SOFTWARE QUALITY

CPTS 583

Software **Process** Quality Metrics

Outline

- Software Process Quality
  - Overview and scope
- Process Quality Metrics
  - Error density and severity metrics
  - Defect/error arrival and removal metrics
  - Process timing metrics
  - Process productivity metrics

Scope of process quality

- Process metrics
  - Measuring characteristics of *development process*
    - As opposed to its outputs
  - How errors and defects arise
  - How effectively errors and defects are removed
  - Process schedule and timing progress
  - How effective the process is

Why measure the process?

- The process creates the product
- If we can improve the process, we indirectly improve the product

- Through measurement, we can *understand*, *control* and *improve* the process
- This will lead to us engineering quality into the process rather than simply taking product quality measurements when the product is done

Process quality metrics: sneak peek

- Average find-fix cycle time
- Average amount of rework time
- Number of person-hours per inspection
- Number of person-hours per KLOC

Efforts and time costs

- Percentage of modules that were inspected

Scope of inspection

- Average number of defects found per inspection
- Number of defects found during inspections in each defect category

Inspection effectiveness

Error Density

- Number of errors per unit of system size
- Weighted total: weighted sum, with weight per error
- Straight total: simple sum

$$\frac{\#errors}{system\_size}$$

Weighted or straight total

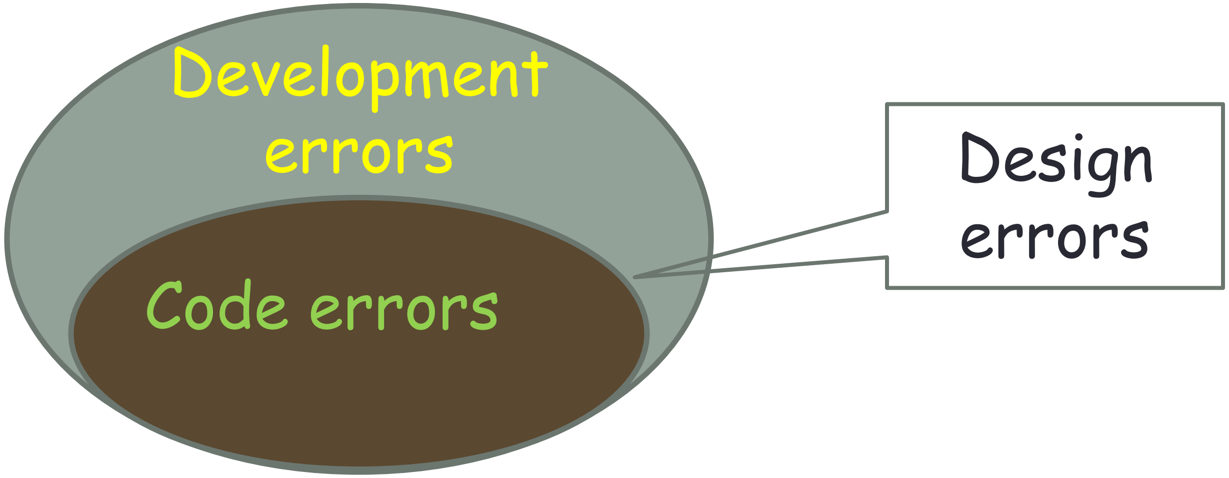
KLOC or NFP

weighting

Error severity class	Relative weight
Low severity	1
Medium severity	3
High severity	9

Error Density

- Code errors
  - Found in code during code inspection and testing
- Development errors
  - Found in design and code reviews, testing reports
  - Include code errors
- Measuring density of both kinds
  - With different metrics



Error Density

Abbrev.	Name	Calculation formula
CED	Code Error Density	$CED = \frac{NCE}{KLOC}$
DED	Development Error Density	$DED = \frac{NDE}{KLOC}$
WCED	Weighted Code Error Density	$WCDE = \frac{WCE}{KLOC}$
WDED	Weighted Development Error Density	$WDED = \frac{WDE}{KLOC}$
WCEF	Weighted Code Errors per Function Point	$WCEF = \frac{WCE}{NFP}$
WDEF	Weighted Development Errors per Function Point	$WDEF = \frac{WDE}{NFP}$

