Software Testing and Quality Assurance

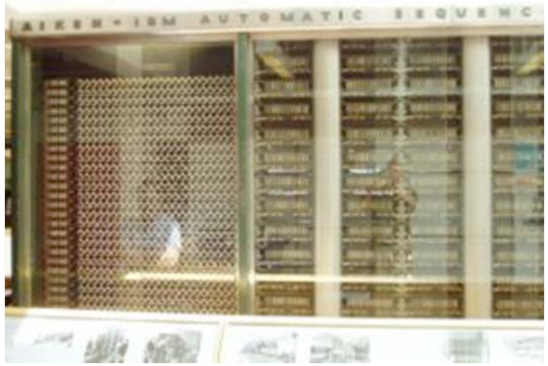
Lecture 1:
Introduction to Software Testing- I

Dr. Jameleddine HASSINE
ICS Department, KFUPM
jhassine@kfupm.edu.sa

Outline

- What is a computer bug ?
- Examples of real world bugs
- Effects of faulty software
- Testing goals
- Definition of testing
- Cost of testing
- Testing levels based on Test Process Maturity
- Types of test Activities
- Types of software testing

What is a computer bug?



In 1988, the log, with the moth still taped by the entry, was in the Naval Surface Warfare Center Computer Museum at Dahlgren, Virginia.

- In 1945 Harvard University was operating a room-sized computer called the Mark II.
  - mechanical relays
  - vacuum tubes
  - technicians program the computer by reconfiguring it
  - Technicians had to change the occasional vacuum tube
- A moth flew into the computer and was zapped by the high voltage when it landed on a relay.
- Hence, the first computer bug !

http://www.history.navy.mil/photos/images/h96000/h96566kc.htm

Bugs a.k.a. …

- Defect
- Fault
- Problem
- Error
- Incident
- Anomaly
- Failure
- Inconsistency
- Feature :-)

Bug in Space Code

- **Project Mercury** was the first human spaceflight program of the United States. It ran from 1959 through 1963 with the goal of putting a human in orbit around the Earth.
- Project Mercury's FORTRAN code had the following fault:

  DO I=1.10 instead of ... DO I=1,10

- The fault was discovered in an analysis of why the software did not seem to generate results that were sufficiently accurate.
- The erroneous 1.10 would cause the loop to be executed exactly once!

Military Aviation Problems

- An F-18 crashed because of a missing exception condition: if ... then ... without the else clause that was thought could not possibly arise.

- In simulation, an F-16 program bug caused the virtual plane to flip over whenever it crossed the equator, as a result of a missing minus sign to indicate south latitude.

Year Ambiguities

- In 1992, Mary Bandar received an invitation to attend a kindergarten in Winona, Minnesota, along with others born in '88.

- Mary was 104 years old at the time.

Shaky Math

- In the US, five nuclear power plants were shut down in 1979 because of a program fault in a simulation program used to design nuclear reactor to withstand earthquakes.

- This program fault was, unfortunately, discovered after the power plants were built!

- Apparently, the arithmetic sum of a set of numbers was taken, instead of the sum of the absolute values.

AT&T Bug: Hello? ... Hello?

- In mid-December 1989, AT&T installed new software in 114 electronic switching systems.
- On January 15, 1990, 5 million calls were blocked during a 9 hour period nationwide.
- The bug was traced to a C program that contained a break statement within a switch clause nested within a loop.

Bank Generosity

- A Norwegian bank ATM consistently dispersed 10 times the amount required.
- Many people joyously joined the queues as the word spread.

Discussion …

- Have you heard of other software bugs?
  - In the media?
  - From personal experience?
- In your opinion, what are the principal causes for such real-life failures ?

Adverse Effects of Faulty Software

- **Communications:** Loss or corruption of communication media, non delivery of data.
- **Space Applications:** Lost lives, launch delays.
- **Defense and Warfare:** Misidentification of friend or foe.
- **Transportation:** Deaths, delays, sudden acceleration, inability to brake.
- **Safety-critical Applications:** Death, injuries.
- **Electric Power:** Death, injuries, power outages, long-term health hazards (radiation).

Adverse Effects of Faulty Software (Cont'd)

- **Money Management:** Fraud, violation of privacy, shutdown of stock exchanges and banks, negative interest rates.
- **Control of Elections:** Wrong results (intentional or non-intentional).
- **Control of Jails:** Technology-aided escape attempts and successes, accidental release of inmates, failures in software controlled locks.
- **Law Enforcement:** False arrests and imprisonments.

Costly Software Failures

- NIST report, "The Economic Impacts of Inadequate Infrastructure for Software Testing" (2002)
  - Inadequate software testing costs the US alone between $22 and $59 billion annually
  - Better approaches could cut this amount in half

- Huge losses due to web application failures
  - Financial services : $6.5 million per hour (just in USA!)
  - Credit card sales applications : $2.4 million per hour (in USA)

NIS: National Institute of Standards & Technology (US)

What Does This Mean?

