Testing

Lec

* software bugs occurss when one of following occurs:
① software does something that project spec. said it shouldn't do.
② " doesn't do " " " " " " should do.
③ Software does something that product specification doesn't mention
④ software doesn't do something that product specification doesn't mention but should.
⑤ The software is difficult to under stand, hard to use.

* The main reason of Bugs occur is the specifications

* The goal of software tester is to find bugs & report them as early as possible.

* A Software quality assurance person's main responsibility is to create & enforce standards & methods to improve the development process & prevent bugs from ocurring.

Lec 2

* why would abug not being fixed?
1- There isn't enough time
2- It's not really abug, it's feature
3- It's too risky to fix.
4- It's Just not worth it.

* Types of Bug Report : ① Minimal
② Singular
③ Reproducible
④ obvious & general ⟹ Extra

① Minimal :-
- It explains Just the facts & necessery details to descripe abug.
- Give an exact sequence of steps that shows the problem.

[2] Singular:-
- There should be only 1 bug per report

[3] Reproducible:
- To be taken seriously, the bug report must show the bug to be reproducible - Following a set of steps will cause software to achive the same state & the bug occurs again.

- Not all bugs are equal.
- Bugs are classified by : (1) serverity
  (2) priority

[1] serverity: Indicates how bad the bug is
It might be classified to following: 1- System crash, data loss or corrup.
2- operational error, wrong result
rare occurance, Misspelling ⟵ 3- Minor problem, UI Layout
4- suggestion

[2] priority: Indicates how important it is to fix a bug and when it should be fixed.
It might be classified to: 1- Immediate fix
2- Must be fixed before released
3- should fix if time permits
4- Would like be fixed

* Bug Life Cycle

$$\begin{tikzcd}[row sep=large]
\text{Bug found} \arrow[d] \\
\boxed{\text{open}} \arrow[d] \\
\boxed{\text{Resolved}} \arrow[d] \\
\boxed{\text{closed}}
\end{tikzcd}$$

Tester find bug & report it
Bug report assigned to programmer

programmer fixes bug
Bug report assigned to tester

Tester confirms bug is fixed
Tester close bug report

Lec 3

* Error: is amistake, misunderstanding on the part of software developer.

. Types of testing
. Testing lvl
. V. model
. Test plan

* Faults (Defects): is introduced to software as aresult of error,

* Failures: it's the inability of asoftware system or Component to perform it's required functions within specification required.

. Types of testing:-

refers
testing

[1] Static	[2] Dynamic
.analysis of the static system representation to discover problems.	. Exercising and observing the Software behaviour

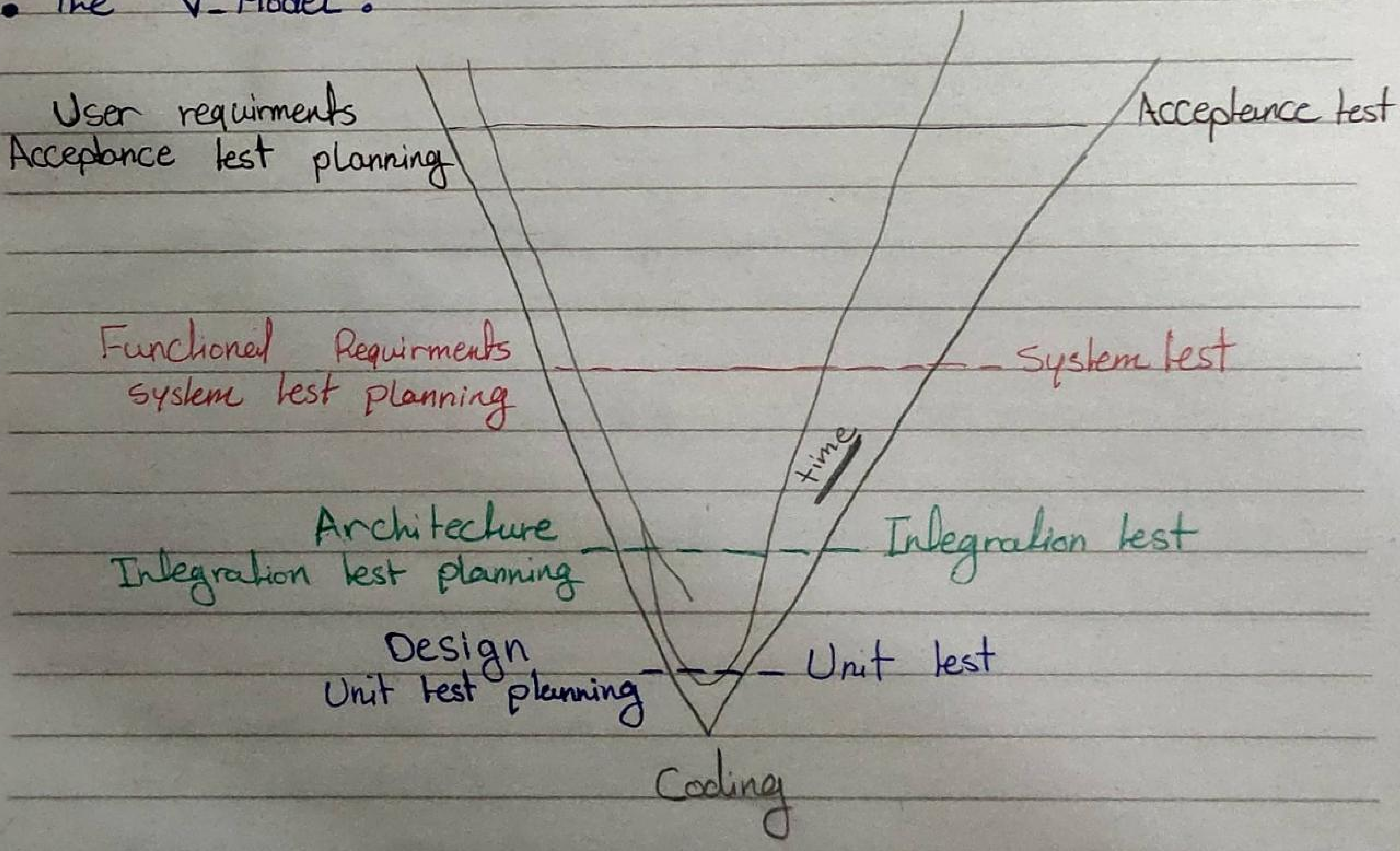
. static testing (before Compile time):
- Static analysis
- Review
- walk through
- Code inspection