Error Density

- Example

Severity	#code errors	Weight
Low	30	1
Medium	40	3
High	10	9

Code size: 10 KLOC

- NCE = 30+40+10 = 80
- WCE = 30*1 + 40*3 + 10*9 = 240

- Code error density = 80 / 10 = 8
- Weighted code error density = 240 / 10 = 24

Error Severity

- Average severity of code errors

$$\frac{\textit{weighted number of code errors}}{\textit{total number of code errors}}$$

- Average severity of development errors

$$\frac{\textit{weighted number of development errors}}{\textit{total number of development errors}}$$

Decreasing error density is not necessarily great!

Error Severity

- Example

Severity	#code errors	Weight
Low	30	1
Medium	40	3
High	10	9

Severity	#code errors	Weight
Low	20	1
Medium	10	3
High	15	9

Code size: 10 KLOC

Weighted Error Density

(30*1+40*3+10*9)/10 = 24

(20*1+10*3+15*9)/10 = 18.5

Average error severity

(30*1+40*3+10*9)/80 = 3

(20*1+10*3+15*9)/45 = 4.1

Defect Density During Machine Testing

- Monitor/predict defect density in product
- vs defect density as product quality metrics
  - During testing phase, not after release (i.e., not in the field)

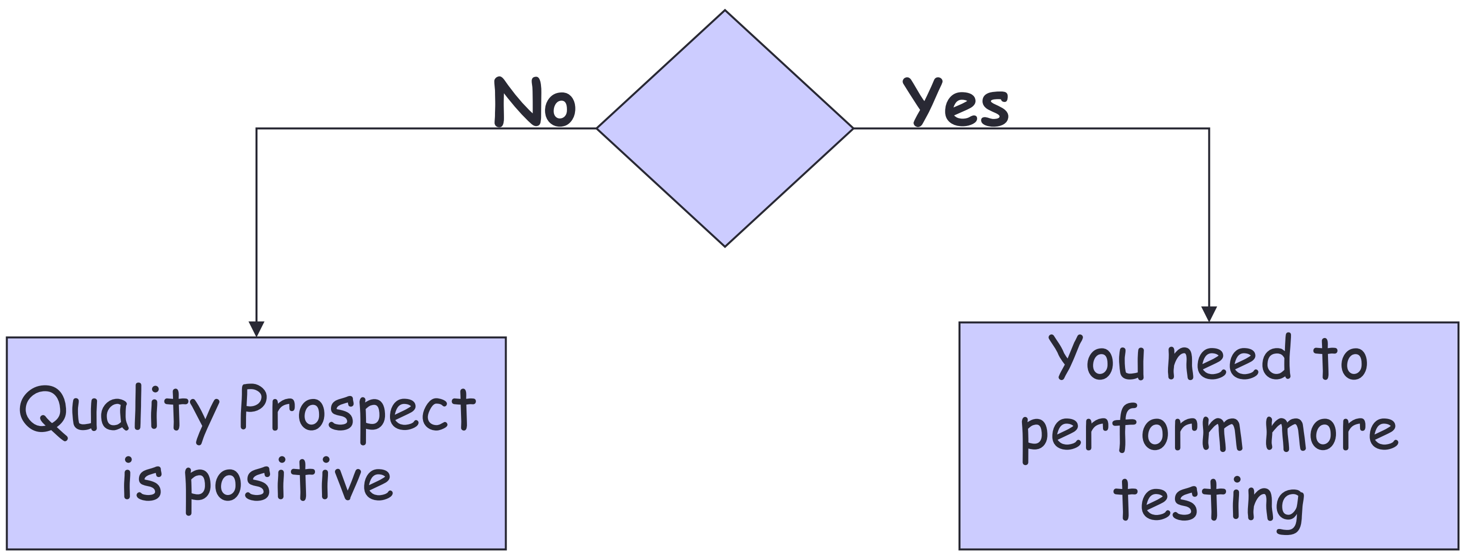
  - Same computation (#defects / system-size)
  - Positive correlation
    - High defect density in testing → higher defect density in product
    - Exceptions
      - Extraordinary testing effort invested
      - More effective testing method is employed

Defect Density During Machine Testing

- **Scenario 1:**

Defect density during testing is the same or lower than previous release.

