

# PROCESS MANAGEMENT

Wednesday, April 26,  
2017

By Saman Gunawardena

# Objectives

2

- Understand process management activities
- Defining the process
- Process measurement
- Process evaluation
- Process maturity

# ICE BREAKER

3

- Ask students to write the process of how to study for an exam
- Time estimate 30 minutes
- De Brief

# Definition of a Process

4

- A process is a vehicle of communication, specifying the methods used to produce a product or service. It is the set of activities that represent the way work is to be performed. The level of communication (detail of the process) is normally commensurate with the skill level associated.

# Why Processes Are Needed (Management Perspective)

5

From a *management perspective*, processes are needed to:

- Explain to workers how to perform work tasks
- Transfer knowledge from more experienced to less experienced workers
- Assure predictability of work activities so that approximately the same deliverables will be produced with the same resources each time the process is followed
- Establish a basic set of work tasks that can be continuously improved
- Provide a means for involving workers in improving quality, productivity, and customer satisfaction by having workers define and improve their own work processes
- Free management from activities associated with "expediting work products" to spend more time on activities such as planning, and customer and vendor interaction

# Why Processes Are Needed (Worker Perspective)

6

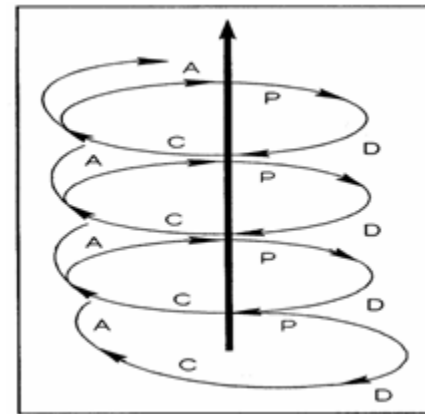
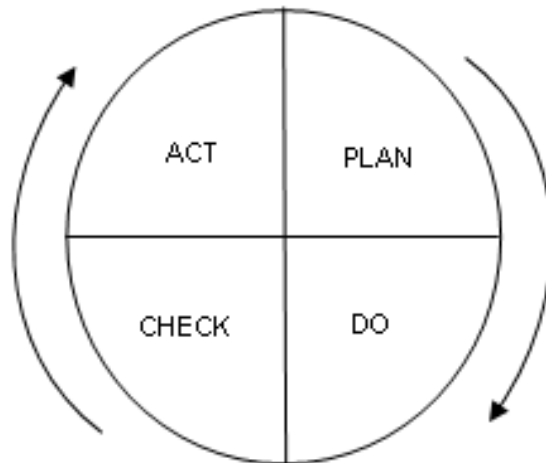
From a *worker perspective*, work processes are important to:

- Increase the probability that the deliverables produced will be the desired deliverables
- Put workers in charge of their own destiny because they know the standards by which their work products will be evaluated
- Enable workers to devote their creativity to improving the business instead of having to develop work processes to build products
- Enable workers to better plan their workday because of the predictability resulting from work processes

# PDCA Cycle

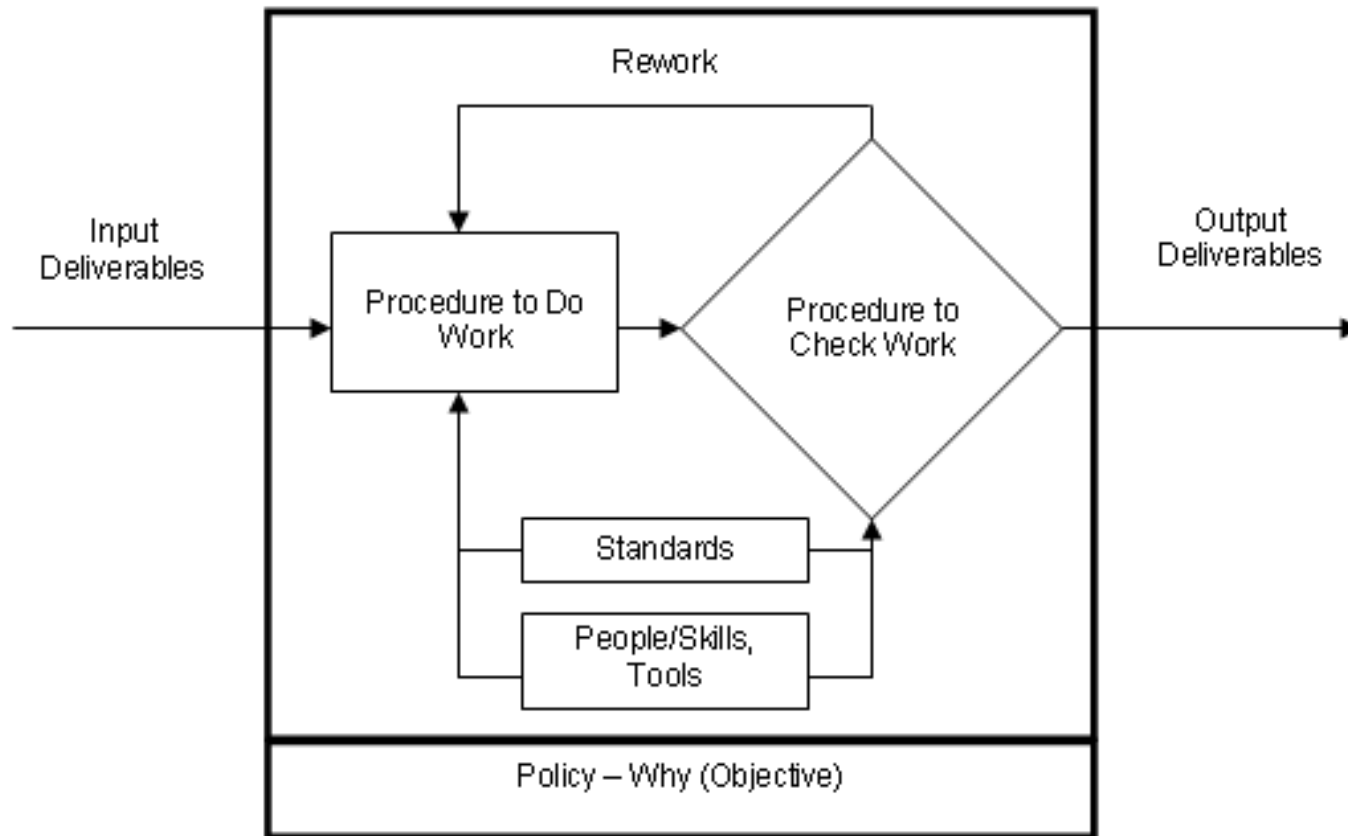
7

- A major premise of a quality management environment is an emphasis on continuous improvement. The approach to continuous improvement is best illustrated using the PDCA cycle, which was developed in the 1930s by Dr. Shewhart of the Bell System.



