ISTQB® Foundation: Testing throughout the Software Development Lifecycle

UNDERSTANDING SDLC MODELS

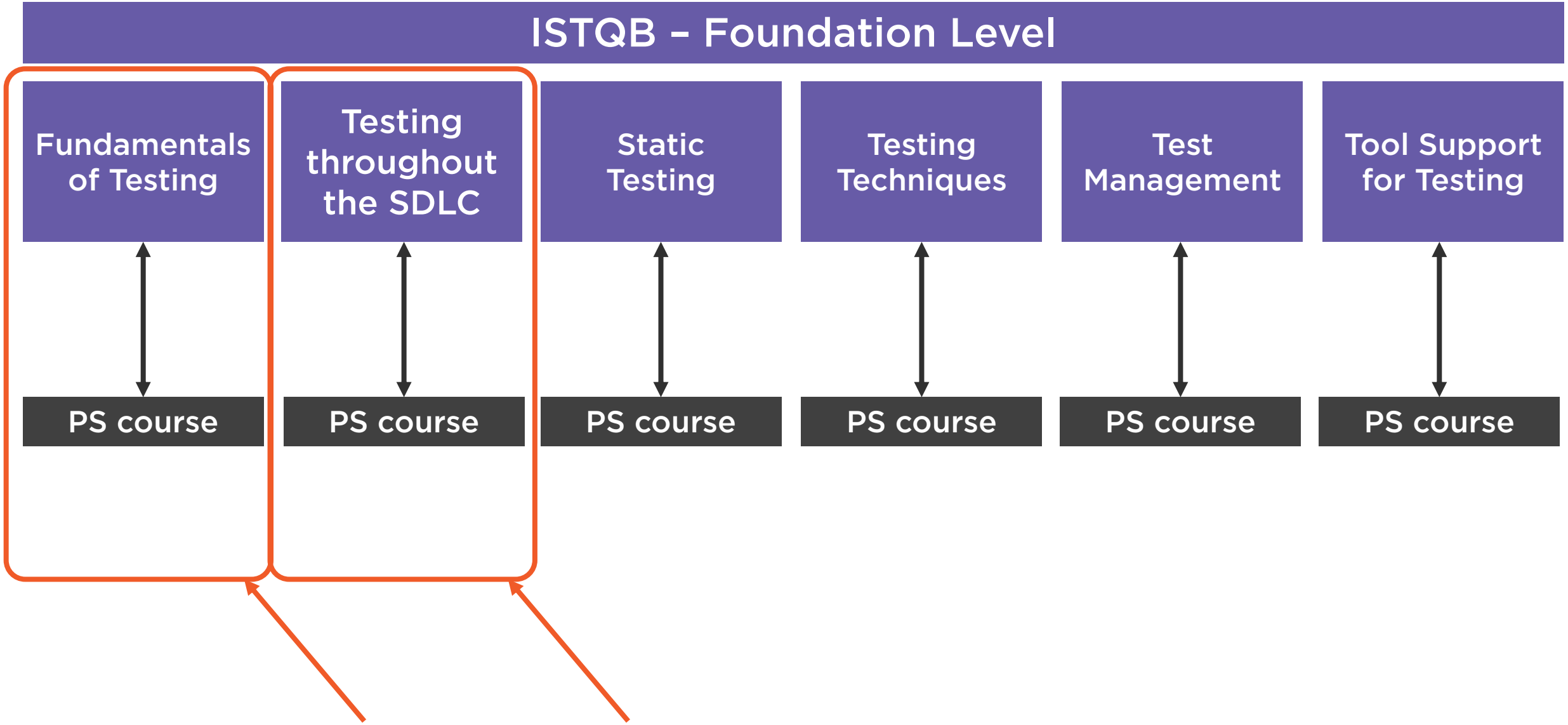
Andrejs Doronins
TEST AUTOMATION ENGINEER

