Instructor Notes:

Defect Reporting and Defect Life Cycle Management

Lesson1: Defect Free Defect Reporting

Instructor Notes:

Lesson Objectives

- To understand the following topics:
  - What is Software Quality?
  - Defect Definition
  - Why find Defects?
  - Impact of Defects
  - Life-cycle workflow
  - Defect Report – Definition, Need
  - Defect Report - Template
  - Example on Severity & Priority
  - Defective Reports - Certain Facts
  - Importance of Effective Defect Reporting
  - Defect Free Report - Recommendations
  - Defect Free Reports - Advantages

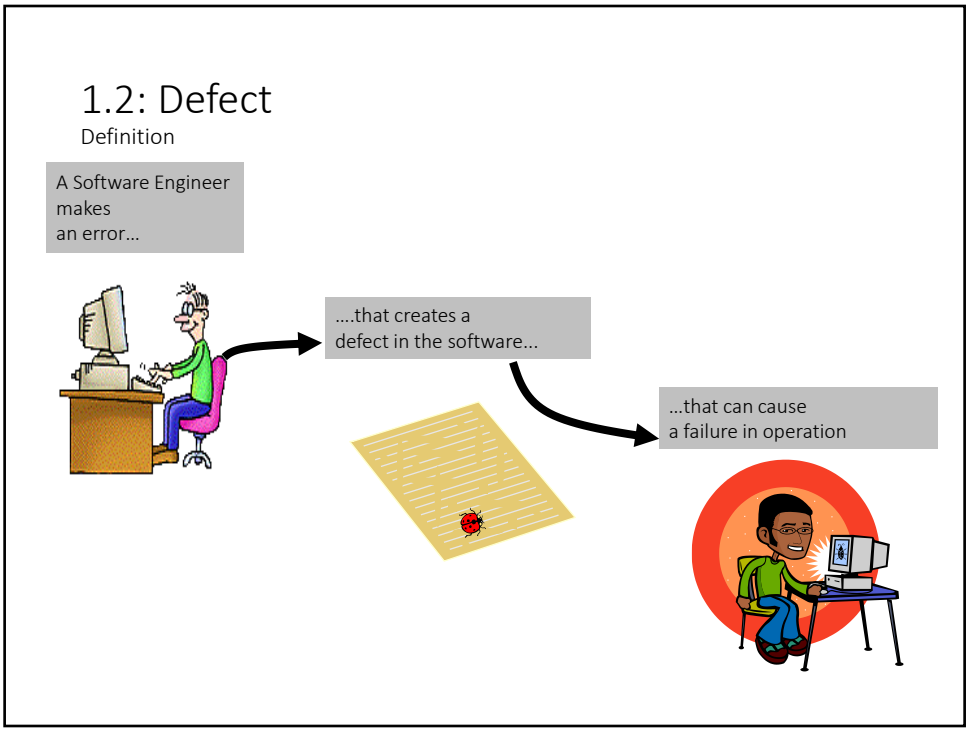
Instructor Notes:

1.1 Software Quality

What is Software Quality?

- Quality of the developed software exhibits the following aspects :
  - It is reasonably bug or defect free
  - Delivered on time and within budget
  - Meets requirements and is maintainable
- ISO 8402-1986 standard defines quality as “Totality of features or characteristics of a product or service that bear on its ability to satisfy Stated and Implied needs
- Acceptance criteria defines what minimum requirements should be met

Instructor Notes:



Instructor Notes:

1.2: Defect

Definition

- Defect
  - If AUT's (Application Under Test's) some feature or function is not working as per what is there in requirement it is called as defect
  - A defect is a variance from a desired product attribute.
  - A problem which, if not corrected, could cause an application to either fail or to produce incorrect results
- Examples
  - Car brakes stop working after crossing speed of 100 km/hr
  - Billing software generates, prints, and mails bills showing amount payable as 0 Rupees
  - A customer goes to withdraw $1000 from his account having a balance of $5000 and minimum bank balance require is $500. ATM Rejects the request saying "Insufficient Balance"
  - Actual amount withdrawn from the ATM and the amount printed on print receipt shows difference

Instructor Notes:

1.2: Defect

Why find Defects?