Software testing is getting more important

Activity

What are the goals of Software Testing ?

What are the goals of Software Testing ?

- Ensure that all the functionalities (requested by the customer) are implemented
- Demonstrate that faults are not present
- Error detection and removal
- Determine the level of reliability of the software
- Gain confidence in the software

Definitions of Testing

- Hetzel: Any activity aimed at evaluating an attribute or capability of a program or system. It is the measurement of software quality.
- Beizer: The act of executing tests. Tests are designed and then executed to demonstrate the correspondence between an element and its specification.
- Myers: The process of executing a program with the *intent* of finding errors.
- IEEE: The process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements or to identify differences between expected and actual results.

What is Software Testing ?

- Testing is a technical process, performed by experimenting with a software product in a controlled environment, following a specified procedure, with the intent of observing one or more characteristics of the product by demonstrating the deviation of the product's actual status from the required status/specification.
- Test cases must be designed to have a maximum fault coverage, minimum time and effort.

Cost of Testing

You're going to spend at least half of your development budget on testing, whether you want to or not

- In the real-world, testing is the principle post-design activity
- Restricting early testing usually increases cost
- Extensive hardware-software integration requires more testing

Why Test?

If you don't know why you're conducting each test, it won't be very helpful

- Written test objectives and requirements must be documented
- What fact is each test trying to verify?
- What are your planned coverage levels?
- How much testing is enough?
- Common objective – spend the budget … test until the ship-date …
  - Sometimes called the "date criterion"

Cost of Not Testing

Program Managers often say:
"Testing is too expensive."

- Not testing is even more expensive
- Planning for testing after development is prohibitively expensive
- A test station for circuit boards costs half a million dollars
- Software test tools cost less than $10,000 !!!

Software Testing Characteristics

- Important to control the quality of the product (and of the process)
- Often very expensive (Not to test is even more expensive)
- Difficult and stimulating
- Time critical

However it is:

- Only rarely practiced
- Unsystematic
- Performed by Hand
- Error-Prone
- *Uncool :"If you are a bad programmer you might be a tester"*
- Destructive

Testing Levels Based on Test Process Maturity (Beizer's Testing Levels)

Level 0 Thinking

- There's no difference between testing and debugging
- Does not distinguish between incorrect behavior and mistakes in the program
- Does not help develop software that is reliable or safe

Testing Levels Based on Test Process Maturity (Beizer's Testing Levels)

Level 1 Thinking

- The purpose of testing is to show correctness
- Correctness is impossible to achieve
- What do we say if there are no failures?
  - Good software or bad tests?
- Test engineers have no:
  - Strict goal
  - Real stopping rule
  - Formal test technique
  - Test managers are powerless

Testing Levels Based on Test Process Maturity (Beizer's Testing Levels)

Level 2 Thinking

- The purpose of testing is to show that the software doesn't work
- Purpose is to show failures
- Looking for failures is a negative activity
- Puts testers and developers into an adversarial relationship
- What if there are no failures?

Testing Levels Based on Test Process Maturity (Beizer's Testing Levels)

Level 3 Thinking

- The purpose of testing is not to prove anything specific, but to reduce the risk of using the software
- Whenever we use software, we incur some risk
- Risk may be small and consequences unimportant
- Risk may be great and the consequences catastrophic
- Testers and developers work together to reduce risk

Testing Levels Based on Test Process Maturity (Beizer's Testing Levels)

Level 4 Thinking

- Testing is a mental discipline that helps all IT professionals develop higher quality software
  - Testing is one way to increase quality
  - Test engineers can become technical leaders of the project
  - Primary responsibility to measure and improve software quality
  - Their expertise should help the developers

Types of Test Activities

- Testing can be broken up into four general types of activities
  1. Test Design
  2. Test Automation
  3. Test Execution
  4. Test Evaluation
- Each type of activity requires different skills, background knowledge, education and training
- No reasonable software development organization uses the same people for requirements, design, implementation, integration and configuration control

Test Engineer & Test Managers

- Test Engineer : An IT professional who is in charge of one or more technical test activities
  - designing test inputs
  - running test scripts
  - analyzing results
  - reporting results to developers and managers

- Test Manager : In charge of one or more test engineers
  - sets test policies and processes
  - interacts with other managers on the project
  - otherwise helps the engineers do their work

1. Test Design – (a) Criteria-Based

- Design test values to satisfy coverage criteria
- This is the most technical job in software testing
- Requires knowledge of :
  - Discrete math
  - Programming
  - Testing
- Requires much of a traditional CS degree
- This is intellectually stimulating, rewarding, and challenging
- Test design is analogous to software architecture on the development side
- Using people who are not qualified to design tests is a sure way to get ineffective tests

