TESTING - 1

WHAT WHY WHEN

Bugs!

- both are annoying!
- if you find one, chances are that others are nearby!
- like to lurk around corners
- become immune to pesticide!

BUG SPRAY

- Specifications
- Planning
- Repeatibility
- Accountability
- Economy

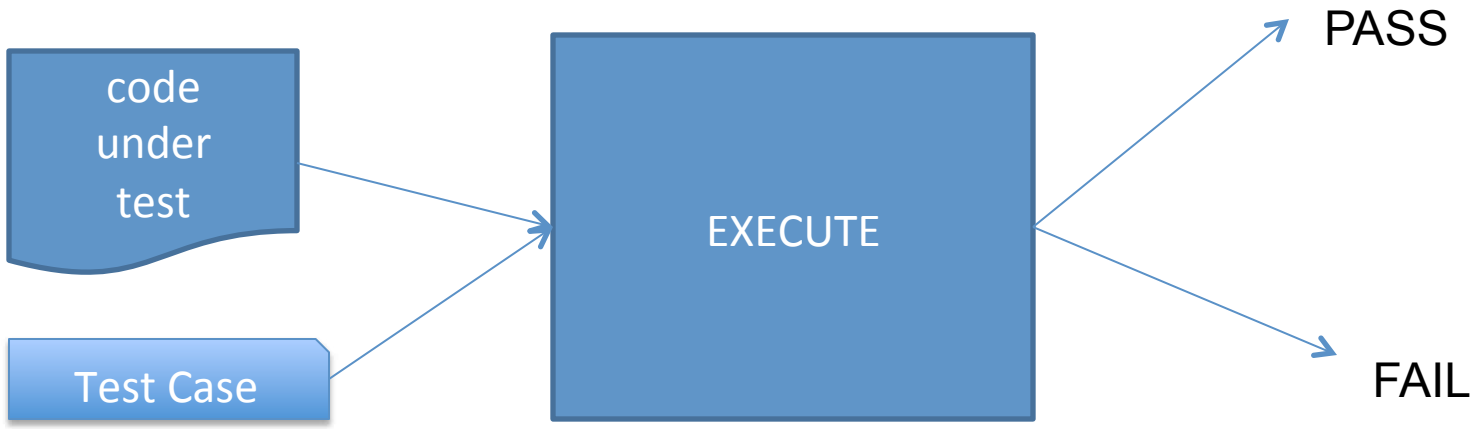
Topics

- What is testing?
- Why Test?
- When to Test?

WHAT IS TESTING?

Dynamic Testing

- This means execute the code and check if actual behavior matches expected behavior.



- Typically – the word "Testing" is used to mean dynamic testing.
- Black Box, White Box, Grey Box

Static Testing

- Code is not executed but is examined using tools like:
  - Compiler!
  - Lint
  - profiler
  - metrics (like Stan4J)
  - memory/resources analysis tools
- Also reviews/inspections
- Many errors can be caught this way saving valuable time and money!

Inspections/Reviews

- A formal process where
  - group inspects a software artifact (like code)
  - and detects errors.
- can be very effective in catching errors.
- has many additional benefits.
- important that it is done correctly.

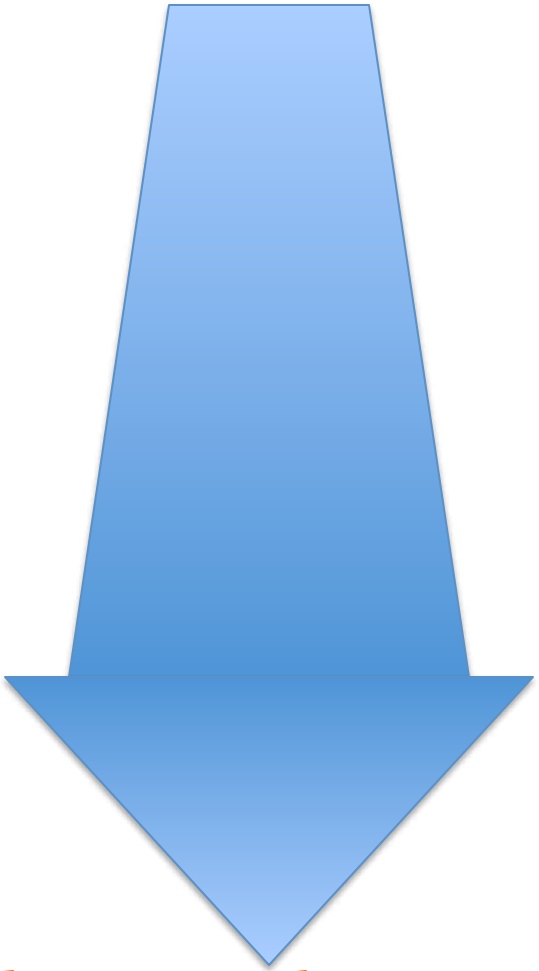
WHY TEST?

