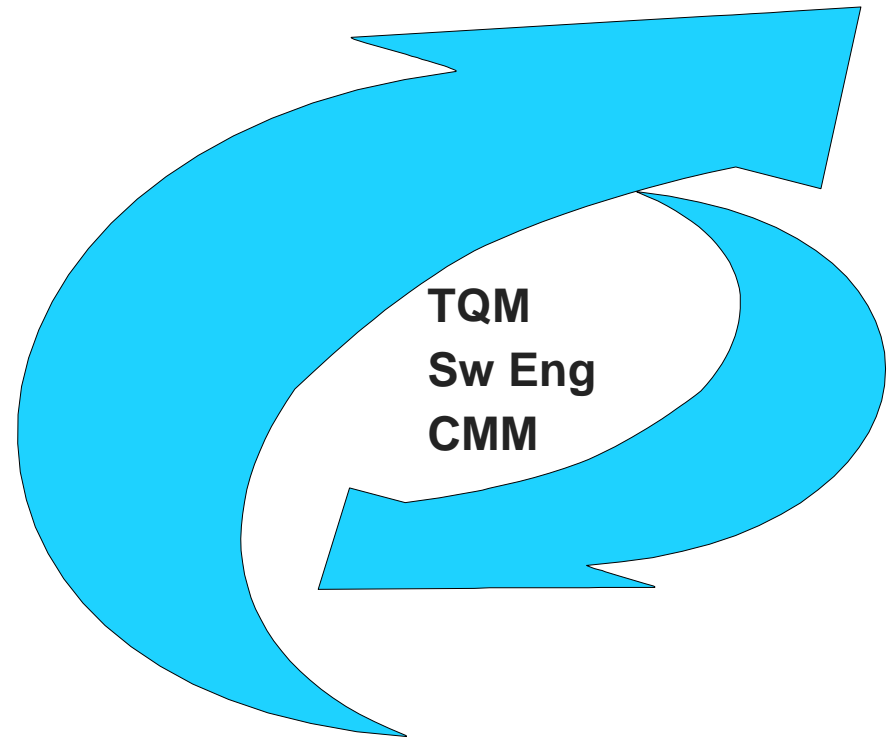
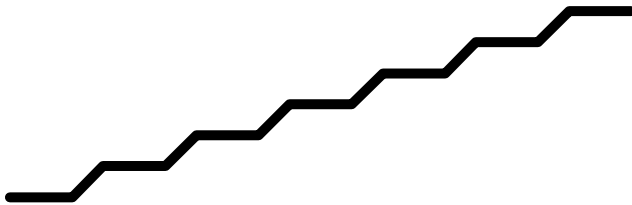
Innovation

- New Ideas
- Radical
- n x change
- Unreliable
- Invasive
- High Risk
- "Western"



Continuous Improvement - *Kaizen*

- Slow, incremental improvements
- $.n$ times improvement
- Reliable
- Not disruptive
- Sustainable