I. Test Design – (b) Human-Based

- Design test values based on domain knowledge of the program and human knowledge of testing
- This is much harder than it may seem to developers
- Criteria-based approaches can be blind to special situations
- Requires knowledge of :
  - Domain, testing, and user interfaces
- Requires almost no traditional CS
  - A background in the domain of the software is essential
  - An empirical background is very helpful (biology, psychology, …)
  - A logic background is very helpful (law, philosophy, math, …)
- This is intellectually stimulating, rewarding, and challenging
  - But not to typical CS majors – they want to solve problems and build things

2. Test Automation

- Embed test values into executable scripts
- This is slightly less technical
- Requires knowledge of programming
  - Fairly straightforward programming – small pieces and simple algorithms
- Requires very little theory
- Very boring for test designers
- Programming is out of reach for many domain experts
- Who is responsible for determining and embedding the expected outputs ?
  - Test designers may not always know the expected outputs
  - Test evaluators need to get involved early to help with this

3. Test Execution

- Run tests on the software and record the results
- This is easy – and trivial if the tests are well automated
- Requires basic computer skills
  - Interns
  - Employees with no technical background
- Asking qualified test designers to execute tests is a sure way to convince them to look for a development job
- Test executors have to be very careful and meticulous with bookkeeping

4. Test Evaluation

- Evaluate results of testing, report to developers
- This is much harder than it may seem
- Requires knowledge of :
  - Domain
  - Testing
  - User interfaces and psychology
- Usually requires almost no traditional CS
  - A background in the domain of the software is essential
  - An empirical background is very helpful (biology, psychology, …)
  - A logic background is very helpful (law, philosophy, math, …)
- This is intellectually stimulating, rewarding, and challenging
  - But not to typical CS majors – they want to solve problems and build things

Other Activities

- Test management : Sets policy, organizes team, interfaces with development, chooses criteria, decides how much automation is needed, …
- Test maintenance : Save tests for reuse as software evolves
  - Requires cooperation of test designers and automators
  - Deciding when to trim the test suite is partly policy and partly technical – and in general, very hard !
  - Tests should be put in configuration control tools
- Test documentation : All parties participate
  - Each test must document "why" – criterion and test requirement satisfied or a rationale for human-designed tests
  - Ensure traceability throughout the process
  - Keep documentation in the automated tests

Types of Test Activities – Summary

1a.	Design	Design test values to satisfy engineering goals
	Criteria	Requires knowledge of discrete math, programming and testing
1b.	Design	Design test values from domain knowledge and intuition
	Human	Requires knowledge of domain, UI, testing
2.	Automation	Embed test values into executable scripts
		Requires knowledge of scripting
3.	Execution	Run tests on the software and record the results
		Requires very little knowledge
4.	Evaluation	Evaluate results of testing, report to developers
		Requires domain knowledge