# Process Workbench and Components

8





# Process Categories

9

- Management Processes
- Work Processes
- Check Processes

# Nokia – Case Study

10

## 5 REASONS WHY NOKIA FAILED

- ❑ NOKIA MOVED TOO SLOWLY
- ❑ ANDROID PAID OFF (FOR SAMSUNG) AND WINDOWS PHONE HASN'T ... YET (FOR NOKIA)
- ❑ HURTING ON BOTH ENDS
- ❑ NOKIA DIDN'T HAVE THE PANACHE
- ❑ EXECUTION IS KEY

**Discuss why check process is important**

# Tutorial – Exercises

11

- Go back to the exercise given at the beginning to writing the process how to study for an exam and ask students re-write the process with the learning from the lecture

# Process Measurement - Beginning

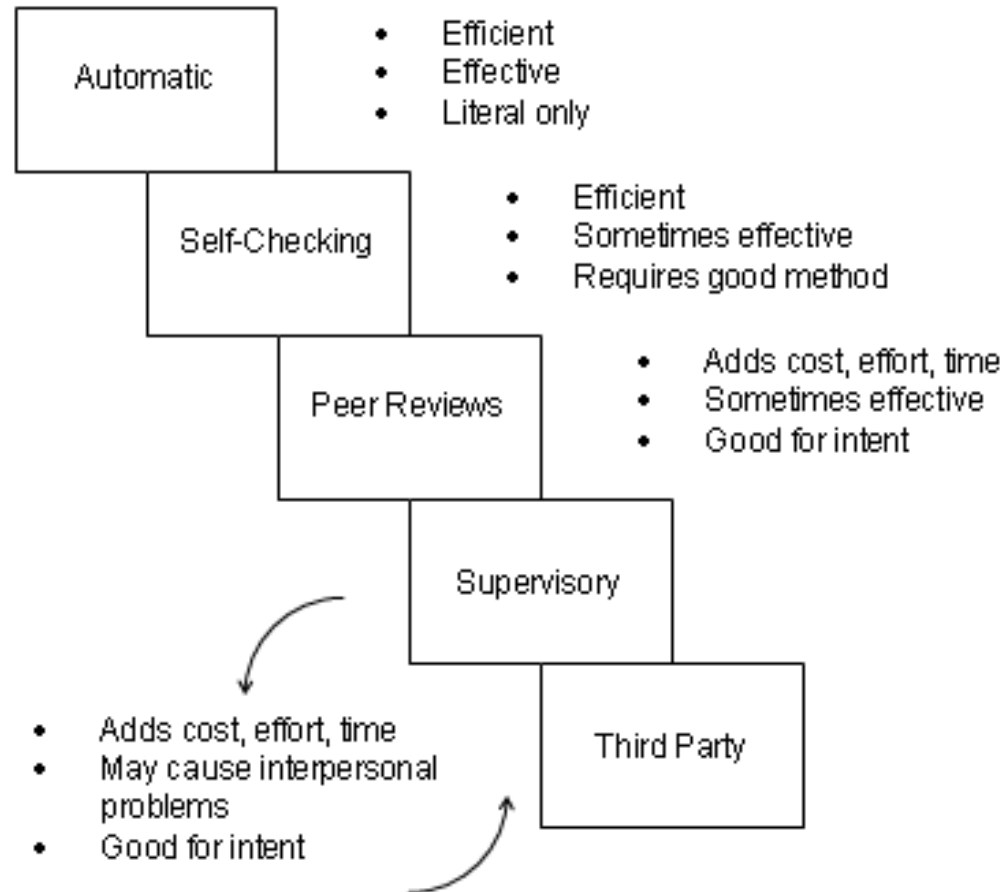
12

**Identify Control Points :** Controls are often placed near the end of a process or workbench, but this is not always the most appropriate location. The first step in process control design is to identify the most logical points in the process to add controls. One way to address this issue is to identify and rank process risks. Process risks are those things that could occur during execution of a process that would result in defects and rework. For example, process risks for an "estimate project effort" process may be:

- ❑ Use of inaccurate historical data
- ❑ Misuse of historical data
- ❑ Inaccurate estimation algorithm or mathematical mistakes
- ❑ Inadequate contingencies used
- ❑ Wrong staffing ratios and loading figures used

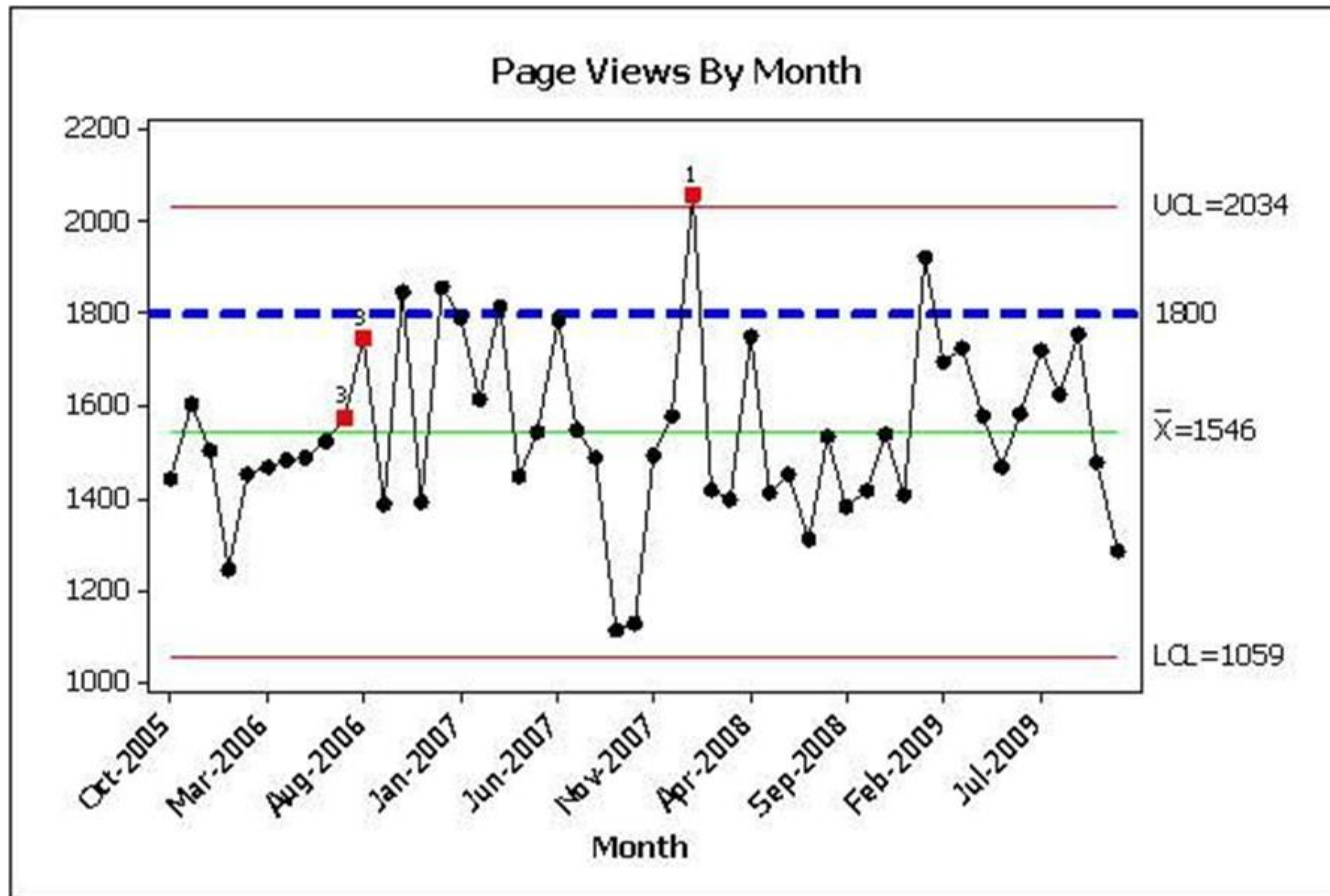
# Check Process – Process Measurement

13



# Control Charts

14



# Process Metrics – Example 1

15


## Example – Project Metrics

Metric	Description
Scope Change Requests	# of scope changes requested by the client or sponsor
Scope Change Approvals	# of scope changes that were approved
Overdue tasks	# of tasks that were started but not finished on time
Tasks	# of task that should have started but have been delayed
Over budgeted tasks	# of tasks that have cost more to complete than expected
Earned Value	Budgeted Cost of Work Performed (BCWP)
Over allocated Resources	# of resources assigned to more than one task.
Turnover	# of project team members who quit or terminated.
Training Hours	# of training hours per project team member.