- Increase confidence in the reliable operation of the system and get more business
- Reduce the likelihood of loss or even life-threatening incidents
- Obtain repeat and referral business from satisfied customer
- Decrease overall system costs associated with quality problems

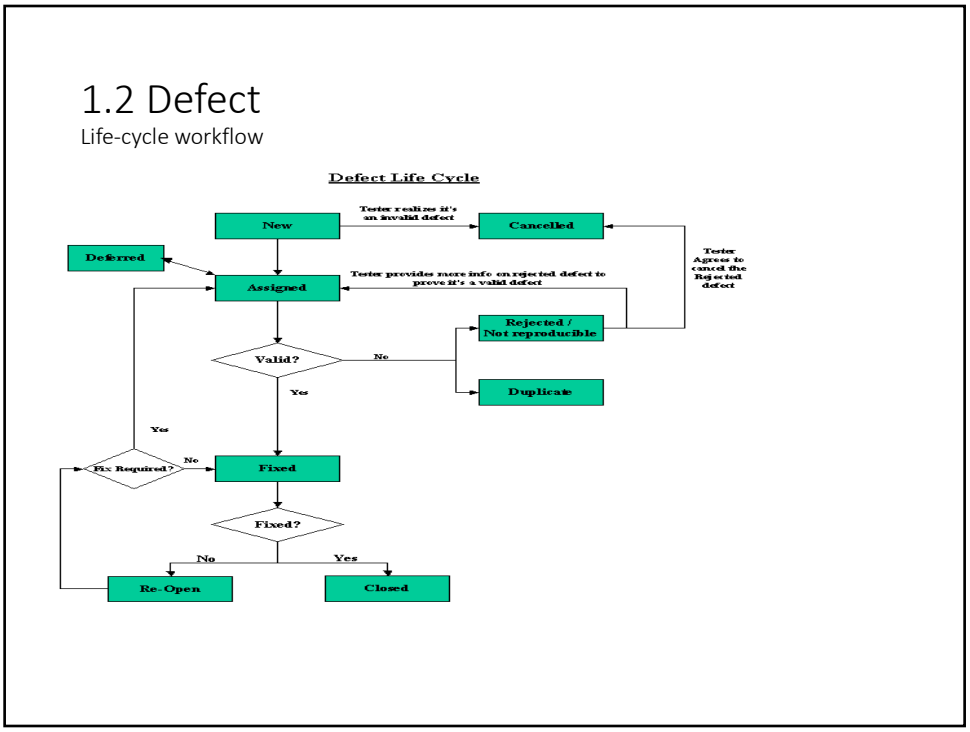
Instructor Notes:

1.2: Defect

Impact of Defects

- A single error can cause nothing or a lot
- It can cause death or injury if it fails in case of safety critical applications
- It can also cause huge financial loss to clients
- And also lead to fine/penalties for us
- Examples
  - NYSE fined Waterhouse Investor Services US $225,000 for its web site failures - inability to file on-line stock orders and inadequate customer service
  - An AA jet crashed in Colombia because the captain entered an incorrect one-letter computer command that sent the jet into a mountain killing 158 people aboard. When there is critical command, the software could have asked for confirmation or verified or have enough validation before processing the command.
  - Hacker hacked into US government computers, including two agencies within the Defense Department, and defaced government Web sites. It shows the insufficient Security Testing.

Instructor Notes:



1. **New** – When a Defect is logged and yet to be assigned to a developer. Usually Project Manager or Dev Lead will decide on which defects to be assigned to which developer.

2. **Assigned** – indicates that the developer who would fix the defect has been identified and has started analyzing and working on the defect fix.

3. **Duplicate** – Manager or Developer will update the status of a defect as "Duplicate" if this defect was already reported.