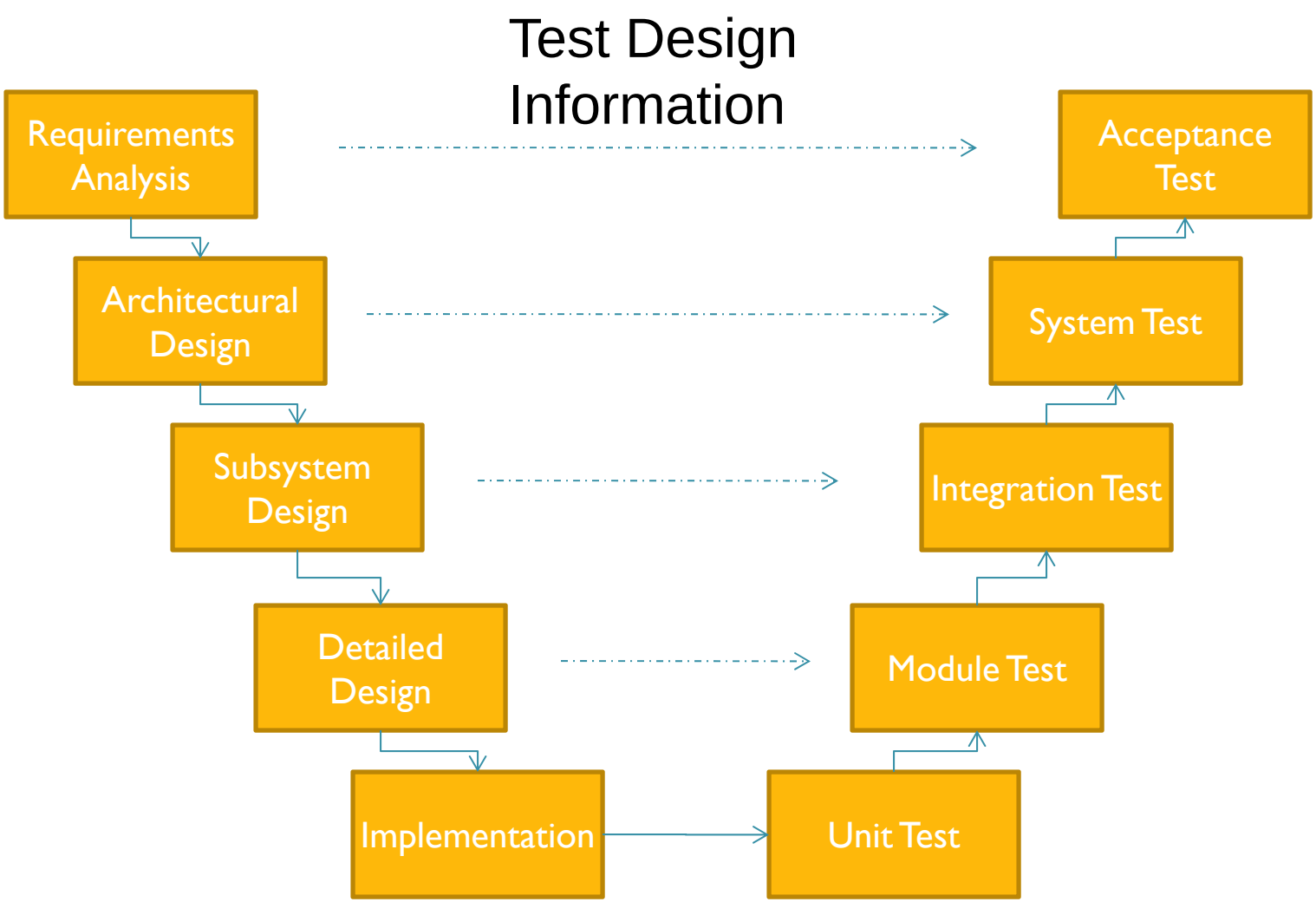
- These four general test activities are quite different
- It is a poor use of resources to use people inappropriately

Organizing the Team

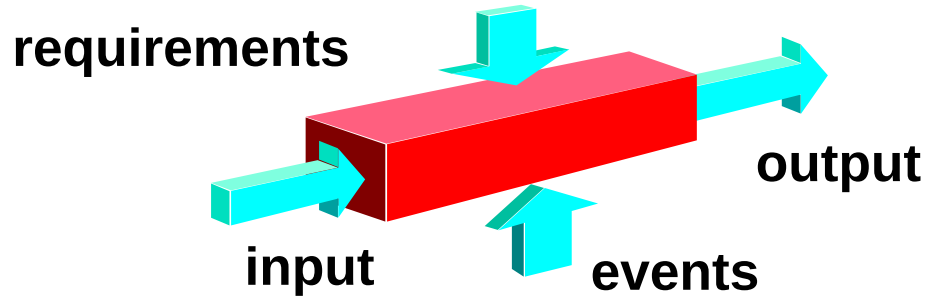
- A mature test organization needs only one test designer to work with several test automators, executors and evaluators
- Improved automation will reduce the number of test executors
  - Theoretically to zero … but not in practice
- Putting the wrong people on the wrong tasks leads to inefficiency, low job satisfaction and low job performance
  - A qualified test designer will be bored with other tasks and look for a job in development
  - A qualified test evaluator will not understand the benefits of test criteria
- Test evaluators have the domain knowledge, so they must be free to add tests that "blind" engineering processes will not think of
- The four test activities are quite different

Many test teams use the same people for ALL FOUR activities !!

V-Model of Development/Testing Activities



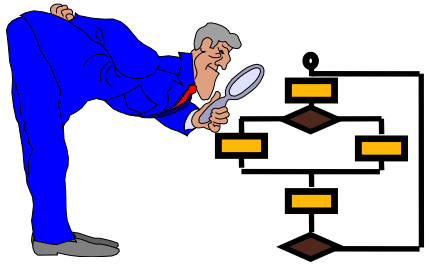
Black Box Testing



- Test cases are derived from formal specification of the system.
- Test case selection can be done without any reference to the program design or code.
- Only tests the functionality and features of the program.
  - Not the internal operation.
- Also known as *functional testing*.
- Advantages
  - Test case selection is done before the implementation of a program.
  - Help in getting the design and coding correct with respect to the specification.

White Box Testing

- Test cases are derived from the actual structure or code for the program.
- Also known as *structural testing or glass-box testing.*



- Advantages
  - Tests the internal details of the code;
  - Checks all paths that a program can execute.

- Limitations
  - Wait until after designing and coding the program under test in order to select test cases.

Questions ?