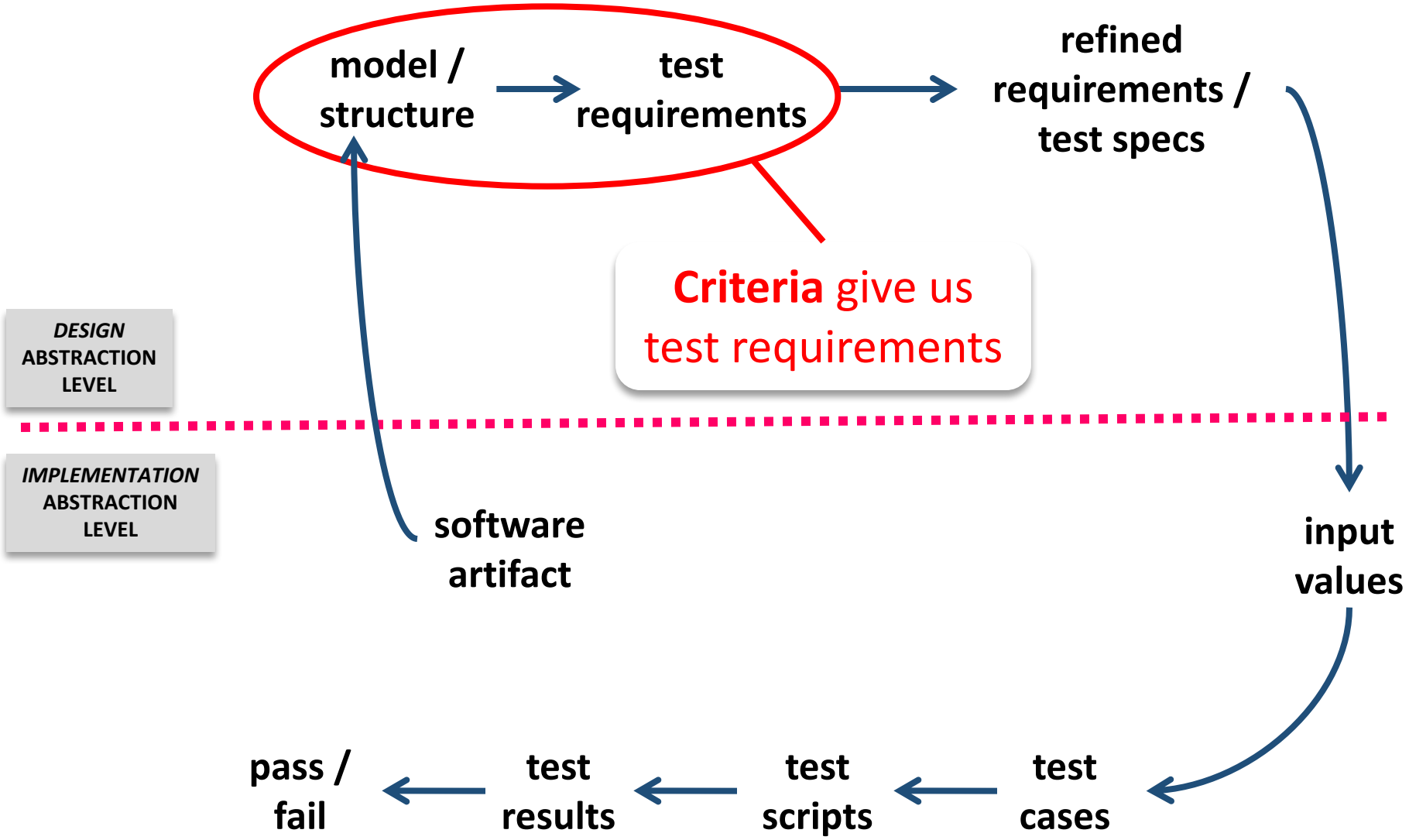
Criteria-Based Test Design

Software Testing

(3104313)

Amirkabir University of Technology
Spring 1399-1400

Model-Driven Test Design



Test Coverage Criteria

A tester's job is **simple!**

Define a model of the software, then **find ways to cover** it

Test Requirements

A specific element of a software artifact that a test case must satisfy or cover.

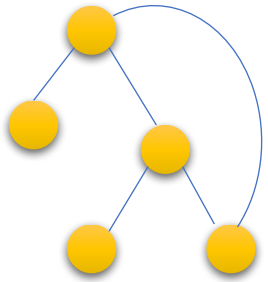
Coverage Criterion

A rule or collection of rules that impose test requirements on a test set.

(not X or not Y) and A and B

```
If (x < y)
      z = x - y
else
      z = 2*x
```

A: {0, 1, >1}
B: {600, 700, 800}
C: {swe, cs, isa, infs}



Four ways to model a software artefact

Coverage

Given a set of test requirements *TR* for coverage criterion *C*, a test set ***T* satisfies *C*** coverage if and only if for every test requirement *tr* in *TR*, there is at least one test *t* in *T* such that *t* satisfies *tr*.

- TR: Test Requirements
  - tr: one test requirement
- C: Coverage Criterion
- T: Test Set
  - t: one test

Coverage

Given a set of test requirements *TR* for coverage criterion *C*, a test set ***T* satisfies *C*** coverage if and only if for every test requirement *tr* in *TR*, there is at least one test *t* in *T* such that *t* satisfies *tr*.

- TR: Test Requirements
  - tr: one test requirement
- C: Coverage Criterion
- T: Test Set
  - t: one test

- 100% coverage is impossible in practice
- **Coverage Level**: The ratio of the number of test requirements satisfied by T to the size of TR

How are test criteria used?

How are test criteria used?

- Generator
- Recogniser

Good Coverage Criteria

?

Good Coverage Criteria

- It should be fairly **easy** to compute test requirements **automatically**
- It should be **efficient to generate** test values
- The resulting tests should reveal as **many faults** as possible

Good Coverage Criteria

- It should be fairly **easy** to compute test requirements **automatically**
- It should be **efficient to generate** test values
- The resulting tests should reveal as **many faults** as possible

- Comparison
  - How can we compare coverage criteria?

Criteria Subsumption

A test criterion ***C1* subsumes *C2*** if and only if every set of test cases that satisfies criterion *C1* also satisfies *C2*

- Must be true for every set of test cases

- Example
  - If a test set has covered every branch in a program (satisfied the branch criterion), then the test set is guaranteed to also have covered every statement

In-Class Exercise #9

- Assume the following:
  - Test criterion C1 subsumes C2
  - Test set T1 satisfies C1 on program P
  - Test set T2 satisfies C2 on program P

- Does T1 necessarily satisfy C2? Explain
- Does T2 necessarily satisfy C1? Explain
- If P contains a fault, and T2 reveals the fault, T1 does not necessarily reveal the fault. Explain.

- You have 5-10 minutes
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

Criteria Subsumption

A test criterion ***C1* subsumes *C2*** if and only if every set of test cases that satisfies criterion *C1* also satisfies *C2*

- Must be true for every set of test cases

- Subsumption is a rough approximation of **fault revealing capability**
- **Other** ways to compare criteria?

Advantages of criteria-based test design

- Criteria maximize the "**bang for the buck**"
- **Fewer** tests that are **more effective** at finding faults
- **Comprehensive** test set with minimal overlap
- **Traceability** from software artifacts to tests
  - The "**why**" for each test is answered
  - Built-in support for **regression testing**
- A "**stopping rule**" for testing—advance knowledge of **how many tests** are needed
- Natural to **automate**