# Process Metrics – Example 2

16



**Metrics for Third Party Application Development and Maintenance**

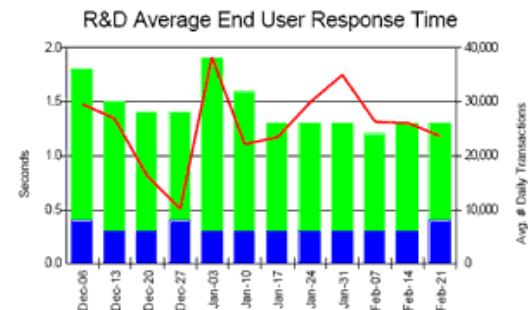
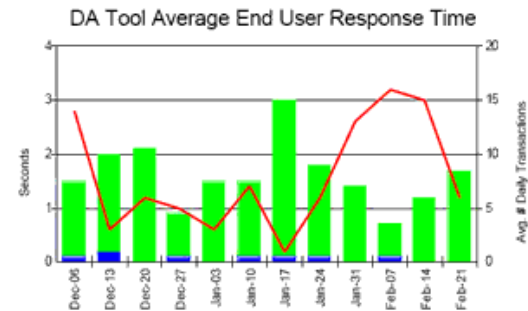
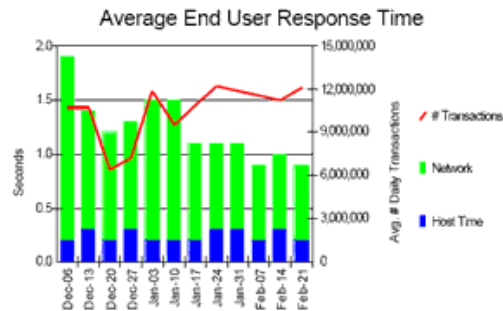
Category	Metric Name	Description	Waterfall		Agile		RESULTS			
			Type	Frequency	Type	Frequency	Q1	Q2	Q3	Q4
QUALITY	Post-Release Defect Rate	Measures the number of defects found after release	Per Thousand Effective Lines of Code (KELOC)	Monthly, after release	Per Thousand Effective Lines of Code (KELOC)	Monthly, after release				
	Robustness	Number of hangs or crashes after release	Count (Hangs or Crashes)	Monthly, after release	Count (Hangs or Crashes)	Monthly, after release				
	Module Size	The size of each module in a program	Count (Effective Lines of Code)	Monthly	Count (Effective Lines of Code)	Monthly				
	Complexity	Measures the number of methods that have more than 15 linearly independant paths through a program compared to the total number of methods	Percentage of total methods	Monthly	Percentage of total methods	Monthly				
	Mean Time Between Failures (MTBF)	Measures the expected time between two failures	Count (Hours)	End of project	Count (Hours)	End of Project				
	Regression	The number of defects that were fixed but have crept back into the code	Per Thousand Effective Lines of Code (KELOC)	Monthly	Per Thousand Effective Lines of Code (KELOC)	Monthly				
	Requirements Coverage	Compares the number of requirements generated with the number of requirements completed	Percentage of total requirements	End of project	Percentage of total new requirements	Per Release				
	Outstanding Defects	Measures the difference between the number of reported defects and resolved defects in each phase	Percentage of total defects	Per testing phase	Percentage of total defects	Per Release				



# Process Metrics – Example 3

17

## System Management Center - System Response Time



System Monitoring Center [SMC SRT (Cars,IW, M&D, DATool) ]

Page: 20

36 of 39  
<http://www.e-janco.com>

# Process Evaluation & Improvement

18

## **Process Improvement Process**

1. Select process and team
2. Describe current process
3. Assess process for control and capability
4. Brainstorm for improvement
5. Plan how to test proposed improvement
6. Analyze results
7. Compare results
8. Change process or redo steps 4-8

# Tutorial – Exercises

19

- Go back to the exercise given at the beginning to writing the process how to study for an exam and ask students re-write the process with further learning from the lecture

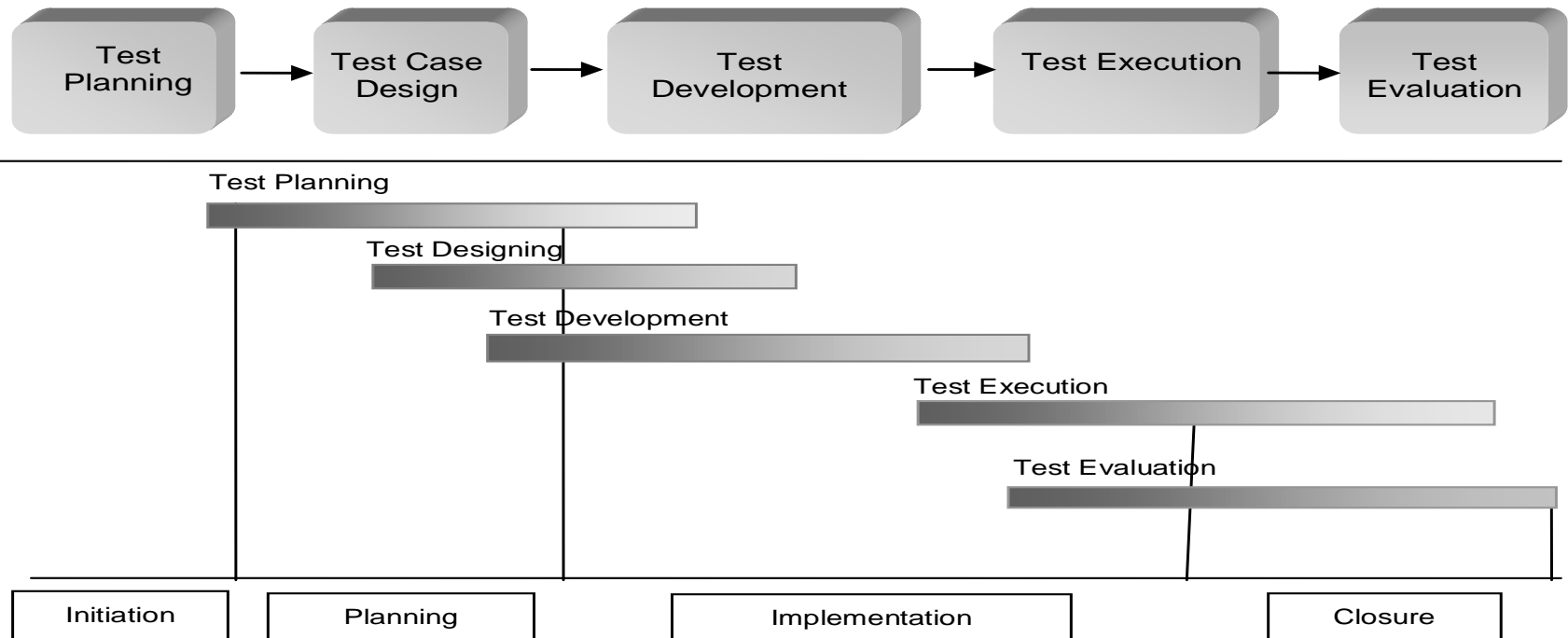
# TEST PHASES

Wednesday, April 26,  
2017

By Saman Gunawardena

# Software Test Phases

21



# Test Planning

22

- Test Planning (Already Covered). Lewis (2000, p. 119) says that the test plan is an on-going document which:
  - Has a good chance of detecting majority of defects
  - Provides a good test coverage
  - Is flexible
  - Is repeatable and executed easily and automatically
  - Defines the types of tests to be performed
  - Clearly documents the expected results
  - Provides defect management
  - Clearly defines test objectives
  - Clarifies test strategy
  - Clearly defines the test exit criteria
  - Is not redundant
  - Identifies the risks
  - Documents the test requirements
  - Defines the test deliverables