Reasoning: Does the testing for the current release deteriorate?

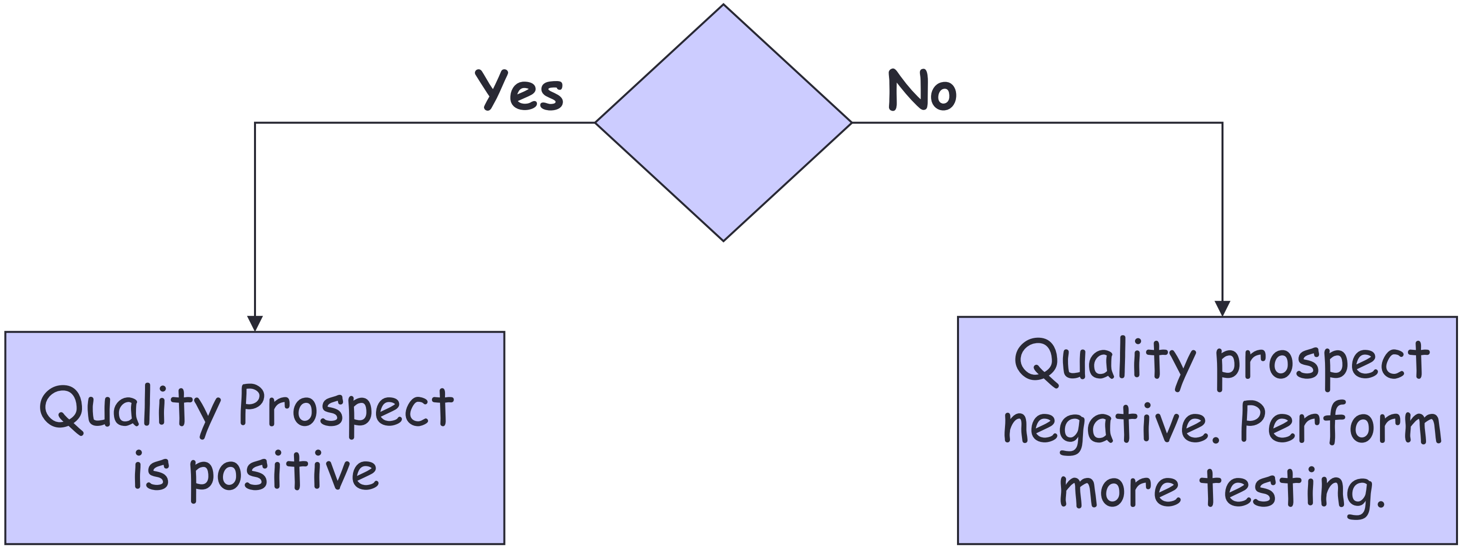


Defect Density During Machine Testing

- **Scenario 2:**

Defect density is substantially higher than that of the previous release.

Reasoning: Did we plan for and actually improve testing effectiveness?