4. **Rejected / Not Reproducible** – This status indicates that the developer is not considering the defect as valid due to following reasons
   a) Not able to reproduce
   b) Not a valid defect and it is as per requirement
   c) Test Data used was invalid
   d) Defect referring to the Requirement has been de-scoped from the current release, tester was not aware of this late changes.

5. **Deferred** – Defect fix has been held back because of time or budget constraints and project team has got approval from customer to defer the defect till next or future release.

6. **Fixed** – Developer has fixed the defect and has unit tested the fix. The code changes are deployed in test environment for verifying the defect fix.

Instructor Notes:

7. **Reopen** – Status is changed to “Reopen” by a tester, when a tester finds the defect is Not fixed or partially fixed. Developer who fixed the defect looks into the comment that was provided by the tester at the time of reopening the defect. Developer will change the status to “Assigned” and starts working on the fix again. Incase the developer wants the tester to re-verify the defect then he/she will add a comment and will change the defect status to “Fixed”.

8. **Closed** – Tester verifies the defects that are in “Fixed” status and once they find the defect is fixed, they change the status to “Closed”. This is the last status of Defect Life Cycle.

9. **Cancelled** – This status indicates that the tester realized that the defect logged by him was invalid and agreed to cancel it.

Instructor Notes:

1.3: Defect Report

Definition

- Defect report
  - Is a document to maintain all the defects, that test engineer found while test execution
  - The most important deliverables to come out of test. It will have more impact on the quality of the product than most other deliverables from test
  - It is important to write effective defect reports

Instructor Notes:

1.3: Defect Reports

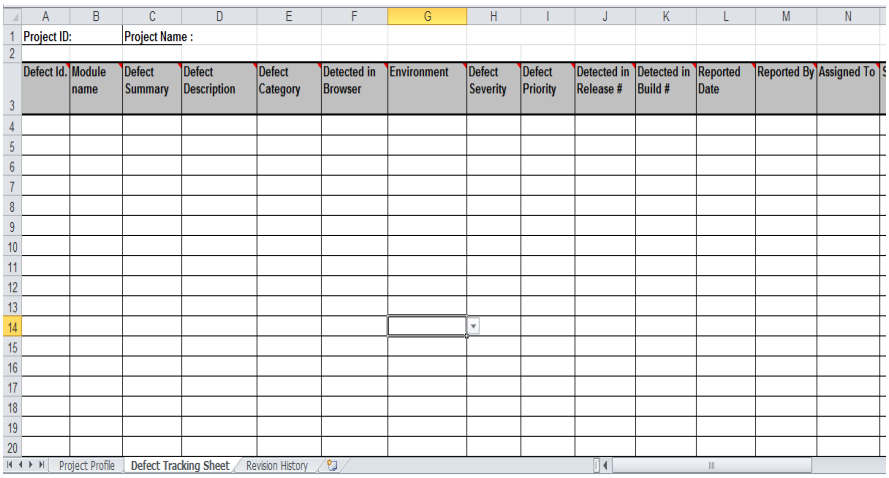
Defect Reporting – The Need

- Emphasize on continuous improvement
- Defect report – an important deliverable
- Inadequate Material
- High impact of defective defect report

Instructor Notes:

1.4 Defect Report

Template



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Project ID:		Project Name :											
2														
3	Defect Id.	Module name	Defect Summary	Defect Description	Defect Category	Detected in Browser	Environment	Defect Severity	Defect Priority	Detected in Release #	Detected in Build #	Reported Date	Reported By	Assigned To
4														
5														
6														
7														
8														
9														
10														
11														
12														
13														
14														
15														
16														
17														
18														
19														
20														