Quality Improvement at Hitachi

Aggregate Spoilage x 100
Aggregate Project Cost

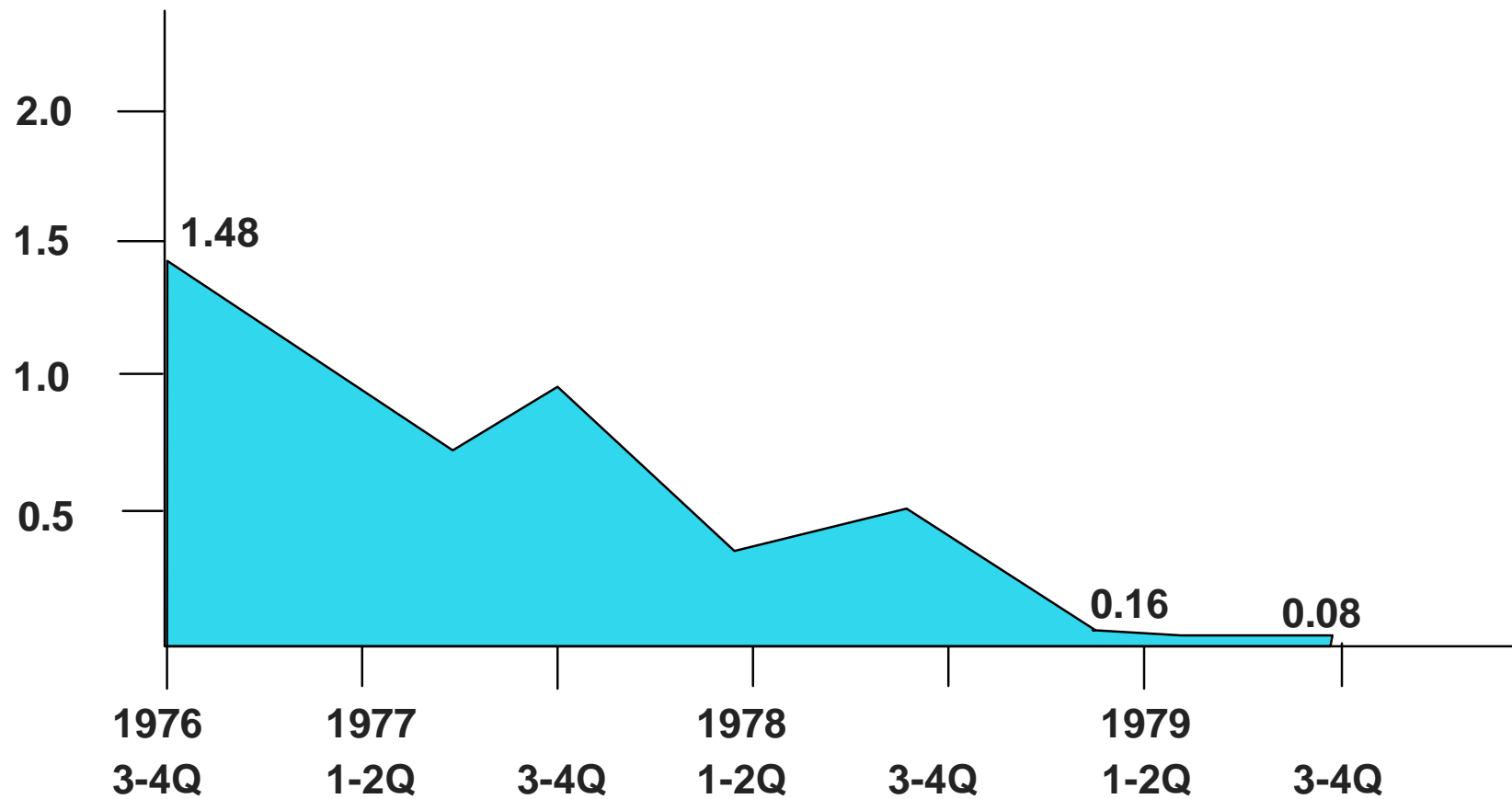


Figure 15.14



Process Improvement Results

- Computer Sciences Corporation
- Six Projects -2.7 Million lines of code
- Card, Clark and Berg, 1987

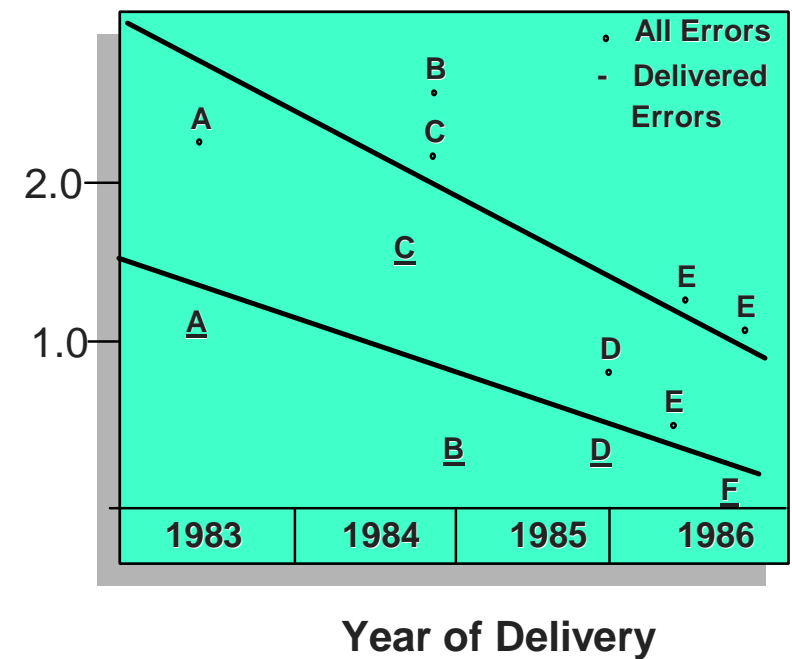
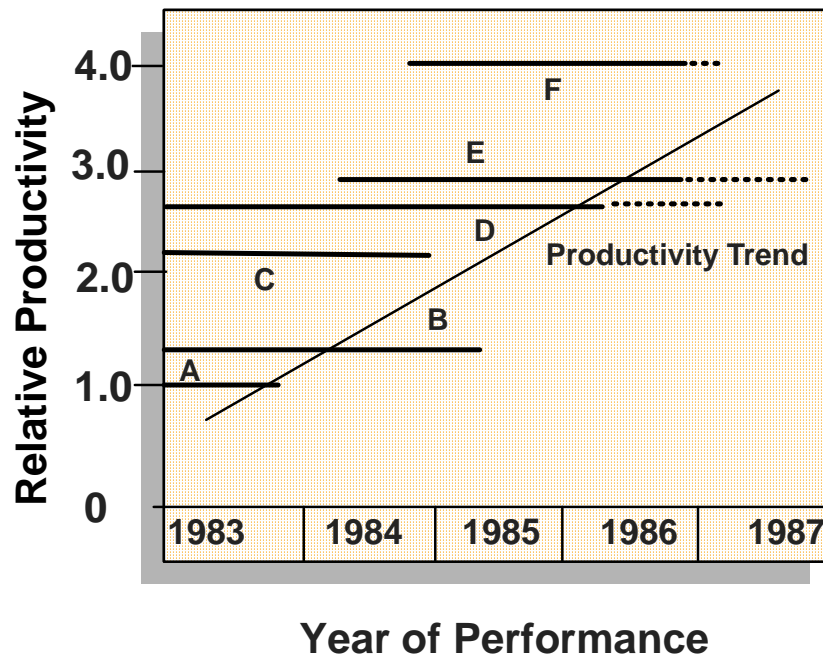
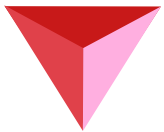