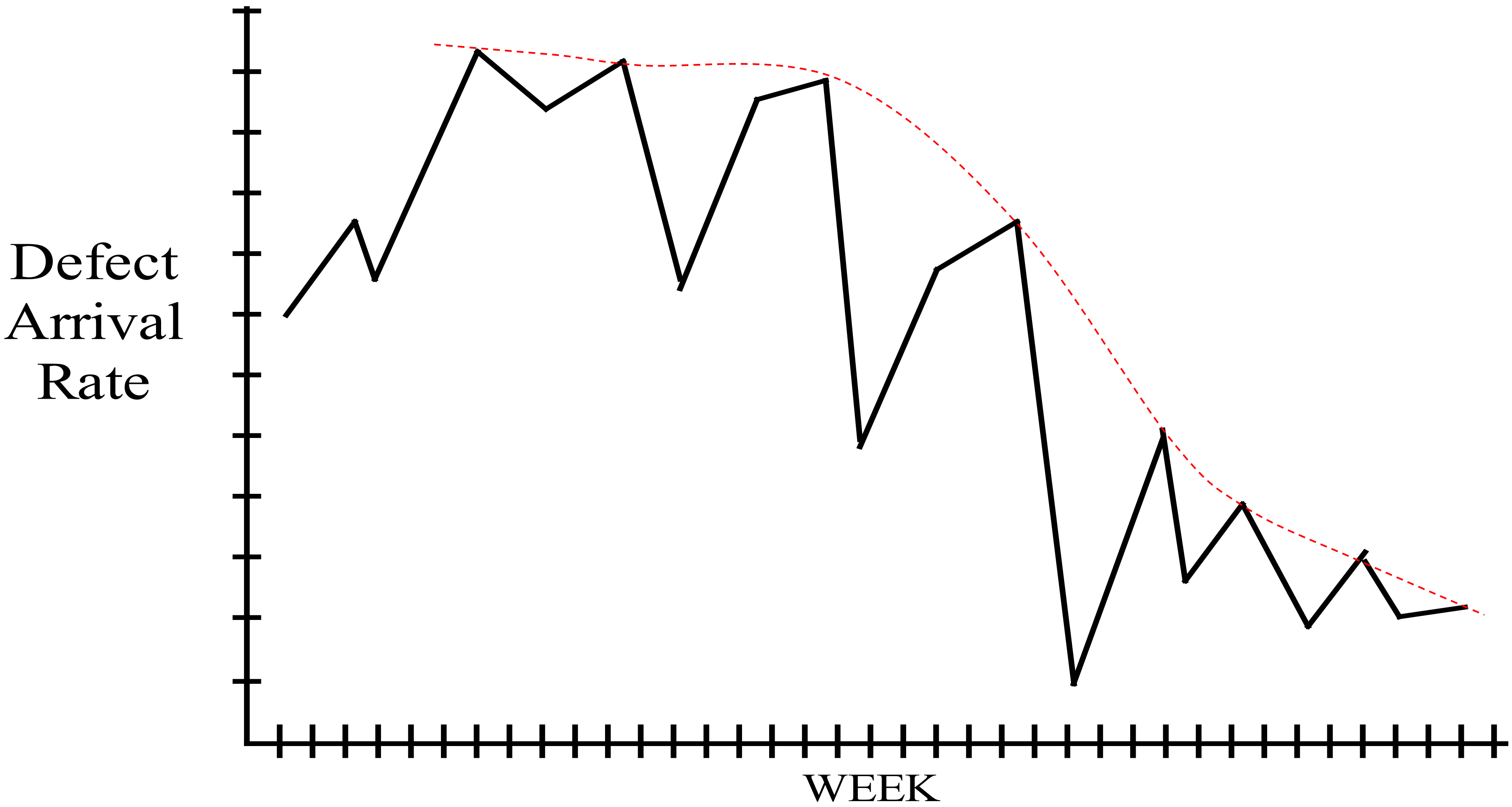


Defect Arrival Pattern During Testing

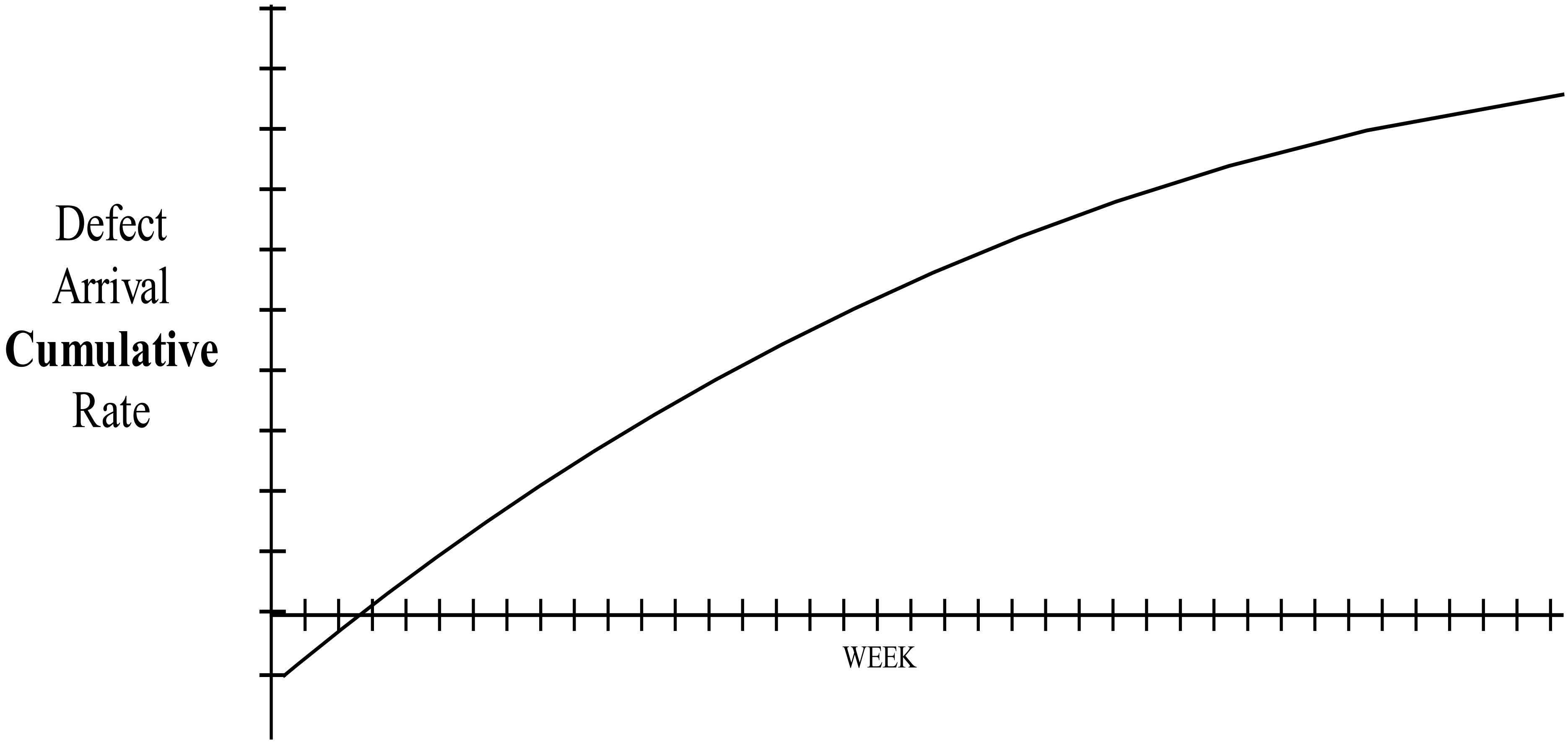
- Overall defect density during testing is a summary indicator
- However, the patter of defect arrivals gives more information

- Even with the same overall defect density during test, arrival patterns can be different
- Different defects arrival patterns indicate different quality levels in the field