Project Profile | Defect Tracking Sheet | Revision History

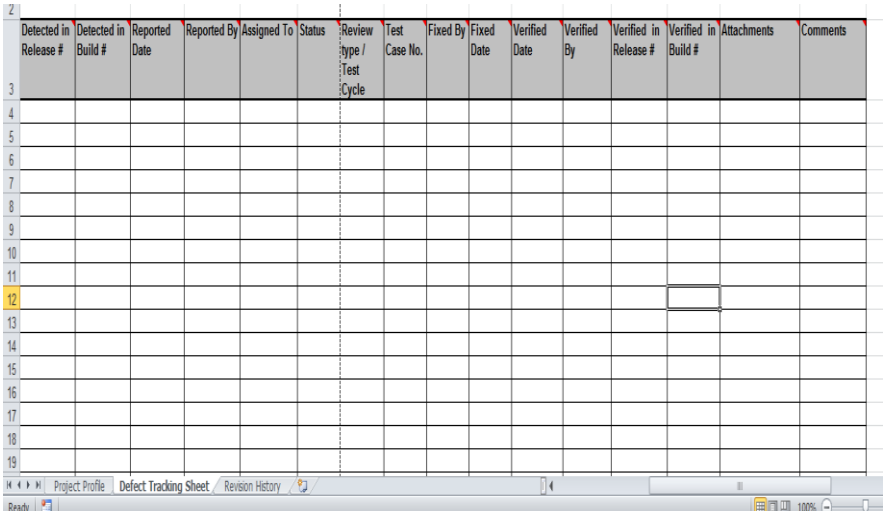
- **Defect ID** - A unique number to each defect. This will help to identify the bug record. If you are using any automated bug-reporting tool then this unique number will be generated automatically each time you report the bug.
- **Module Name-** It will contain the name of the module being tested.
- **Defect Summary-** A short summary of the defect. It can be in 1 or 2 lines.
- **Defect Description**- A detailed description of bug. It includes
    1. Clearly mention the steps to reproduce the bug.
    2. How application should behave on above mentioned steps.
    3. What is the actual result on running above steps i.e. the bug behavior.

- **Defect Category-** Categorizing the defect into the appropriate category. The categories considered are:
    - Coding: When the defect is found in the code.
    - Design :When the defect is found in the design
    - Enhancement: If the defect stated is actually an enhancement to the present requirement
    - New Requirement :If the defect stated is actually a new requirement
    - Query : is any question or doubt which might be raised by the tester, it need not be an actual defect.
    - Documentation: is any error found in the documents of the application like the help document etc.
    - Master Data: Is any error found in the master data received
    - Test Review: is while doing the testing any review comments which might be suggested by the tester ,again it may not be a defect itself.
    - DO – Documentation related defects
    - This includes all defects related to missing or misstated requirement in Functional specification
    - BD/PK/LD - Build / Package / Load
    - Change management, library, version control.
    - FN/LO (Coding)
    - Function/ Program Logic (Logic, pointers, loops, recursion, computation); Not functioning as per the design / requirement.
    - EN - Environment
    - Environment (Design, compile, test, or other support system problems)
    - OP
    - If the optimal usage is lacking or performance is not up to the mark. Design will adversely affect the product's performance
    - SG – Suggestion
    - If the defect reported is a suggestion.

Instructor Notes:

1.4 Defect Report

Template(contd.)

