Validation & Verification

John D. McGregor
Clemson University
Dept.of Computer Science
johnmc@cs.clemson.edu

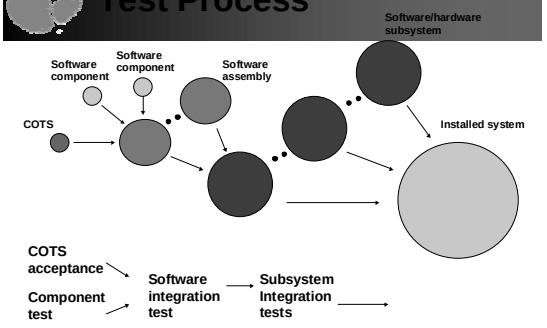
Definitions

- error - people make errors
  - programmer misinterprets a requirement
- fault - this plants a fault in the program
  - writes code that calculates the trajectory incorrectly
- failure - that results in a failure during execution
  - when the program is executed it calculates the amount incorrectly and makes a right turn

Definitions

- Test plan
  - defines the resources needed for testing activities
  - provides a schedule of activities
- Test case
  - previous state
  - inputs
  - expected outputs

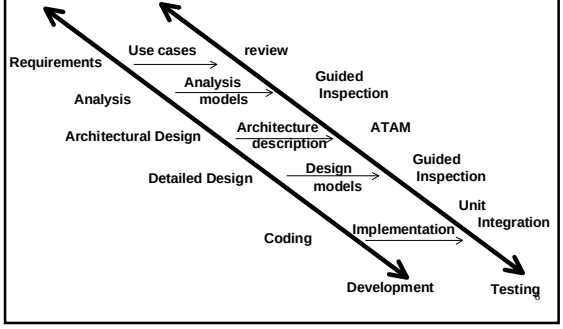
Test Process



Separate Process

- Model test
  - will a system built according to this model do what it is supposed to
- Unit test
  - does the unit produce correct answers
- Integration test
  - do the separate pieces still work when joined
- System test
  - does the system do what it is supposed to do

Interactions between processes



Other types of testing

- performance testing
- load testing
- security testing
- acceptance testing
- installation testing
- self-test

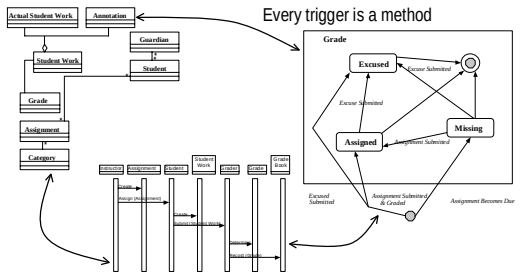
Test Planning

- After prioritizing the use cases and rationing the number of test cases, use orthogonal defect classification (ODC) to guide the selection of test cases.
- ODC triggers represent common ways failures are caused.
- Guided Inspection can use all of the review and inspection triggers and some of the system test and field operation triggers.

ODC Triggers

- Design conformance
- Understanding flow
- Concurrency
- Backward Compatibility
- Lateral Compatibility
- Rare situations
- Side effects
- Document consistencies
- Language dependencies
- Normal mode
- Recovery or exception
- Start up and shut down
- Software configuration
- Can not do
  - Stress
  - Hardware configuration

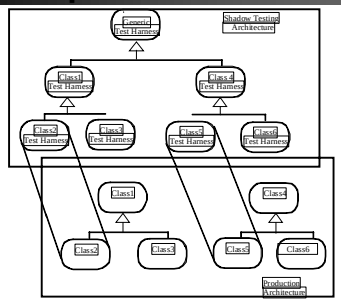
Guided Inspection



Unit Test

- Types of testing
  - functional
  - structural
  - interaction
- Responsibility of the unit's owner

Parallel Architecture for Component Test



Functional test

- Based on specification
  - pre: x > 10
  - post:
    - if x > 10 & x < 20 then return 30
    - if x >=20 then return 40
  - inv: x> 0
- test case - combination of values between pre and post conditions to yield a unique test case