Defect Arrival Pattern During Testing

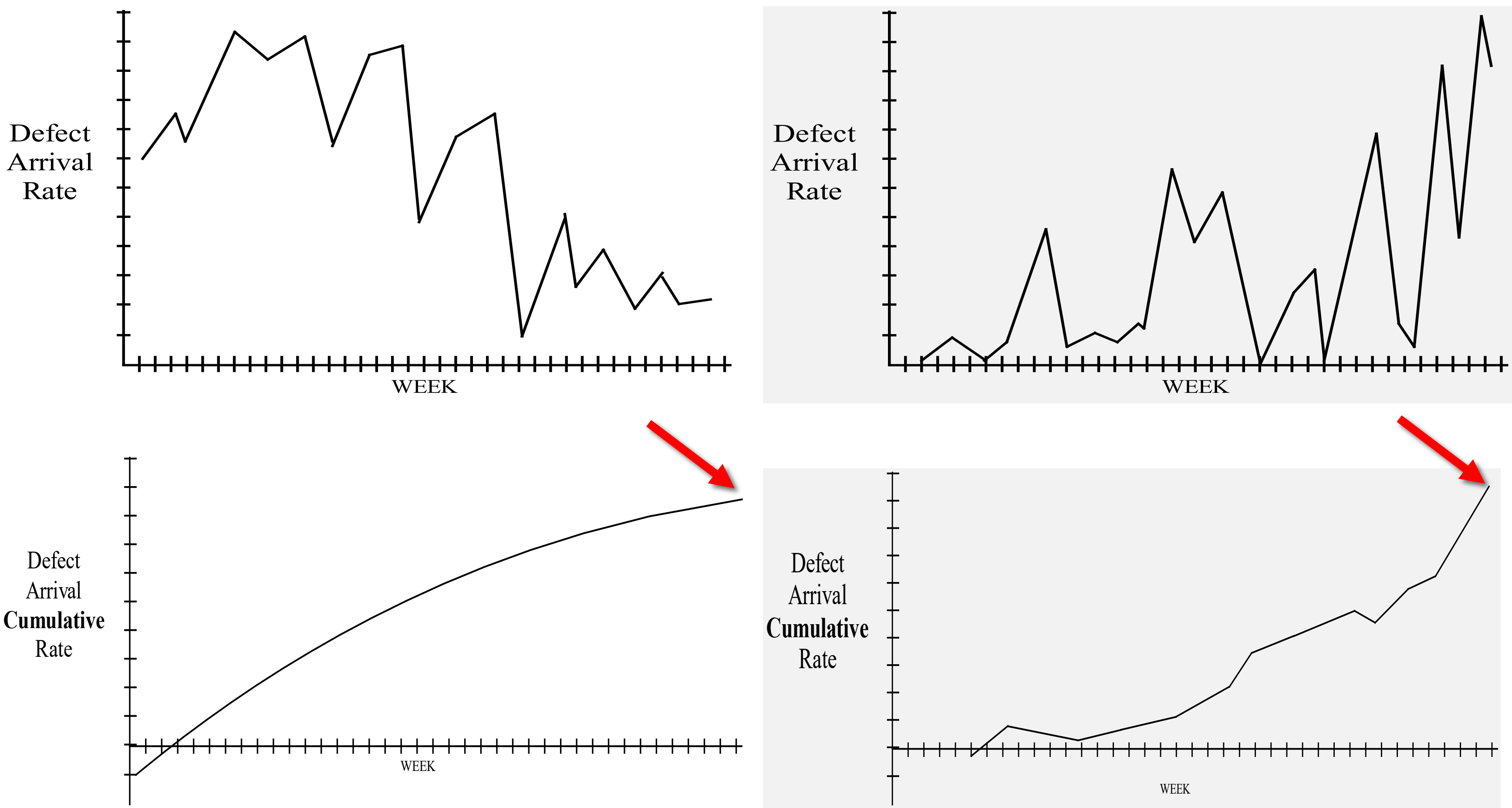


Defect Arrival Pattern During Testing



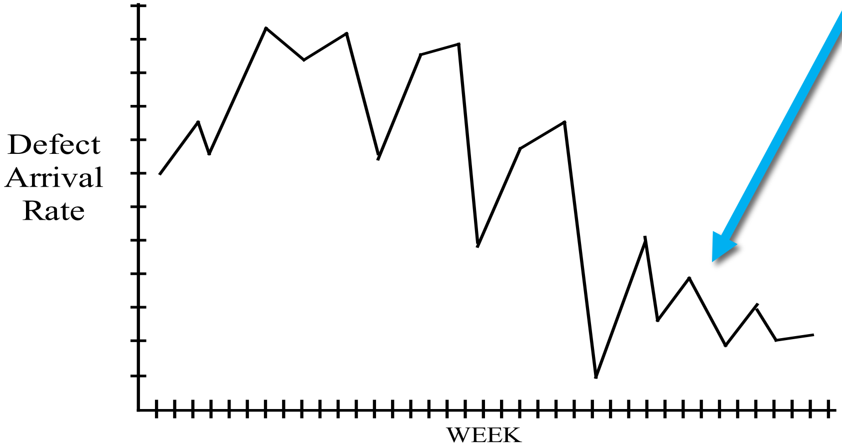
Defect Arrival Pattern During Testing

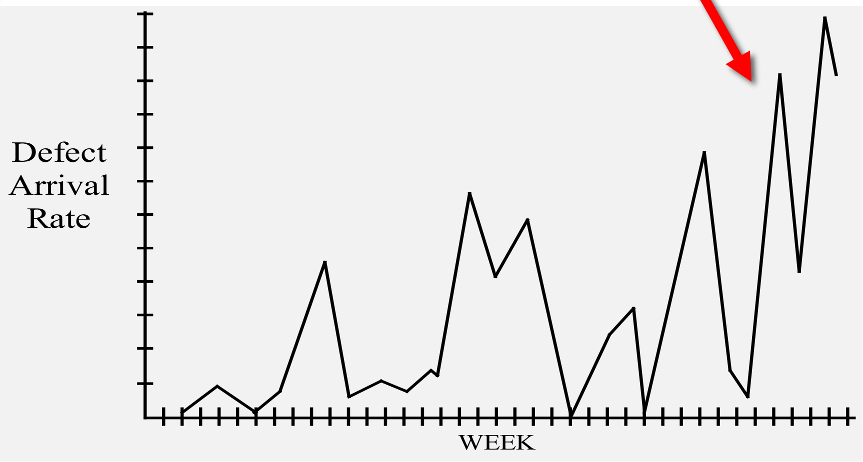
- Same defect density with different arrival pattern



Interpreting Defect Arrival Patterns

- Always look for defect arrivals stabilising at a very low level.
- If they do not stabilise at a low rate, the product will be very risky





Measuring Defect Arrival Patterns

- Raw defect arrivals
  - Number of raw defects occurred in a time interval during testing
- Valid defect arrivals
  - Only the defects that are determined as valid in a time interval
- Defect backlog over time
  - Record of defects that have not been addressed (and are to be investigated and fixed)

It is useless detecting defects if they are not fixed and the system re-tested.

Error Removal Effectiveness

- Development Errors Removal Effectiveness (DERE)

$$\frac{\#\ development\ errors}{\#\ development\ errors + \#\ system\ failures}$$

Number of failures detected during a year of maintenance service

Example:

	Project 1	Project 2
#development errors	80	120
#system failures	20	25
DERE	0.8	0.83

Error Removal Effectiveness

- Development Weighted Errors Removal Effectiveness (DWERE)

$$\frac{\text{weighted development errors}}{\text{weighted development errors} + \text{weighted system failures}}$$

