Groundwork for Understanding Analysis and Test

Dr. Paul West

Department of Computer Science
College of Charleston

January 9, 2014
Verification and Validation

- **Validation:** does the software system meet the user’s real needs?
  - are we building the right software?

- **Verification:** does the software system meet the requirements specifications?
  - are we building the software right?
Verification and Validation

Validation (of the requirements)
Includes usability testing, user feedback

Verification (of the design)
Includes testing, inspections, static analysis
Verification or Validation Depends on the Specification

Example: Elevator Response Time

Unverifiable (but validatable) spec:
If a user presses a request button at floor 1, and available elevator must arrive at floor 1 soon

Verifiable Spec
If a user presses a request button at floor 1, an available elevator must arrive at floor 1 within 30 seconds.
Verification and Validation

Actual Needs and Constraints

User Acceptance (alpha, beta test)

Delivered Package

System Test
Analysis / Review

Integration Test
Analysis / Review

Module Test

System Specifications

Subsystem Design/Specs

Unit/Component Specs

Subsystem

Unit/Components

User review of external behavior as it is determined or becomes visible

Validation

Verification
Correctness

- Correctness properties are undecidable
- The halting problem can be embedded in almost every property of interest.

Halting Problem

Wolfram: The determination of whether a Turing machine will come to a halt given a particular input program. The halting problem is solvable for machines with less than four states. ... The problem of whether a general Turing machine halts is undecidable, as first proved by Turing.
Quality Trade-offs

- Optimistic inaccuracy: we may accept some programs that do not possess the property (i.e., it may not detect all violations).
  - testing
- Pessimistic inaccuracy: it is not guaranteed to accept a program even if the program does possess the property being analyzed
  - Automated program analysis techniques
- Simplified properties: reduce the degree of freedom for simplifying the property to check

Verification and Validation

Correctness

Assignment
Simplified Property: Unmatched Semaphore Operations

Original Problem

```java
if ( .... ) {
    ...
    lock(S);
}
...
if ( .... ) {
    ...
    unlock(S);
}
```

Simplified Property

Java prescribes a more restrictive, but statically checkable construct.

```java
synchronized(S) {
    ...
    ...
}
```
Choose and complete any two Chapter 2 Exercises (pg 25-27)
Due in the dropbox by January 23, 2014 2359