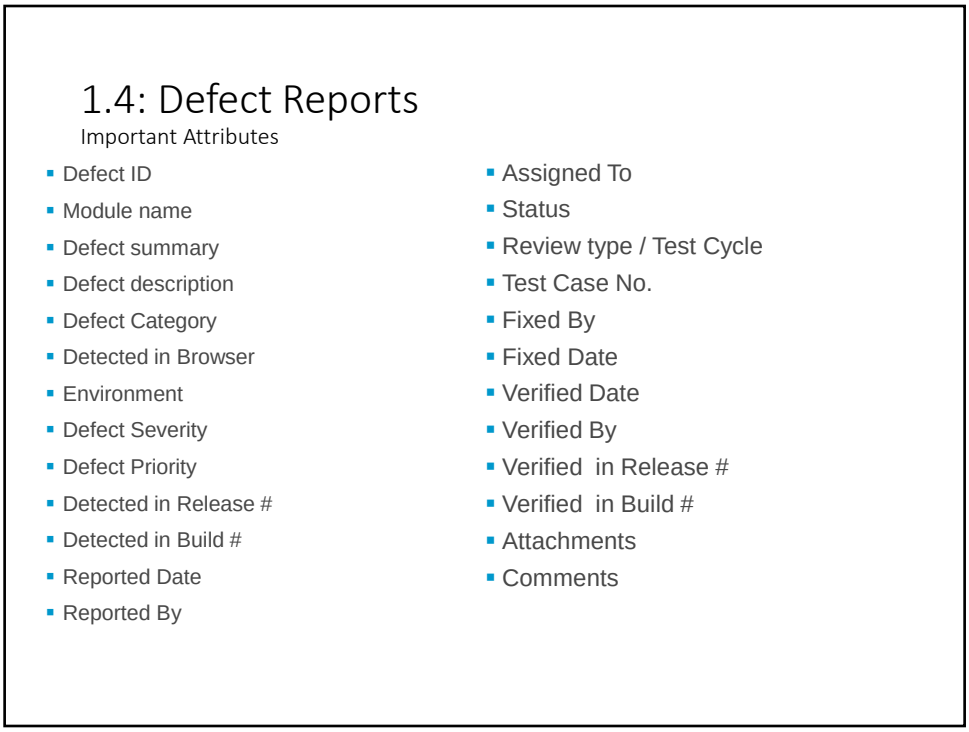


Detected in Release #	Detected in Build #	Reported Date	Reported By	Assigned To	Status	Review type / Test Cycle	Test Case No.	Fixed By	Fixed Date	Verified Date	Verified By	Verified in Release #	Verified in Build #	Attachments	Comments

Project Profile | Defect Tracking Sheet | Revision History

- **Detected in Browser :**The name of the browser the defect was found in (IE, Firefox ,etc.)
- **Environment:** Mention the environment where the defect was found. It can be internal Dev,internal QA,External Dev or External QA.
- **Defect Severity :** Impact of the defect on functionality of application. The values it can contain :
  - P0 defect is a defect wherein the main function of the software does not work. e.g. crash, hang, data corruption
  - P1 defect is a defect wherein a function does not work and a tedious work around exists. e.g. One of menu options does not work.
  - P2 defect is a defect wherein the software does useful work but a degree of inconvenience is caused. Correction is not deferrable and easy work around exists.
  - P3 defect is a tolerable defect as corrections are deferrable. E.g. cosmetic problems in user interface like spelling.
  - “High” severity defect is a defect wherein the main function of the software does not work. e.g. crash, hang, data corruption, some of the menu functions do not work
  - “Medium” severity defect is a defect wherein the software does useful work but a degree of inconvenience is caused. Correction is not deferrable and easy work around exists.
  - “Low” severity defect is a tolerable defect as corrections are deferrable. E.g. cosmetic problems in user interface like spelling.

Instructor Notes:



- **Priority** – It indicates When bug should be fixed? The values are low, medium and high
  - High - This bug should be resolved as soon as possible in the normal course of development activity, before the software is released.
  - Medium - This bug should be repaired after serious Defects have been fixed.
  - Low - It can be resolved in a future major system revision or not be resolved at all.
- **Detected in Release #** : Denotes the Release # in which the defect was detected.
- **Detected in Build #:** Denotes the Build # in which the defect was detected.
- **Reported Date:** The date on which the defect was reported. It need not be same as the date on which it was detected.
- **Reported By:** Name of the person who reported the defect.
- **Assigned To:** Name of the person the defect is assigned to , to fix it.
- **Status:** Gives the status of the defect. It can contain the following values
  - Verified
  - Closed
  - Fixed
  - Active
  - CNR-Could Not be Reproduced
  - NOD: reported bug Not a Defect
  - REP: Repeated bug
- **Review type / Test Cycle:** It can contain following values:
  - PPR-Peer to Peer Review
  - PR-Peer Review
  - SR-Self/Programmer Review
  - QPR-10% Quality Probe(Review)
  - RT-Random Testing
  - UTC-Using Test Cases