Real life examples

- Loss of life and limb!
  - Therac 25
  - Missile defense error – 28 soldiers killed
  - Airplane crash at Guam 1997– 225 people killed
- Financial losses
  - Ariane rocket 1996 $500 million
  - NASA Mars Polar lander 1999
  - Intel Pentium fp division error - $400million

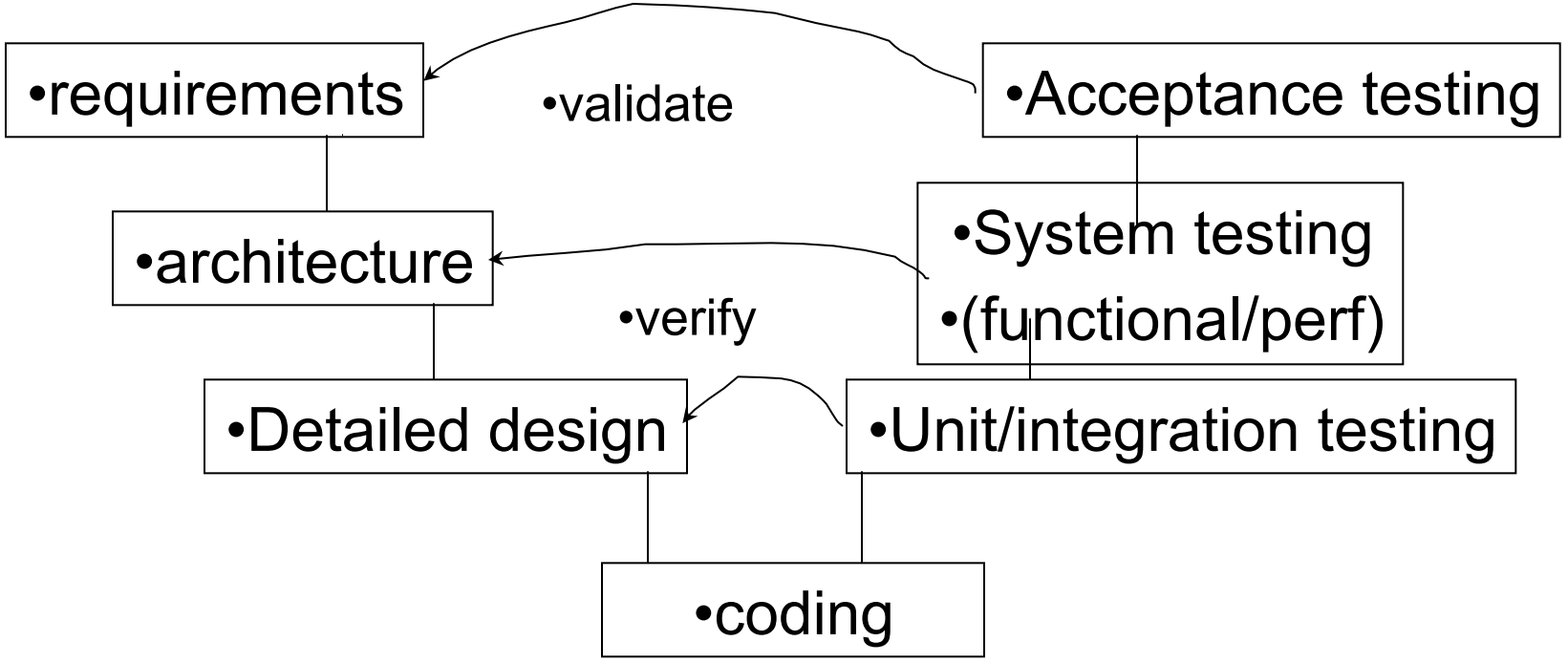
Range of problems

- Annoyance/Discomfort!
- Loss of work/productivity
- Information Loss
- Privacy loss
- Security Loss
- Loss of money
- Loss of life
- …
- Testing costs can vary from almost nothing to 2/3rds of project costs



Other reasons

- Ethical reasons
- Legal /due diligence
- Certification (FAA certification Rockwell collins)
- Loss of market share, competetive advantages.
- Effort/$ to manage/fix once released to customers.



V MODEL

WHEN TO TEST

Implications of V-Model

1. Errors in upstream processes are more expensive to debug and fix.

2. Not only that - Industry data also shows frequency of errors occuring in upstream processes is **higher**!

- incremental/iterative/agile forms of development
  - frequent releases to customer (find defects early)
  - involve a customer from the beginning in working with the development team

What can you do - reqs?

1. Show prototypes/screensketches
2. Evaluate each requirement for testability
3. Design Fit-criterion and corresponding acceptance tests
4. Make sure requirements are validated

- Testers should be also involved in requirements and design phases of development.
- Inspections/reviews to TRAP errors from flowing "downstream" is essential.

What can you do -arch/coding?

1. Logging (levels of verbosity)
2. Checkpointing (recoverable states)
3. Testable: Controllable/Observable
4. Plan out top-down and otherwise integration testing mechanisms
5. pre-conditions, post-conditions, assertions

- Testers should be also involved in requirements and design phases of development.
- Inspections/reviews to TRAP errors from flowing "downstream" is essential.