# Test Design

23

Steps	Tasks
Design Function Tests	Refine Functional Test Requirements
	Build Function / Test Matrix
Design GUI Tests	Define Application GUI Components
	Design GUI Tests
Define System/Acceptance Tests	Identify Potential System Tests
	Design System Fragment Tests
	Identify Potential Acceptance Tests
Review Approve Design	Schedule/Prepare for Review
	Obtain Approvals

# Test Development

24

Steps	Tasks
Develop Test Scripts	Script GUI/Function Tests
	Script System Fragment Tests
Review Approve Design	Schedule/Prepare for Review
	Obtain Approvals



# Test Execution & Evaluation

25

## □ Test Execution

- ▣ Execute the planned tests as per the test types being identified and scheduled
- ▣ Track and manage defects (Defect Management)

## □ Test Evaluation

- ▣ Analyse Metrics
- ▣ Refine Test Schedule
- ▣ Identify Requirement Changes

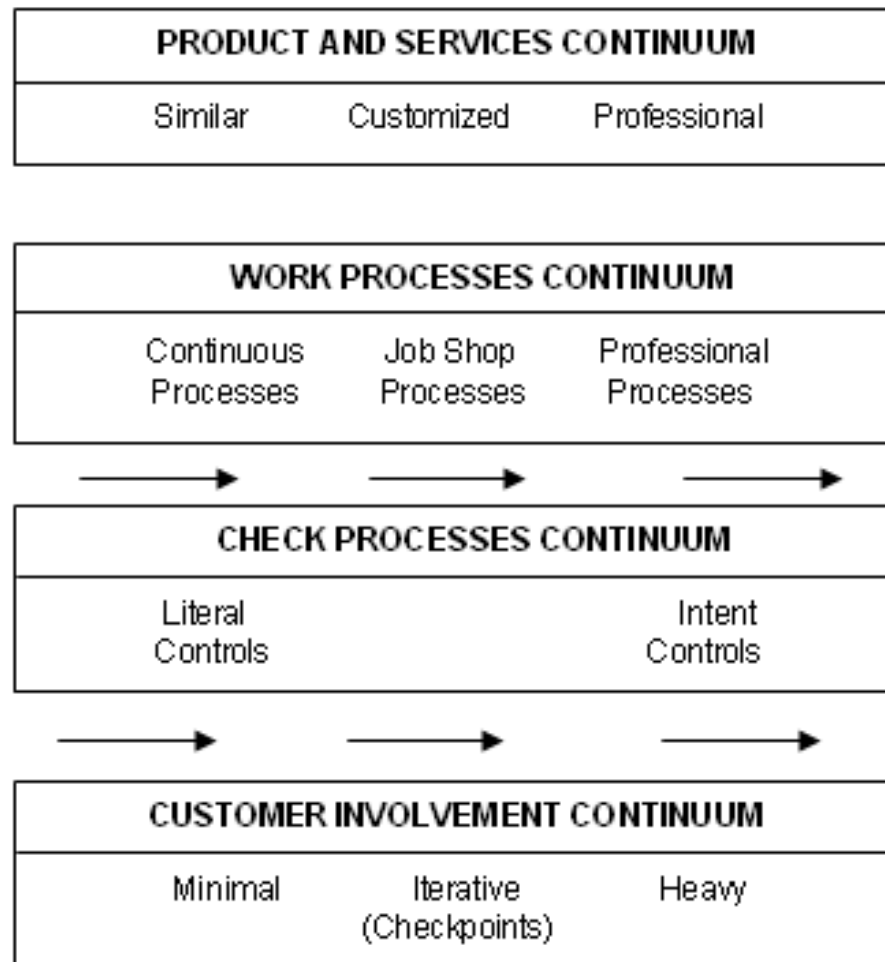
# PROCESS MATURITY

Wednesday, April 26,  
2017

By Saman Gunawardena

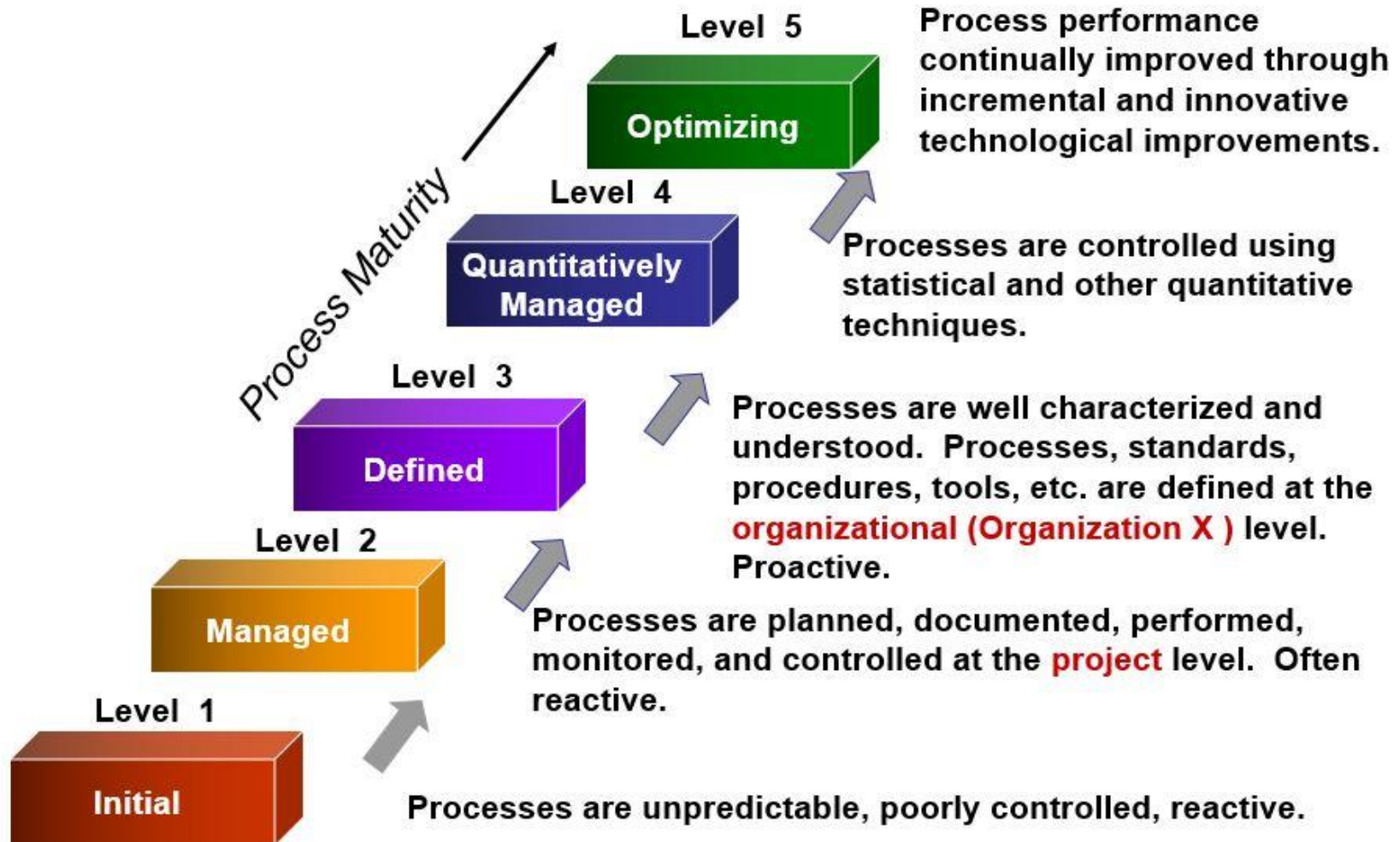
# The Process Management Continuum

27



# CMMI – Capability Maturity Model Integration

28



# Tutorial – Exercises

29

- Go back to the exercise given at the beginning to writing the process how to study for an exam and ask students re-write the process with the learning from the lecture

# Exercise

30

- Define a Bug Tracking and Management Process

# INCIDENT AND BUG MANAGEMENT

Wednesday, April 26,  
2017

By Saman Gunawardena

# Objective of Defect Management

- ❑ Provide developers and other parties with feedback about the problem to enable identification, isolation and correction as necessary.
- ❑ Provide test leaders a means of tracking the quality of the system under test and the progress of the testing.
- ❑ Provide ideas for test process improvement.





# What do Software Defects Cost?

## The Intel Pentium Bug

Enter the following in your PC calculator:  $(4195835/3145727)*3145727-4195835$ . If the answer is not zero, you have an old Intel Pentium CPU with the floating-point division bug