Weighted sum of failures detected during a year of maintenance service

Defect Removal Effectiveness

- DRE =

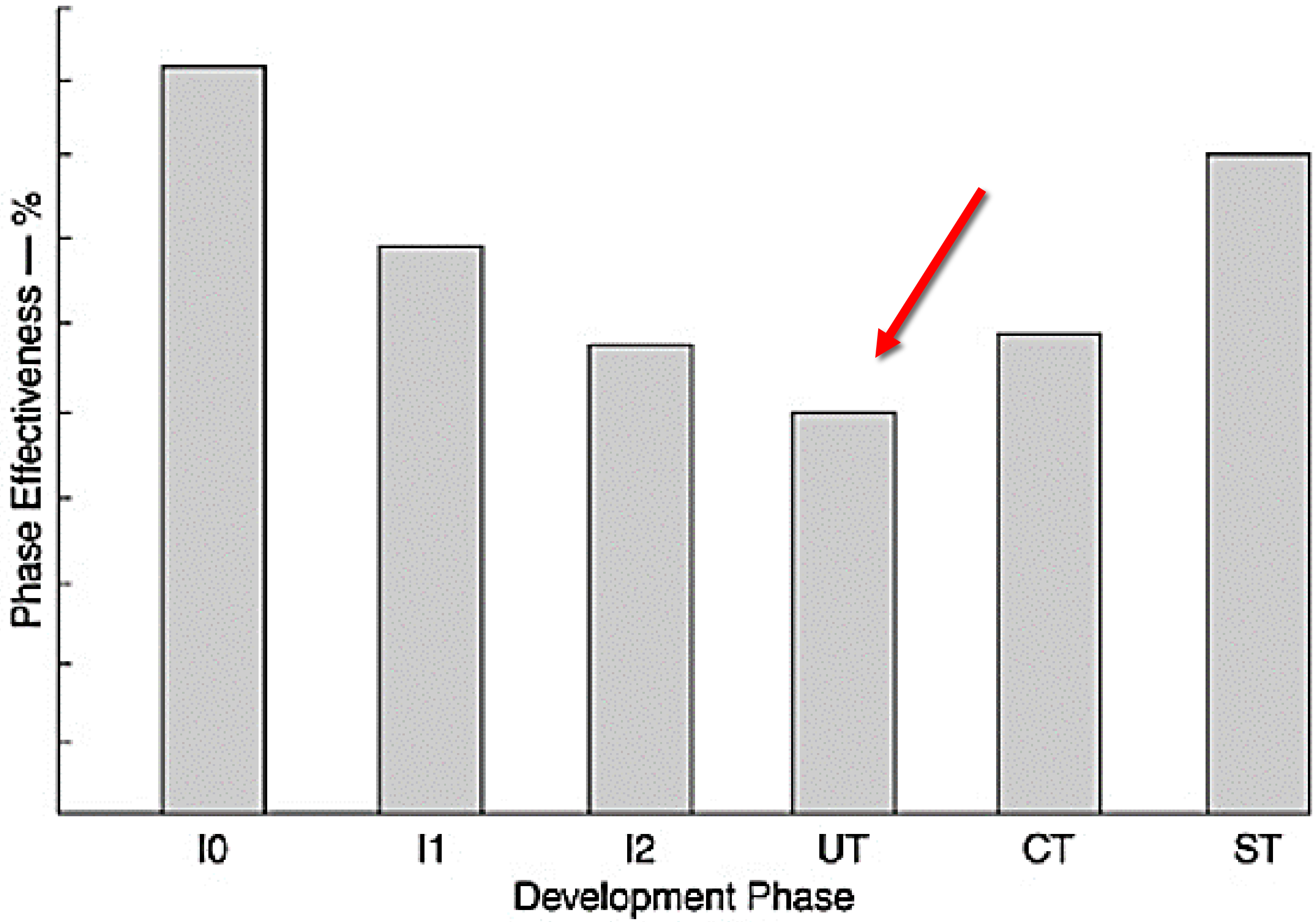
$$\frac{\#defects\ removed}{\#defects\ removed + \ \#defects\ detected\ later}$$

\# latent defects in product

- DRE of a particular phase
- DRE of an entire development process

Defect Removal Effectiveness

- Example: DRE for individual phases



Processing timing metric

- Timing observance

$$\frac{\#milestones\ completed\ on\ time}{total\ number\ of\ milestones}$$

- Average Milestone Delay

$$\frac{sum\ of\ milestones\ completion\ delays}{total\ number\ of\ milestones}$$

Processing timing metric

- Real Examples

- Eight milestones in the project (during the process)
- Four finished 0, 1, 2, 3 days before deadline
- Four delayed by 1, 2, 3, 4 days

- Timing observance = 4 / 8 = 0.5
- Average milestone completion delay = (1+2+3+4)/8 = 1.25 (days)
  - Alternatively: (-1 + -2 + -3 + 1 + 2 +3 +4) / 8 = 0.5 (days)

Measuring Process Productivity

- Development productivity

$$\frac{\textit{total hours invested for development}}{\textit{system size}}$$

KLOC or NFP

- Example
  - 100 hours spent on developing a system of 10 KLOC
  - Metric = 100 / 10 = 10

Measuring Process Productivity

- Code Reuse

$$\frac{\text{number of thousands of reused lines of code}}{\textit{system size in KLOC}}$$

- Documentation reuse

$$\frac{\textit{number of reused documentation pages}}{\textit{number of total documentation pages}}$$

Other useful process metrics

- Fix response time
  - Average time to fix a defect
- Percent delinquent fixes
  - Fixes which exceed the recommended fix time according to their severity level
- Fix quality
  - Percentage of fixes which turn out to be defective

Summary

- Scope of process quality measurement: *defect/error existence, arrival, removal, timing, productivity*
- Presence of errors and defects: error density/severity, defects density during machine testing,
- Arrival patterns of defects: raw arrivals, valid arrivals, defects backlog over time
- Removal of errors/defects: removal effectiveness
- Timing and productivity of process: timing observance, average completion delay, development productivity, code/documentation reuse