ISTQB

International Software Testing Qualifications Board

ISTQB – Foundation Level

Fundamentals of Testing	Testing throughout the SDLC	Static Testing	Testing Techniques	Test Management	Tool Support for Testing


ISTQB – Foundation Level
Fundamentals of Testing
Testing throughout the SDLC
Static Testing
Testing Techniques
Test Management
Tool Support for Testing
PS course
PS course
PS course
PS course
PS course
PS course

Who This Course Is For

Aspiring Tester

Junior Tester

Test Automation Engineer

Course Overview

SDLC models

Test levels

Test types

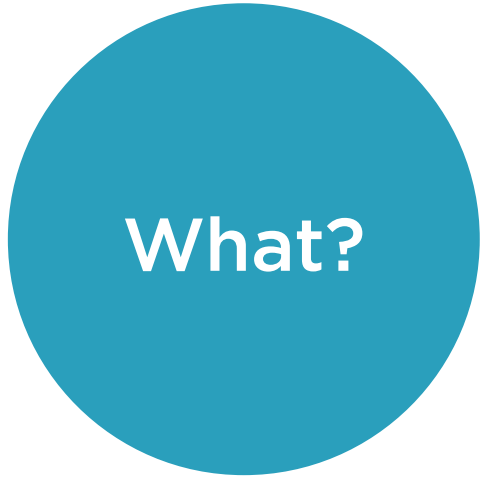
Maintenance testing

Overview

Define SDLC

Broad SDLC categories

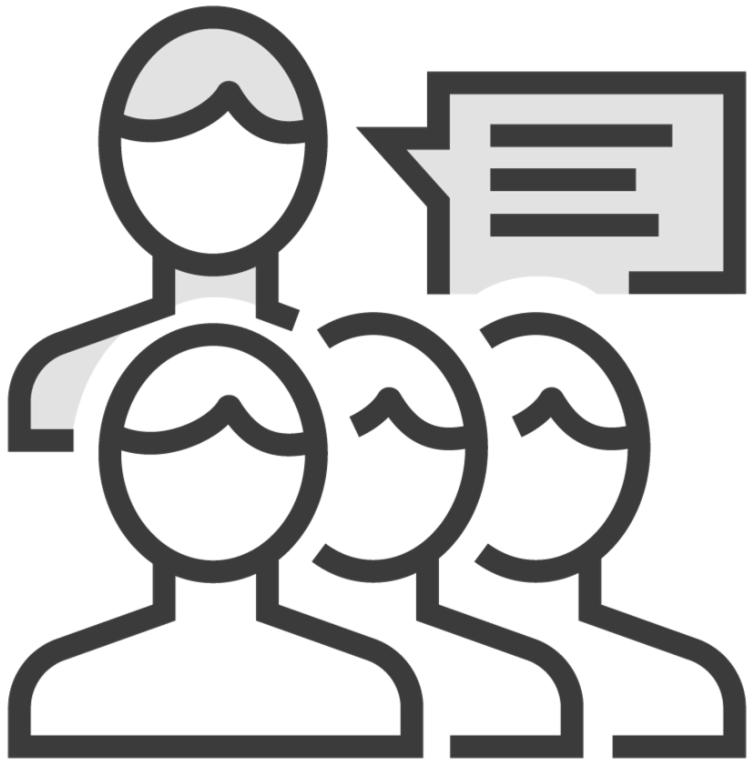
SDLC



SDLC

A systematic plan that defines a process and the what? who? when? and how? to deliver software in an organized and efficient way

SDLC



- Offers a basis for project planning and control
- Helps every team player know their role
- Ensures smooth progress
- Increases development speed
- Keep costs low

Not all lifecycle models are created equal

SDLC

A model that describes the types of activity performed at each stage in a software development project, and how the activities relate to one another logically and chronologically