- ❑ Discovered in 1994
- ❑ Intel's test engineers had found the problem before release.
- ❑ When pressured, Intel offered to replace the faulty chip, but only if a user could prove that he was affected
- ❑ Ultimately, there was a public outcry. Intel had to spend \$400 mn to replace the faulty chip

## Software faults can cause death and injury also

- ❑ Radiation treatment kills patients due to UI misinformation w.r.t sensitivity between Gamma and X-Ray radiations.
- ❑ Aircraft crashes due to failure of Auto-Pilot in the calculation of ground proximity signals

# How a defect/bug is produced?



Failure: Deviation of the Software from its expected delivery or service

Compared with specification or desired use/functionality



# Fault



A programmer makes a **mistake**.

The mistake manifests itself as a **fault** [or defect] in the program.

A **failure** is observed if the fault [or defect] is made visible. Other faults remain **latent** in the code until they are observed (if ever).

**Figure :** The progression of a software failure. A purpose of testing is to expose as many failures as possible before delivering the code to customers.

**Mistake/Error:** a human action that produces an incorrect result

**Fault:** An incorrect step, process, or data definition in a program. Also known as Defect or Bug

# Why do Software have faults?

- **Software is written by human beings**
  - ▣ Who knows something, but not everything
  - ▣ Who have skills, but aren't perfect
  - ▣ Who do make mistakes (Errors)
- **Pressure to deliver on strict deadlines**
  - ▣ No time to check the assumptions made which may be wrong
  - ▣ Systems may be incomplete
- **Inexperience of the software professionals and complexity of design**
  - ▣ Professionals new to coding
  - ▣ Professionals new to the technology used in spite of having experience
- **Poor Communication**

# Who and when can we report a defect?

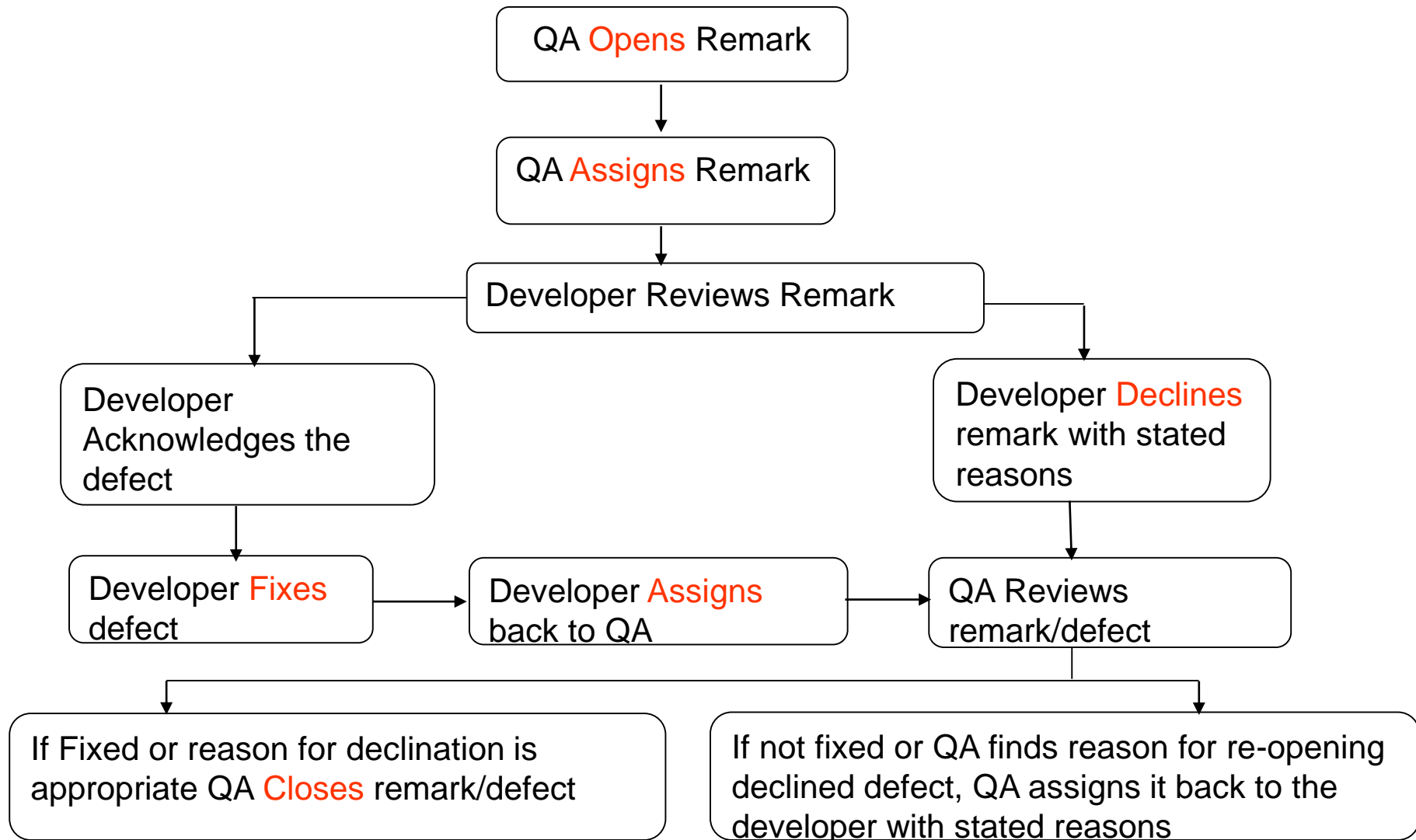
## Who?

