INF 111 / CSE 121: Software Tools and Methods

Lecture Notes for Fall Quarter, 2007
Michele Rousseau
Set 8

(Some slides adapted from Susan E. Sim)

Announcements

- **Assignment 1 – updated**
  - DUE: Tuesday October 30th at 11:50p
- **Lab 3 will be posted tomorrow**
- **Reading: Van Vliet Ch. 13**

Previous Lecture

- **Started Testing**
- **NSB**

Today's Lecture

- **More on Testing**
  - Terminology
    - Failures, Faults, & Errors
  - Ariane 5 example
  - Test Process

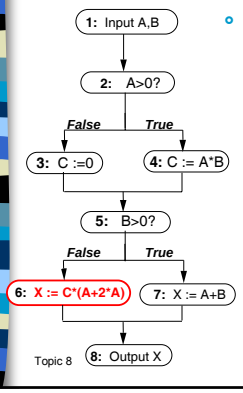
V & V

- **Validation**
  - Have we built the right system?
- **Verification**
  - Have we built the system right?

Testing Terminology

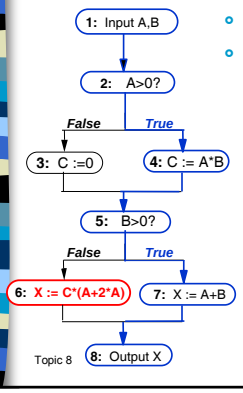
- **Failure: Incorrect or unexpected output, based on specifications**
  - System does not behave according to specifications
  - Symptom of a one or more fault
- **Fault: Invalid execution state**
  - Symptom or consequence of an error
  - May or may not produce a failure
  - May produce Many Failures
- **Error: Defect or anomaly or "bug" in source code – Human Error**
  - May or may not produce a fault

Examples: Failures, Faults, and Errors



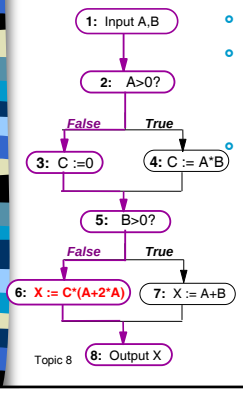
- **ERROR => Node 6 should be X:= C*(A+2*B)**

Examples: Failures, Faults, and Errors



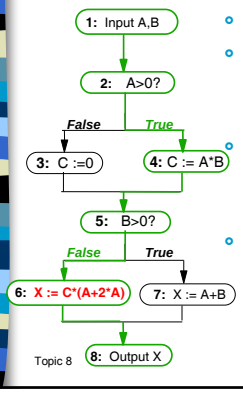
- **ERROR => Node 6 should be X:= C*(A+2*B)**
- **Error – but no Fault or Failure**
  - Inputs: **A = 2, B = 1**
  - Executed Path => (1,2,4,5,7,8)
  - Fault is not Revealed
    - Node 6 is not executed

Examples: Failures, Faults, and Errors



- **ERROR => Node 6 should be X:= C*(A+2*B)**
- **Error – but no Fault or Failure**
  - Inputs: **A = 2, B = 1**
  - Executed Path => (1,2,4,5,7,8)
  - Fault is not Revealed
    - Node 6 is not executed
- **Error – Fault – No Failure**
  - Inputs: **A = -2, B = -1**
  - Executed Path => (1,2,3,5,6,8)
  - Fault Not Revealed
    - because C = 0

Examples: Failures, Faults, and Errors



- **ERROR => Node 6 should be X:= C*(A+2*B)**
- **Error – but no Fault or Failure**
  - Inputs: **A = 2, B = 1**
  - Executed Path => (1,2,4,5,7,8)
  - Fault is not Revealed
    - Node 6 is not executed
- **Error – Fault – No Failure**
  - Inputs: **A = -2, B = -1**
  - Executed Path => (1,2,3,5,6,8)
  - Fault Not Revealed
    - because C = 0
- **Need select proper test cases**
  - Definitions of C at Nodes 3 and 4 both affect the use of C at node 6
  - Path **(1,2,4,5,6,8)** will reveal the failure
    - *but* only if B <> 0
    - e.g. Inputs: **A = 1, B = -2**

Why do we care about Errors / Faults that never show up?

- **Latent faults**
  - Can be subsumed by previous statements
  - Maybe that state is never entered
- **Software is often reused later**
- **Conditions not hit in prev. version may be accessed later**
  - Code Changes

For Example: Ariane 5

- **Capable of hurling 2 – 3 ton satellites into orbit**
- **10 years**
- **$7 Billion**
- **Would have given Europe supremacy in the commercial satellite business**

Some Slides Adapted from Sommerville

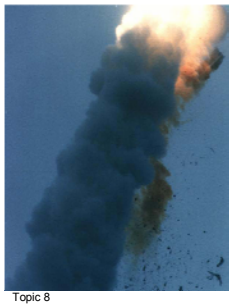
Arian 5 (2)

- **Successor to the successful Ariane 4 launchers**
- **Ariane 5 can carry a heavier payload**

Whoops!

- **40 seconds into maiden flight**
  - veers off course & self-destructed
- **39 seconds after lift off**
  - Altitude reaches 2.5 miles
  - Ariane 5 goes into self destruct
  - Carrying 5 expensive - uninsured satellites

Why?



- **Why did it go into self destruct mode?**
  - Incorrect control signals were sent to the engines and these swivelled - Ariane 5 swerved
  - Pressure in boosters and main engine
- **Why did it swerve?**
  - It was making a course correction that was not needed.

Launcher Failure

- **Why the course correction?**
  - Steering controlled by onboard computer
  - Thought course change was necessary because of numbers being displayed by the inertial guidance system
  - The numbers looked like data – impossible data- but was actually an error message

  → The guidance system had shutdown
- **Why did the guidance system shutdown?**
  - Tried to convert a 64-bit format velocity to a 16-bit format
  - Overflow error
- **What about the backup?**
  - Backup system failed too..
    - It was running the same software

In a nutshell…

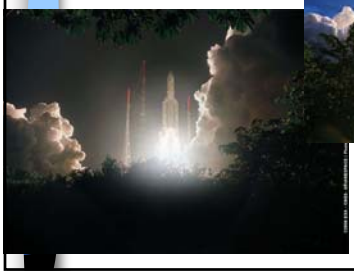
- **Software Failure**
- **Software was reused form Ariane 4.**
  - Fault was never found when testing for Ariane 4
  - Ariane 4 → Physically smaller
    - lower initial acceleration and build up of horizontal velocity than Ariane 5
  - The value of the variable on Ariane 4 could never reach a level that caused overflow during the launch period.

Avoidable?

- **The computation that resulted in overflow was not used by Ariane 5.**
- **Decisions were made**
  - Not to remove the facility as this could introduce new faults
  - No exception handling for overflows
    - Processor was heavily loaded
    - Wanted spare processorcapacity for dependability
- **Since there was no requirement → no test (not a validation error)**

Happy Ending…

- **They fixed the error and…**



Why not exhaustively test everything?

```
for (i = 0; i<100; i++) {
   if (a[i] == true ){
        System.out.println("1");
   }
   else {
   System.out.println("0");
   }
}
```

- How long would it take to test exhaustively?
  - **Possible outputs?**
  - **How long for each output?**
- 2^100 outcomes @ 10 000 000 print statements/ second = 3 x 104 years

Why not exhaustively test everything?

- Not feasible to run all those test cases
- Not feasible to validate them once they are run
  - **Need to know the output**
  - **Need to compare expected to actual**

Typical Testing Process