Slides #19

# Testing

Sections 8.0-8.4

CMPT 276

© Dr. B. Fraser



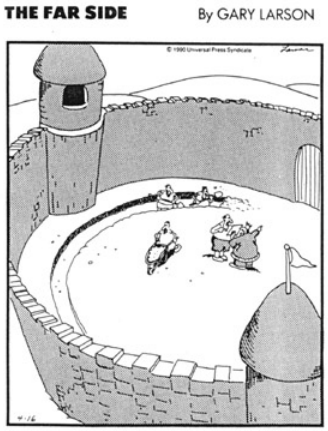
## Topics

1) What is the purpose of testing?
2) What types of tests do we use?
   a) Development testing
      - Unit Testing
      - Component Testing
      - System Testing
      Test-driven development
   b) Release testing
   c) User testing



## Verification vs validation

- Verification:
  - Software should be free of defects. (defect testing).
- Validation:
  - Software should do what user really requires.




Suddenly, a heated exchange took place between the king and the moat contractor.



## Validation and defect testing

- Validation:
  - Show system performs correctly when given data that matches its expected use.
  - A successful validation test shows that...
- Defect Testing:
  - Expose defects by testing obscure inputs which may not normally be used.
  - A successful defect test...
- Testing can reveal the presence of errors...

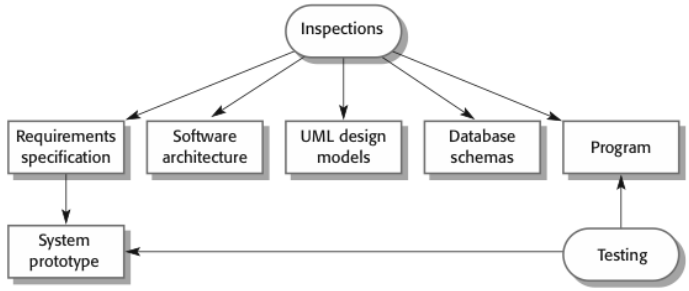
# V & V confidence

- Aim of V & V:
  - establish confidence that the system is...
- Level of "fitness" depends on:
  - Software purpose
    - The level of confidence required depends on...
  - User expectations
    - Users may have low expectations of certain kinds of software.



# Inspections and testing

- Software testing is...
  - Execute the system and observe its behaviour.
- Software inspection is...
  - Analyze the system's code, designs, or tests to look for problems.






# Code review

- Inspect the source code for defects.
  - Sometimes required before every checkin; sometimes just for important/high-risk features.
  - Done by another developer or a separate team.
- Check for:
  - (variable names, ...)
  - Error handling, memory leaks, thread-safe.
  - Array out of bounds, uninitialized variables, missing break statements, if(x=1){...} or while (true); {...}
- Code reviews have been shown to be an effective technique for discovering program errors.



# Inspections and testing

- Inspection advantages:
  - Can look for defects plus... (portability, maintainability, ...)
  - Can consider all possible paths through the code; testing may not exercise them all.
- Inspections and testing are complementary verification techniques.
  - Both should be used during the V & V process.
  - Inspections cannot check
    - performance, usability, etc.

# Stages of testing

- 
  - Tests during development to discover bugs.
  - Can use test driven development.
- 
  - A separate team tests a complete version of the system before it is released to users.
- 
  - System users test the system in their own environment.



Development Testing



# Development Testing

- Development testing:
  testing carried out by the development team.
  - 
    - test the functionality of objects or methods.
  - 
    - test the integrated interface of several combined objects.
  - 
    - test component interactions of the system as a whole.



# Unit testing

**Development Testing:**
- **Unit Testing**
- Component Testing
- System Testing

- Unit testing is the process of testing...
  - It is a defect testing process.
- Units may be:
  - Individual functions or methods within an object
  - Classes with several attributes and methods
  - Composite components with defined interfaces.
- Complete test coverage of a class involves:
  - 
  - Exercising the object through...

# Effective unit tests



- Unit testing should be..
- Test unit for:
  - Does what it's supposed to when used correctly.
  - No defects for extreme or invalid inputs.
- Testing strategies
  -
    - group input values which are "similar"
    - test based on these groupings.
  -
    - use guidelines to choose test cases.
    - guidelines cover common programming errors.