- Anyone who finds that the software is not working as designed can report a defect
  - Testers/QA Personnel
  - Developers
  - Technical Support
  - BA
  - Beta sites
  - End users
  - Sales and Marketing staff (those interacting with the customers)

## When?

- When a difference is found between Expected and Actual Results
- When in doubt, write it up as a remark

# Defect Tracking (Defect Life Cycle)



# Defect Life Cycle (Cont....)

- By default, all defects will be created as 'Remark'. The Tech Lead/Project Manager will "Open" the remark for review, a valid remark will then become a defect which is assigned to a Software Engineer. The status of the defect becomes 'Open/Assigned'
- The Software Engineer will analyze and reproduce the defect. The status of the defect is then changed to Fixed or Not Reproducible or Not a defect etc depending on the findings of the developer
- The respective QA Engineer will retest to verify the fix and change the status to 'Verified Fixed' if verified. On the other hand, if the fix is found to be unsuccessful, then it will be re-opened

# Defect Life Cycle (Cont...)

- If the 'Remark' is declined it is reassigned to the person who has reported it with any the following status assigned to the remark
  - ▣ 'Duplicate' if the remark already exists
  - ▣ 'As Designed' if the remark is as per the requirements
  - ▣ 'Deferred' if it is not possible to fix the defect in the current release
  - ▣ 'Not Reproducible' if the developer is not able to reproduce the remark
- Used to describe and quantify deviations from requirements.
- Defects are reported to:
  - ▣ Correct the defect
  - ▣ Avoid the cost incurred in post-release phase
  - ▣ Report the status of the application
  - ▣ Gather statistics used to develop defect expectations in future applications
  - ▣ Improve the software development process

# Contents Of Defect

- Test ID#
- Test Environment
  - Document any operating system and browser name and version that pertains to the defect along with all the configuration settings
- Software under test ID
- Actual & Expected results
- Severity, scope, priority
- Status of the defect (e.g. open, deferred, duplicate, fixed, closed etc.)
- Other relevant information like Summary/Title
  - One sentence keyword enough to understand the defect
    - For e.g., “Fatal error when printing landscape”



# Contents Of Defect (Cont...)

☐ Date Submitted

☐ Submitted By

☐ **Description and Steps to Reproduce** (Numbered sequence of events to follow to recreate defect)

☐ Build Information

☐ Product Name

☐ Build Number

☐ Build Date

☐ Install Type

# Defect Severity

- Impact of the failure caused by this defect
  - Critical – Involves data loss, data corruption or whole system failure. Customer impact could be devastating
  - High – Major functionality is missing in the key component of the product. Some kind of workaround usually exists. Customer impact would be significant
  - Medium – Minor functionality issues. Easily reproducible workaround available
  - Low – Cosmetic errors which does not have any impact on the functionality

# Defect Priority

- Priority determines the order defects get fixed
  - ▣ Urgency to fix the defect
    - High – This has major impact on the customer. This must be fixed immediately
    - Medium – This problem should be fixed before the release of the current version
    - Low – This has to be fixed if there is time or can be deferred to the next release

# Examples



Think of the following type of defects:

- Example 1 : A spelling error on a user-interface screen. What severity/priority does this issue deserve?
- Example 2 : The anomalous server crash. We've all seen this type of issue. A server crash that occurs on the first full moon of every leap year

# Defect/Bug Reporting

- Compare actual Outcome with expected outcome. Log discrepancies accordingly based on the defect type
- At a high level defect can be identified in 3 main phases of software life cycle:
  - ▣ Design Phase
    - Specifications Incorrect
    - Documentation not proper
    - Design not correct
  - ▣ Implementation/Coding Phase
    - Database fault
    - Coding Fault
    - Suggestion
  - ▣ Maintenance Phase
    - Build/Package
    - Installation
    - Environment
- Additional Notes
  - ▣ Any document that helps investigate a defect
  - ▣ Screen shots

# TEST PROGRESS MONITORING AND CONTROL

Wednesday, April 26,  
2017

By Saman Gunawardena

# Agenda



- Test Monitoring and Test control
- Test Progress Examples
- Defect density and Failure rate
- Test control
- Test Reporting
- Entry and Exit criteria

# Test Monitoring and Test control

- Test Monitoring and Test Control come under Test management activities in addition to the estimations.
  - At any time we should know how well we are doing and understand what controlling actions we can take to keep testing on target.
  - Once we have started testing we must monitor our test progress and take corrective action should things go wrong.
  - Effective monitoring and control is vital in the test management process.



# Test Monitoring

- What is Test Monitoring ?

Test monitoring is a process of monitoring the test activities to give feedback and visibility about test activities.

- Why is Test Monitoring required ?

Once we have started testing we must monitor our test progress and take corrective action should things go wrong. Effective monitoring and control is vital in the test management process

- How do we do test Monitoring

Recording the number of tests run against the number of tests passed and the number of tests planned is one good way to show test progress.

# Measuring Test Execution progress 1

*tests planned*

*tests run*

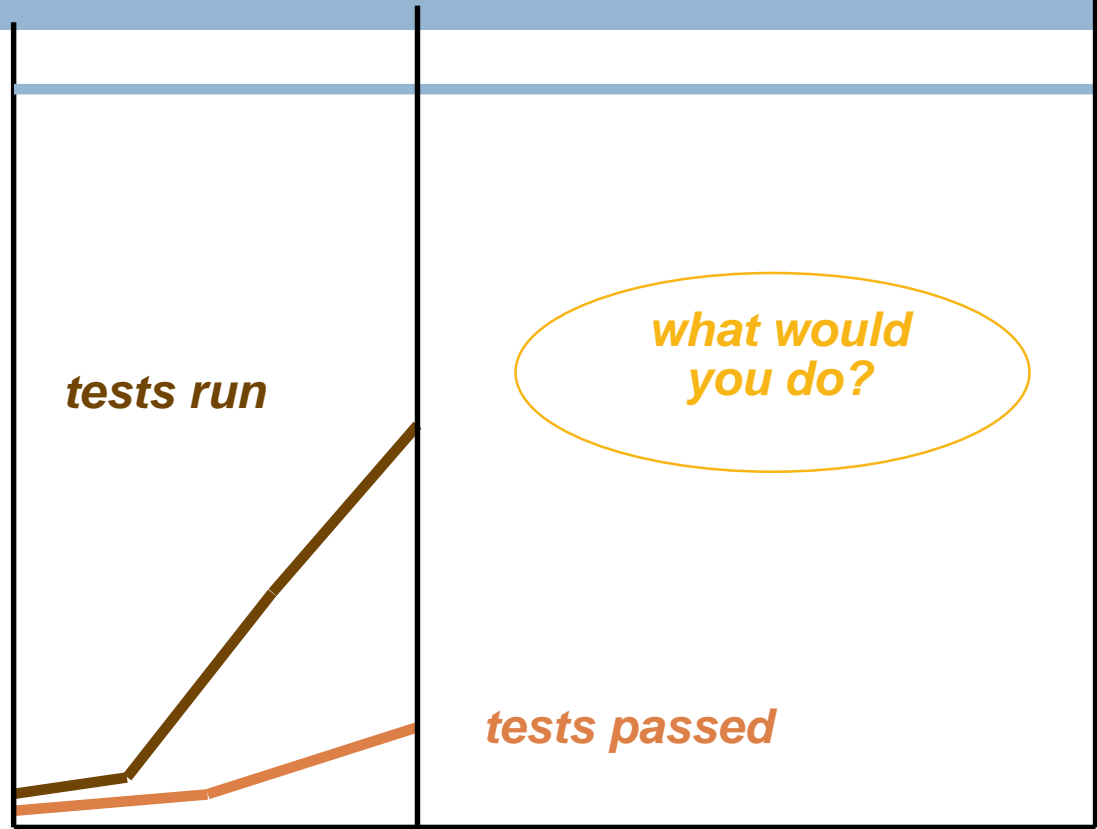
*what would  
you do?*

*tests passed*

*what does this  
mean?*

*now*

*release  
date*



# S- Curve

There are many graphs to monitor the test progress. One of them is the S curve .