Structural test

- Test cases are created based on the structure of the code

```
if (x > 0){
   …
} else if (x < 0){
   …
} else {…}
```

At least three paths so at least 3 test cases

Interaction testing

	$Method_1$	$Method_2$	$Method_3$	$Constructor_1$
$Method_1$		MM	MOM	
$Method_2$			MOM	
$Method_3$	MOM			
$Constructor_1$				
$Constructor_2$				MM

Test case & test script

```
protected boolean testScript1(){
    OUT = new Puck(/*new BricklesPlayingField(new
            BricklesGame(new BricklesView()))*/);
    boolean result = testCase1();
    logTestResult("Script1",result);
    return result;
}

protected boolean testCase1(){
  if(((Puck)OUT)._currentVelocity.getDirection() == 135){
     return true;
  }
  return false;
}
```


Test suite

```
public void functionalSuite(){
        testScript1();
        testScript2();
        testScript3();
        testScript4();

        }
```


Hierarchical Incremental Testing (HIT)

Super Class		Not Defined	Redefinable	Redefinable	Abstract	Abstract
SubClass		New Code	No New Code	New Code	No New Code	New Code
Functional Tests	Write?	Yes	No	No	No	No
	Execute?	Yes	No	Yes	No	Yes
Structural Tests	Write?	Yes	No	Yes	No	Yes
	Execute?	Yes	No	Yes	No	Yes
Interaction Tests	Write?	Yes	No	Yes	No	Yes
	Execute?	Yes	Yes	Yes	No	Yes

Integration testing

Step	Deliverable
Assemble and test each cluster	Certified clusters plus complete PACT mechanism
Assemble clusters into system increment and test	Certified increment plus complete PACT mechanism
Integrate this increment with existing increments	Partially completed system plus complete regression testing mechanism

OATS

- The basic idea is to select so that only two-way interactions are covered as opposed to higher order interactions.
- Consider three variables, each of which can take on the values 1,2 or 3. To test all possible combinations would require 27 tests.
- A pair-wise selection only requires 9 tests.

A	B	C
1	1	3
1	2	2
1	3	1
2	1	2
2	2	1
2	3	3
3	1	1
3	2	3
3	3	2

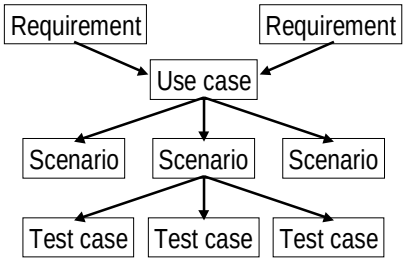
Mapping array to problem

A	B	C
1	1	3
1	2	2
1	3	1
2	1	2
2	2	1
2	3	3
3	1	1
3	2	3
3	3	2

Map →

OS	Memory Size	Database Size
Unix	64mg	20000
Unix	128mg	5000
Unix	256mg	100
Win95	64mg	5000
Win95	128mg	100
Win95	**256mg**	**20000**
W2000	64mg	100
W2000	128mg	20000
W2000	256mg	5000

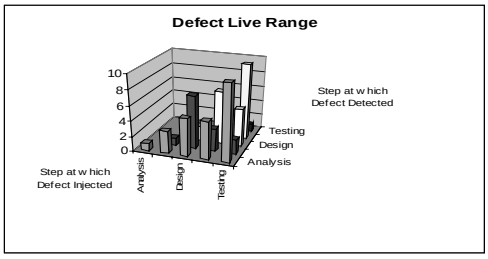
System testing



Use Case to Test Case

- Use case - The user maps a problem onto an orthogonal array
- Test case - The integration tester maps a problem involving 4 factors each with 3 levels onto an orthogonal array.

Metric



Organization

- Test manager
- Tester
- Developer
- Dedicated test team
- matrixed test team
- Integrated test team

Conclusion

- Testing can be thought of as a separate process
- These are procedures that verify a number of qualities about the product