Instructor Notes:

- **Test Case No:** Mention the test case No that the defect was found while testing
- **Fixed By:** Name of the person who fixed the defect
- **Fixed Date:** Date on which the defect was fixed
- **Verified Date:** Date on which the defect was verified
- **Verified By:** Name of the person who will be verifying whether the defect has got fixed after the defect is being worked upon
- **Verified in Release #:** Denotes the Release # in which the defect was verified
- **Verified in Build # :**Denotes the Build # in which the defect was verified
- **Attachments:** Attach the proofs showing the defect- screenshots, temp files, etc.
- **Comments:** Should include all the comments made by all touching the defect on the defects till date .It should be captured date wise.

Instructor Notes:

1.5: Defect Reports

Example on Severity & Priority

- Think of the following type of problem:
- A spelling error on a user-interface screen
  - What severity and priority does this issue deserve?
- Well, judging from our earlier definitions, it would seem that this is a low-severity item. After all, the server doesn't crash due to a spelling error.
- But is this truly a low-severity problem?
- A spelling error will probably not hinder a customer's ability to use the system, but it greatly affects the customer's perception of the company that created the product and of the quality of the product. So from customer-relations and corporate-image points of view, the severity of this type of issue is indeed high. But the severity field doesn't allow us to express that properly. So the need for the priority field becomes apparent. The priority field does allow product management to define this issue as high priority, but this creates the case where something is low severity but high priority.

Instructor Notes:

1.5: Defect Reports

Example on Severity & Priority

- Let's consider another case:
- 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 but that is not reproducible by any human means on a consistent basis.
- So how would this issue be categorized within the defect tracking system?
- Well, since it is a server crash, many would argue it should be a high-severity issue. After all, the system is inoperable until the server is restarted. But what is the impact to the customer? In this case, the impact is quite small. Since the customer may never see this issue present itself at all in a production environment, it would be given a low priority and high severity by Product Management

Instructor Notes:

1.6 Defect Report

Users

- Management
- Maintenance Team lead
- Maintenance Engineer
- Testing Team Lead

Instructor Notes:

1.6 Defect Report

Benefits for Maintenance Team Leads

- Allocate the Defect to the appropriate team member as soon as possible
  - Quickly understand the software version and component responsible for issue
- Effectively prioritize defect for fixing
  - Is it halting the testing process?
  - Are other functionalities dependent on this?
  - Is it important under part release?
  - Expected fixing date
- To accurately take corrective and preventive actions for future developments
  - Category wise, severity wise Defect status (functionality, modules, layers)
  - Average turn around time for Defect fixing
  - Defect density

Instructor Notes:

1.6 Defect Report

Benefits for Maintenance Engineer

- Identify the application version (mainly for products)
  - Isolate the application version when multiple versions are being maintained
- Quickly get to the root cause
  - Concentrate not on symptoms but root cause to isolate the component creating issue
- Know reproducibility and environment/situation of reproducibility
  - Else do not waste time, arrange for necessary dependencies/settings
- Analyze the log file and get clear understanding about the issue
  - Check the exact details - data entered, actions taken, results generated, tables updated (application log, database log)
- Contact the tester who found the Defect
  - To get a first hand information & clarification directly and quickly

Instructor Notes:

1.6 Defect Report

Benefits for Testing Team Lead

- Plan retesting efforts
  - Tentative dates when defects are expected to be fixed
  - Estimated defects
- Analyze quality of Defect reporting process
  - Defect Acceptance Rate, Defect Communication Effectiveness
- Increase accuracy on future estimates
  - Generate accurate summaries for status reporting
  - Application module wise, Severity wise total/open defects
  - Functionality wise, severity wise total/open defect counts
- Monitor performance of the testers
  - Tester wise metrics

Instructor Notes:

1.6 Defect Report

Benefits for the Management

- To know the status of the Defects
  - Application module wise, severity wise defect summary of open and closed defects
  - Category wise severity wise open and closed defects
  - Expected dates for fixing and closing of high severity defects
- To analyze the performance of the teams
  - Team wise – team member wise metrics – count, productivity
- Effective follow-up
  - Generate exception reports – Actions due in next n days, Actions pending for more than n days
  - To take go/no go decision for next cycle/phase/production
  - Defect summary in conjunction with Test Case execution summary
- To know the risks involved
  - Summary and details of known defects (with impact)

Instructor Notes:

1.8 Defective Reports

What is Defective Defect Report?

- Defective defect Report
  - The inaccurate, incomplete and unclear defects results into defective defect report
  - Impact of Defective defect report
  - Wastage of time that is precious in tight schedule
  - Inaccurate / incomplete status leading to wrong / no decisions
  - Inaccurate statistics leading to inaccurate corrective / preventive measures
  - Frustration and ill feeling between development and testing teams

Instructor Notes:

1.8 Defective Reports

The reasons for defective reports

- Cannot reproduce
  - If the maintenance engineer is not able to reproduce the bug, by using steps mentioned in defect report
  - Already reported (Duplicate)
  - Functionality is as per requirement
  - Some one else is responsible
  - Additional information needed – Error message detail, Data input, options selected, previous tasks executed etc
  - Details provided are not clear
  - Some attributes are not provided or not correct – Severity, Transaction Id, version, category etc
  - Is a new requirement or change in requirement

For all above reasons the developer will change the status of the bug as rejected.

Instructor Notes:

1.8 Defective Reports

Root causes for defective reports

- An Assumption - Developer should be able to understand defect quickly and easily with little hint
- Providing all the steps, test data etc. takes lot of time to report
- Testers do not know the importance (usage) of details other than defect description
- Providing evidences for the defect are not considered important
- Informal communication process - through emails, verbal, and historical details are not maintained
- Some defects gets unknowingly fixed due to fixing of other defects
- Features of tool, process not known

Instructor Notes:

1.11 Defect Free Reports

Importance of Effective Defect Reporting

- From the Development Perspective
  - Real Defects in the system/program
  - Clear but brief information about the bug
  - Steps to easily reproduce the problem
  - Proper description of the problem if they happened to be more general
  - Developer should be able to isolate the problem reported
  - Increased productivity – in fixing the problem with least amount of effort
  - Expect reports that convey the proper message and simplifies the process

Instructor Notes:

1.11 Defect Free Reports

Importance of Effective Defect Reporting

- From the Test Perspective
  - Reduce the defect life cycle
  - Ensure that the defects get fixed by developers in the
  - Agreed timeframe
  - Improve the credibility of the test
  - Enhance teamwork between development and test
  - Get better response from the development team
  - Reduce things like “Need more feedback”, “Works fine on my machine”

Instructor Notes:

1.11 Defect Free Reports

Importance of Effective Defect Reporting

- From the Management Perspective
  - Improve productivity
  - Get accurate information on the defects reported
  - Reduce the time to market
  - Get the correct metrics
  - Take proper actions in timely resolution of the defects

Instructor Notes:

Summary

- In this lesson, you have learnt:
  - Defect report is an important deliverable since it gets referred by maintenance team, testing team, management
  - The inaccurate, incomplete and unclear defects results into wrong decisions
  - Follow process and guidelines
  - People fixing defects are most likely to be different than original developers
  - Institutionalizing process and building competencies for defect free defect reporting

Instructor Notes:

Answers:
Question1: Option4
Question2: False
Question3: True

Review Question

- Question1: Which of the following will be entered by test engineer in the defect report
  - Option 1: Reported By
  - Option 2: Resolution Details
  - Option 3: References
  - Option 4: All of the above
- Question 2: Before you log a defect it is not necessary to verify whether it is duplicate because it is time consuming.
  - True/ False
- Question 3: Adding attachments is easy if we are using any tool to log the defect
  - True/ False