- This is a powerful visual aid when plotted on a graph. This type of graph is called an 'S-curve' because it's shape resembles the letter S.
- S-curves give early warning of problems
- For example, if the number of tests passing falls significantly below the number run there may be a number of different reasons. Perhaps there are lots of faults being found or there are only one or two faults that are affecting many tests. In either case more development effort needs to be resourced to fix the faults. A faulty test environment could cause the problem, so in this case more development resource is likely to resolve it.

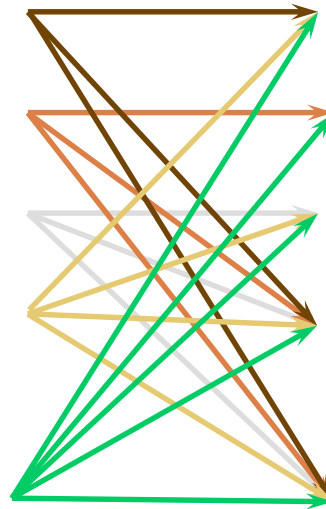
# Diverging S-curve

## Possible causes

poor test entry criteria  
ran easy tests first  
insufficient debug effort  
common faults affect all tests  
software quality very poor

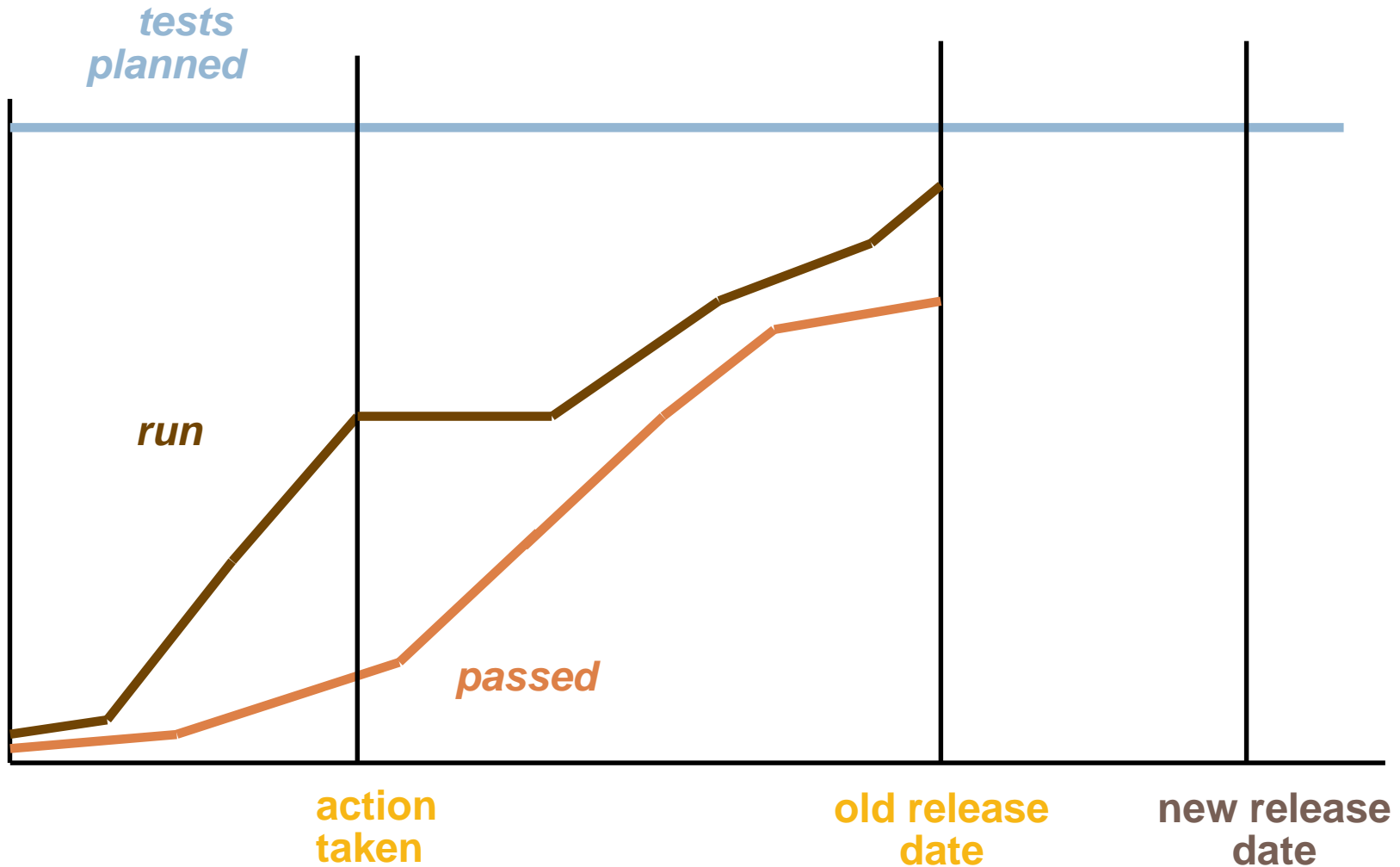
## Potential control actions

tighten entry criteria  
cancel project  
do more debugging  
stop testing until faults fixed  
continue testing to scope  
software quality