# Partition testing



- Identify groups, or regions of values in the input data and output results which...
- Ex: Multiplying two integers.
  - Input: Positive vs negative input values
  - Output: Positive vs negative result.
- Each of these groups is an...
  - Program behaves in an equivalent way for each group member.
- Test cases should be chosen from each partition.
  - test the extremes of the partitions...
  - test a middle value of the partition



# Equivalence partitioning



- Identify the equivalence groups:

```
/** Return a grade based on the percent:
 *   50 to 100 = 'P'
 *   0 to <50 = 'F'
 *   otherwise throw an exception.
 */
char assignGrade(int percent);
```



# General testing guidelines

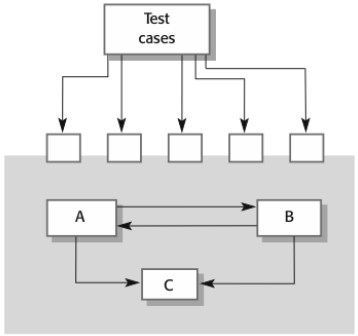


- Choose inputs that...
- Design inputs that cause input buffers to overflow;
- Force calculation results to be too large or too small: (overflow & underflow).
- Testing With Arrays:
  - Different # elements. Ex..
  - If working with a specific element, put that element..

# Component testing



- Components are made up of...
  - Each object has been unit tested,
  - But their integration has not been tested.
- Test the functionality of these...
  through the defined component interface.




Skip



# Interface testing



- Want to find faults due to interface errors or invalid assumptions about interfaces.
  - Consider component X calling component Y:
- 
  - X makes an error in its use of Y's interface
    e.g. parameters in the wrong order.
- 
  - X makes assumptions about Y's behaviour which are incorrect.
    e.g. ... m/s; m vs feet, nautical miles vs miles.

Skip



# Interface testing guidelines



- Guidelines
  - Tests with parameter values at extreme ends of their ranges.
  - Always test pointer parameters with...
  - Design tests which cause the component to fail.

Skip



# System testing



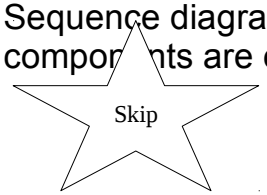
- System testing during development involves:
  - integrate components to create a version of the system,
  - and test the integrated system.
- System testing checks that components...

Skip

# System and component testing

**Development Testing:**
- Unit Testing
- Component Testing
- **System Testing**

- System testing vs component testing
  - Both tests integrated parts together.
  - System testing includes integrating parts made by (off the shelf too).
- Use-case testing
  - Use-cases from requirements/design...
  - Sequence diagrams from use-case can show which components are executed.






# Test driven development



# Test-driven development

- Test-driven development (TDD):
- Idea:
  - Write a test that fails, add code to make it pass.
- TDD started with agile methods; however, it can also be used in plan-driven development processes.



# Process

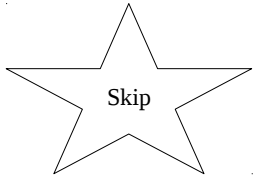
1. Find some small functionality to implement
   - 2-3 lines of code.
2. Write the JUnit test for it.
3. Run all tests
   - Will give you...
4. Write code to fix the bug.
5. Run all tests, hopefully gives you a...
   - Refactor as required.
6. Goto 1.

# Benefits of test-driven development

- 
  - Each line of code written to satisfy a test, so all lines are tested.
- 
  - Run old tests after new changes to prove have not broken previous features.
  - A regression test suite is continually developed as a program is developed.
- 
  - When a test fails, it should be obvious where the problem lies (the new stuff!).
- 
  - The tests themselves are a form of documentation that describe what the code should be doing.



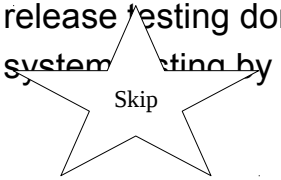
# Release testing






# Release testing

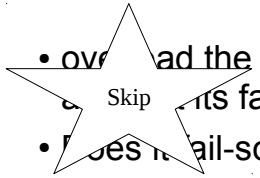
- Release testing:
  testing a system release which is intended for...
- It is validation testing, not defect testing
  - show that the system delivers required functionality,
  - 
- Release testing is a form of system testing
  - release testing done by a separate team.
  - system testing by the development team is...






# Types of release testing

- Requirements based testing
  - examine each requirement and...
- Scenario Testing:
  - walk through a full scenario as the user.
- Performance Testing:
  - test the system's performance and reliability.
  - 
    - ov ad the system to find its limits,
      its failure behaviour
    - oes it ail-soft, or go down in flames?



User testing



## User testing

- User (or customer) testing
  - Have a real user use the system.
- Needed because:
  - User can how to use the software.
    - It's eye-opening to sit down and watch someone else try and use your software!
  - User's environment can disrupt the process of using the software.
    - User interrupted by others, hardware breaks-down, printer out of paper.



## Types of user testing

-
  - User comes in to try software at the developer's site.
-
  - A release of the software given to users to try at their site.
-
  - Customer tests a system to decide whether or not it is ready for deployment.



## Summary

- Testing can only show the presence of errors;
- Software developers test during development;
  - separate team tests before release to customer.
- Development testing includes:
  - unit testing: test individual objects and methods;
  - component testing: test related groups of objects;
  - system testing: test partial or complete systems.
- Test Driven: all code written to resolve a red-bar.
- Release testing: validate features.
- User testing: alpha, beta, and acceptance testing.