Figure 15.15



Product Structure Model

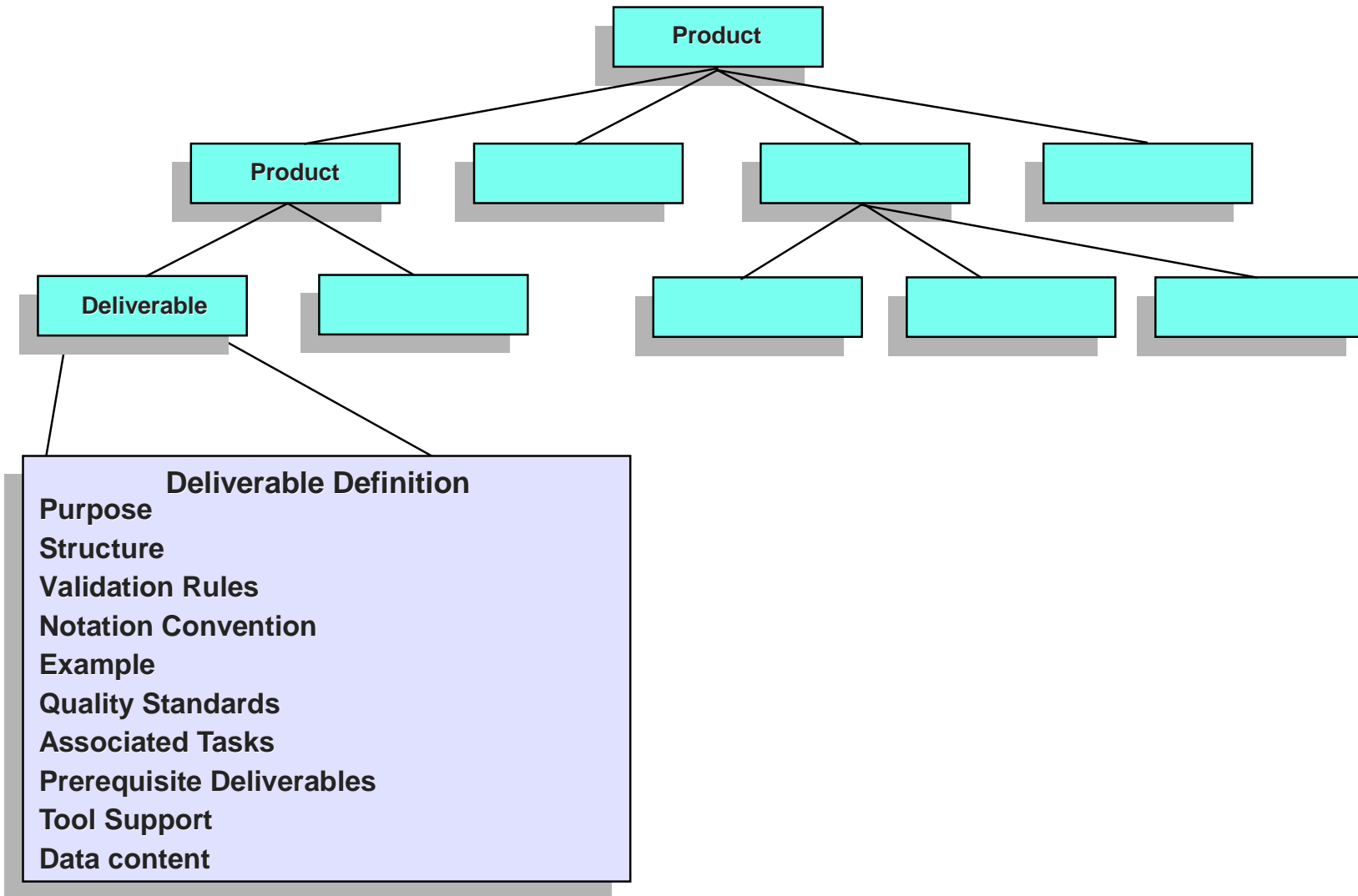


Figure 4.2