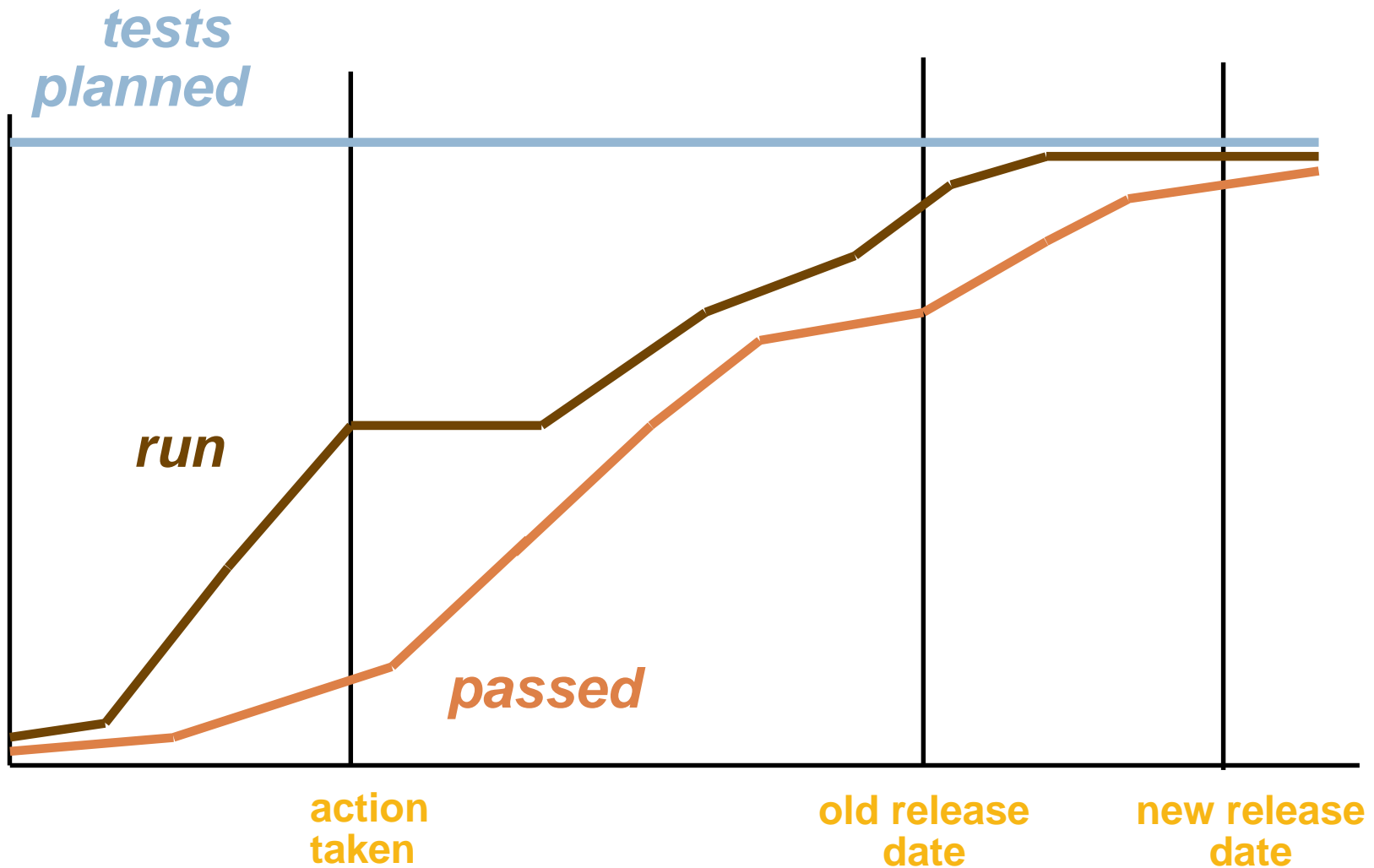


Note: solutions / actions will impact other things as well, e.g. schedules

# Measuring Test Execution Progress 2



# Measuring Test Execution Progress 3



# Defect Density, Failure rate

- Defect density is another method of monitoring the progress of Test. Defect density is :The ratio of the number of defects to program length
- Failure rate : Is the Common term for the curve that describes the expected failure rate of electronics with time: initially high, dropping to near 0 for most of the system's lifetime, then rising again as it "tires out"

# Test Control

## □ What is Test Control ?

- ▣ Test control is about the management actions and decision that affect the testing process, tasks and people with a view to making a testing effort achieve its objectives. This may be the original or a modified plan. Modifying an original plan in the light of new knowledge (i.e. what testing has revealed so far) is a frequently necessary and prudent step.

## □ Management actions and decisions

- ▣ affect the process, tasks and people
- ▣ to meet original or modified plan
- ▣ to achieve objectives

**Feedback is essential to see the effect of actions and decisions**



# Examples

- Tightening (or loosening) entry and exit criteria are just one the actions managers can take. There is a change made to either of the criteria and sent out for a formal approval with the updates.
- Reallocation of resources such as acquiring more testers or developers, and moving people from one task to another to focus attention on more important areas is often an effective controlling action

# Test Reporting



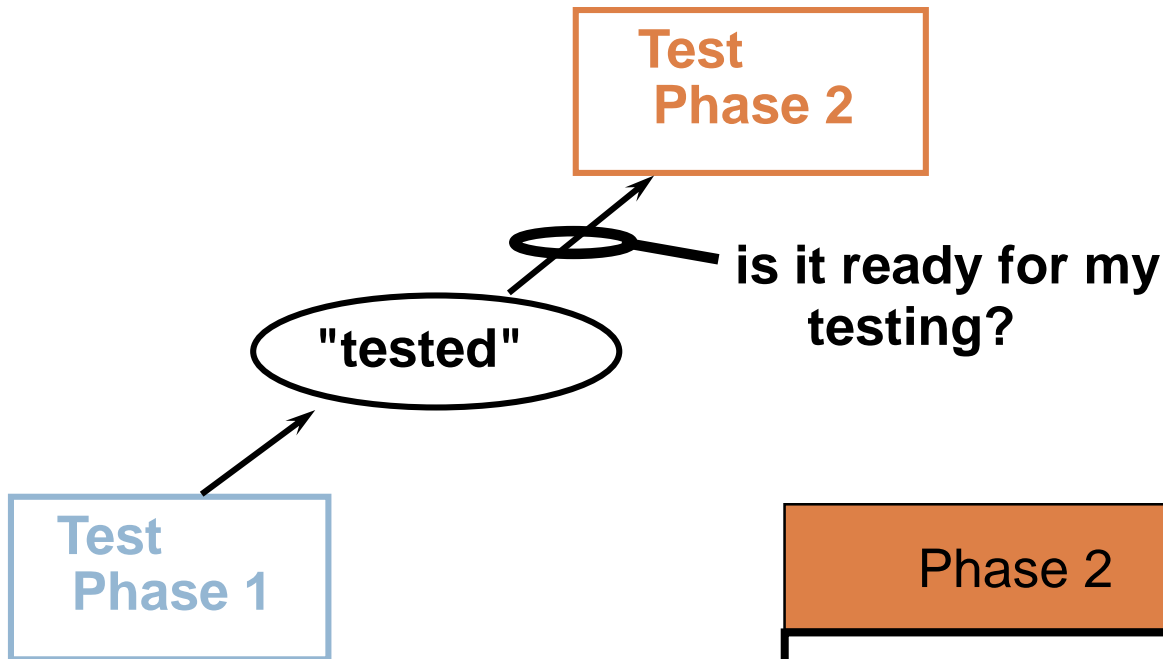
Test reporting is concerned with summarizing information about the testing Endeavour, including:

- What happened during a period of testing, such as dates when exit criteria were met.
- Analyzed information and metrics to support recommendations and decisions about future
- actions, such as an assessment of defects remaining, the economic benefit of continued
- testing, outstanding risks, and the level of confidence in tested software.

# Test Reporting Contd...

- Metrics should be collected during and at the end of a test level in order to assess:
- The adequacy of the test objectives for that test level.
- The adequacy of the test approaches taken.
- The effectiveness of the testing with respect to its objectives.

# Entry and Exit criteria



Phase 2	Phase 1
Entry criteria	Exit criteria
Acceptance criteria	Completion criteria

# What actions can you take?

- What can you affect?
  - ▣ resource allocation
  - ▣ number of test iterations
  - ▣ tests included in an iteration
  - ▣ entry / exit criteria applied
  - ▣ release date
- What can you not affect:
  - ▣ number of faults already there
- What can you affect indirectly?
  - ▣ rework effort
  - ▣ which faults to be fixed [first]
  - ▣ quality of fixes (entry criteria to retest)

# Summary

- Tests must be
  - ▣ Estimated
  - ▣ Monitored and
  - ▣ Controlled
- Different methods to monitor test execution is outlined with examples.
- Test reporting definition and test summary report discussed.
- Entry/Exit criteria for a test is explained