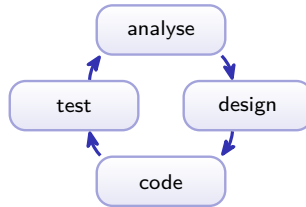


Software Development

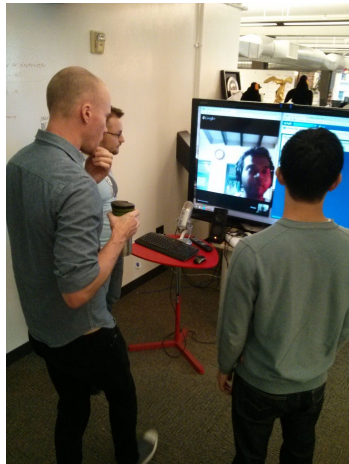
AS Computer Science ■ Topic 12

A-Level Computer Science

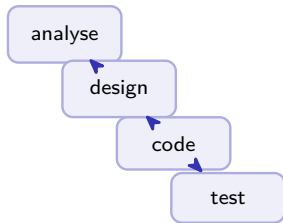


What we will cover

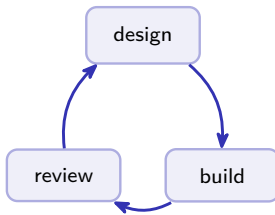
- 1 Development life cycle
- 2 Design tools
- 3 Errors
- 4 Testing
- 5 Maintenance
- 6 Recap



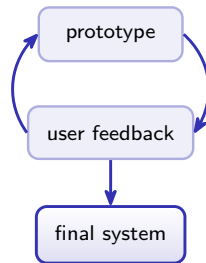
Development life-cycle models



waterfall: linear



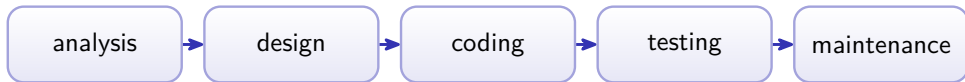
iterative: repeated passes



RAD: quick prototypes

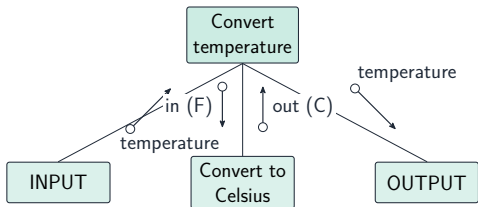
Waterfall 瀑布模型: stable needs. **Iterative** 迭代模型/**RAD** 快速应用开发/**Agile** 敏捷: needs discovered or changing.

The standard stages



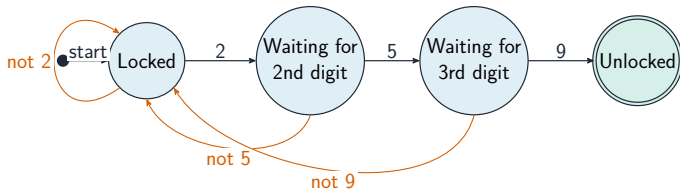
analysis = **what**; design = **how**; then code, test and maintain.

Design tools

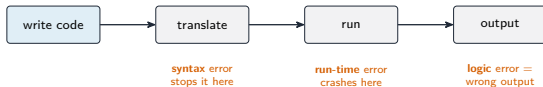


○ → parameter passed along a link (data couple)

A **structure chart** 结构图 splits the system into modules and shows the **parameters** passed between them.

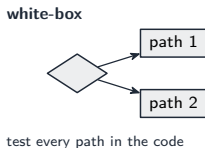
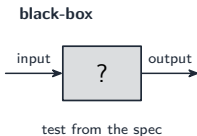


Three kinds of error



- **syntax** 语法错误 – caught at translation
- **run-time** 运行时错误 – crashes mid-run
- **logic** 逻辑错误 – runs, but **wrong output**

Testing methods



- **black-box** 黑盒测试 – from the spec
- **white-box** 白盒测试 – every code path
- **alpha** α 测试 (in-house),
beta β 测试 (real users),
acceptance 验收测试 (customer)

Test strategy vs test plan

test strategy 测试策略

the **high-level approach**:
which kinds of testing,
who runs them, and when

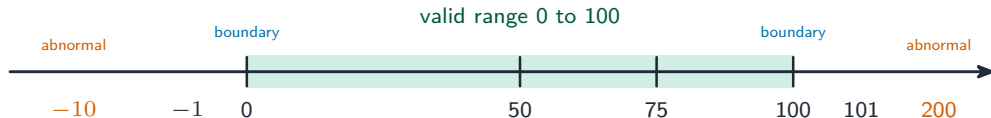
test plan 测试计划

the **detailed list** of tests:
input data, expected output,
actual output

The plan's "expected vs actual" columns are exactly how logic errors get caught.

Choosing test data

Four kinds of test data for a 0–100 field:



normal (50, 75) inside; extremes (0, 100) the accepted edges; abnormal (–1, 101, –10, 200) rejected

normal 正常数据

50, 75
well inside

extreme 极端数据

0, 100
edges, still accepted

abnormal 异常数据

–10, 200
must be rejected

boundary 边界数据

0/–1, 100/101
both sides of the line

Worked example: catch the logic error

“Award a pass for marks of 50 or above”:

```
INPUT Mark
IF Mark > 50 THEN
  OUTPUT "Pass"
ELSE
  OUTPUT "Fail"
ENDIF
```

the bug: > should be >=

Mark	expected	actual	
75	Pass	Pass	✓
30	Fail	Fail	✓
50	Pass	Fail	X

Only the **boundary value** 50 exposes it – the normal values both pass.

A logic error runs without crashing – compare **expected vs actual** output, and always test the exact boundary.

Maintenance and amending code

perfective
improve features
or speed

adaptive
keep it working in a
changed environment

corrective
fix bugs found
in use

- **perfective** 完善性维护 – improve
- **adaptive** 适应性维护 – new environment
- **corrective** 纠正性维护 – fix bugs

Amending a program: read it, make the **smallest** change, update callers, then

regression test 回归测试: check the new behaviour *and* that you broke nothing old.

Topic 12 – key takeaways

1

Life cycle

waterfall vs iterative/RAD/agile

2

Errors

syntax, run-time, logic

3

Testing

black/white box; normal, extreme, abnormal data

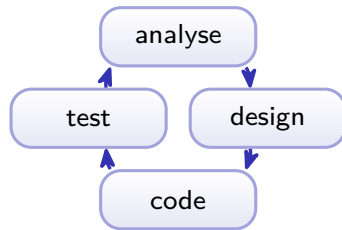
4

Maintenance

perfective, adaptive, corrective;
regression test

Check yourself

- 1 Which error gives wrong output but does not crash?
- 2 Which testing uses only the specification?
- 3 Test data 0 and 100 for a 0–100 field – which kind?
- 4 What is regression testing?



Answers 1. a logic error 2. black-box 3. extreme 4. re-test old features still work