. Dynamic testing (Run time):
- Black-box testing
- white-box testing
- Testing Scope

. Testing Levels based on Software activity:
- Acceptance testing $\xrightarrow{on}$ Requerments Analysis
- System testing $\xrightarrow{on}$ Architectural Design
- Integration testing $\xrightarrow{on}$ Sub-System Design
- Module testing $\xrightarrow{on}$ Detailed Design
- Unit testing $\xrightarrow{on}$ Implementation

- Acceptence testing: is the software acceptable to users?
- System testing: test the overall functionality of the system.
- Integration testing: test how modules interact with each other.
- Module (Developer) testing: test each class, file module.
- Unit (Developer) testing: test each method individually.

- The V-Model:



- Test plans:
- a plan is a document that provides a framework or approach for acheiving a set of goals.

1- overall test objectives
2- what to test (Scope)
3- who will test
4- How to test
5- When to test
6- When to stop

1 over all test obj → Introduction
→ Risks & Contingencies

2 Scope of test → Items to be tested (eg. classes, Libraries)
→ Features to be tested (eg. Functional requirments)
→ Features not to be tested

3 Who will test → Responsibilities
→ staffing and training needs

4 How to test → Approach
→ Test deliverables
→ Know : tools, strategies, techniques

5 When to test → schedule

6 When to stop → Approach

extra things in plan:
- Item pass / Fail Criteria
- Test plan identifier
- suspension / resumption Criteria
- testing enviroment → needed software & hardware
- testing Cost
- testing tasks

Lec 4

- Software Fault (bug) → a static defect in software (incorrect Line of Code)
- Software error → an incorrect internal state
- Software Failure → incorrect behavior with respect to the requirments

PAGE

DATE

- program State is defined during execution of a program as the current value of all living varibles & the current Location as given by the program Counter

- program Counter is the next statement in the program to be execuled

---

- Testing : evaluating Software by observing it's execution Lec 5
- Test Failure : Execution of a test that results in a software Failure.
- Debugging : The process of finding a fault given a failure.

* 4 Conditions necessary for Failure to be observed :

1- Reachability : the Location that Contains the fault must be reached.
2- Infection : the state of the program must be incorrect.
3- propagation : the infection state must Cause some output or the final state of the program to be incorrect.
4- Reveal : the tester must observe part of the incorrect portion of the program state.

*Can be Called RIP-R Model

- How to deal with bugs & errors & Failures ?
  - Avoidance → Better Design
  - Detection → Testing , debugging
  - Tolerance → Redundancy

- Test Case : is a test-related item which Contains : 1- set of test input
  2- Execution Condition
  3- Expected output
- Test suite : is a group of related test Case.
- Test oracle : a program , document that produces or specifies the expected outcome of a test

Scanned by TapScanner

- Verification vs Validation :
  - verifying that the product has been developed right.
  - verifying that the right product has been developed.

- Verification is the process Confirming that the software meets its specification.
- Validation is the process Confirming that the software meets the user's requirments.

- A programmer should avoid attempting to test his own program.

- The probability of existence of more errors is proportional to the number of errors already found.

- Test must be repeatable and reusable.

- Don't plan atest effort under the assumption that no errors will be found.

- old view of testing was depending on phases Lec 6
  - Unit , module , integration , system

- New view is interm of structure & criteria
  - Input space , graph , Logical expressions , syntax

$\rightarrow$ Test design is largely the same at each phase
  - Creating model is different.

• The tester Job is simple : Choose the model then find away to cover it.

* Test Requirments: a specific element of a software artifact that a test case must satisfy or cover.
* Coverage criterion : a collection of rules that impose test requirments on a test set.

* Structure (4 ways of models)
1. Input Domain characterization (Input space)
2. Graph
3. Logical expersssion
4. syntactic structures (grammars)

* Coverage LvL : the ratio of the no. of test requirments satisfied by t to the size of T.

* Criteria subsumption ⟹ a test Criterion C1 subsumes C2 if and only if every set of test cases that satisfies criterion C1 also satisfies C2

* Monkey testing is bad
→ human sits at Keyboard & bangs it
→ No automation
→ Minimal training required

* Some Companies are using both automation & Criteria-based testing which Leads to
→ Save money
→ find more faults
→ Build better Software