SDLC Models

Sequential

Waterfall, V-Model

Iterative

RUP, Scrum, Kanban, Spiral

SDLC Stages

Analysis

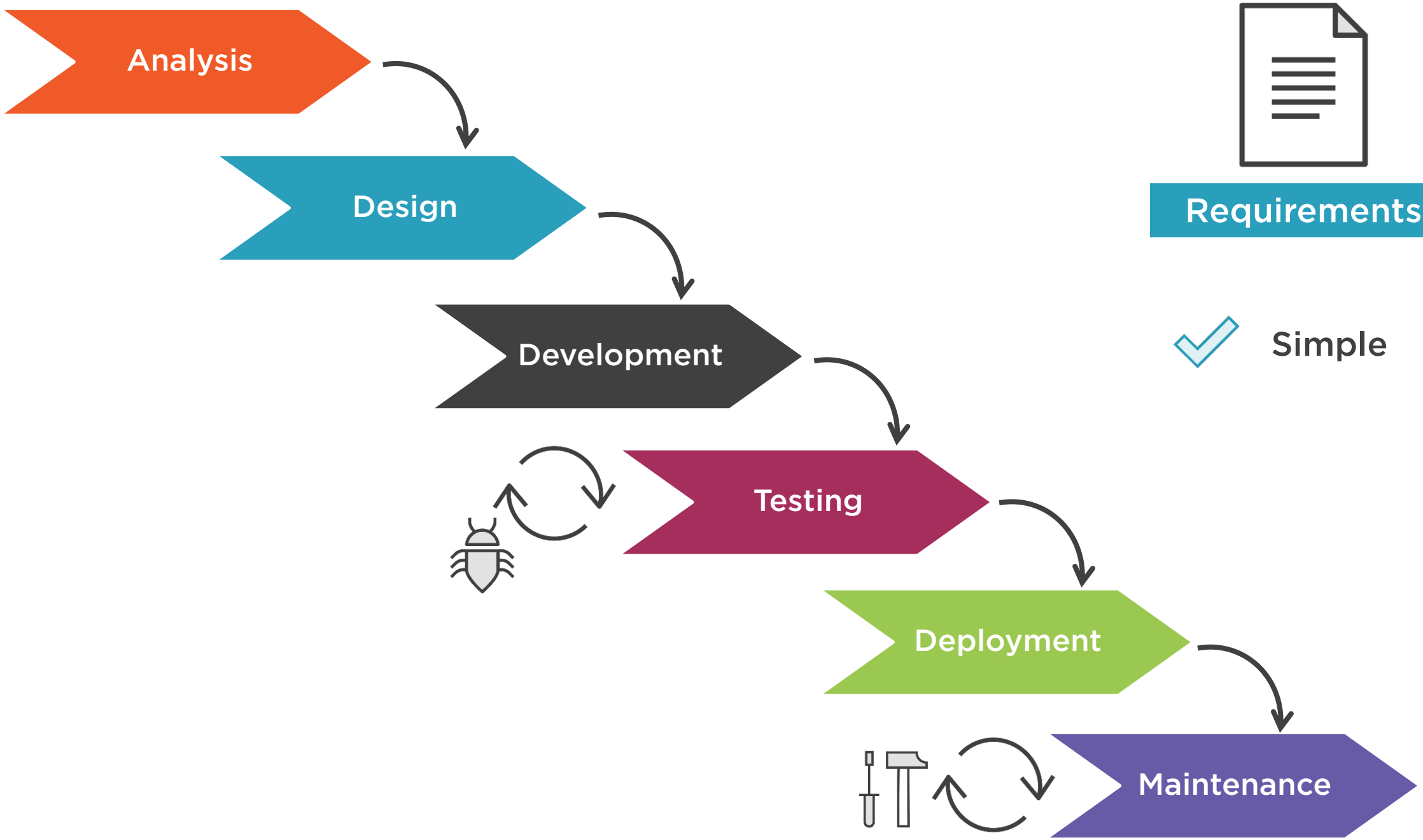
Design

Development

Testing

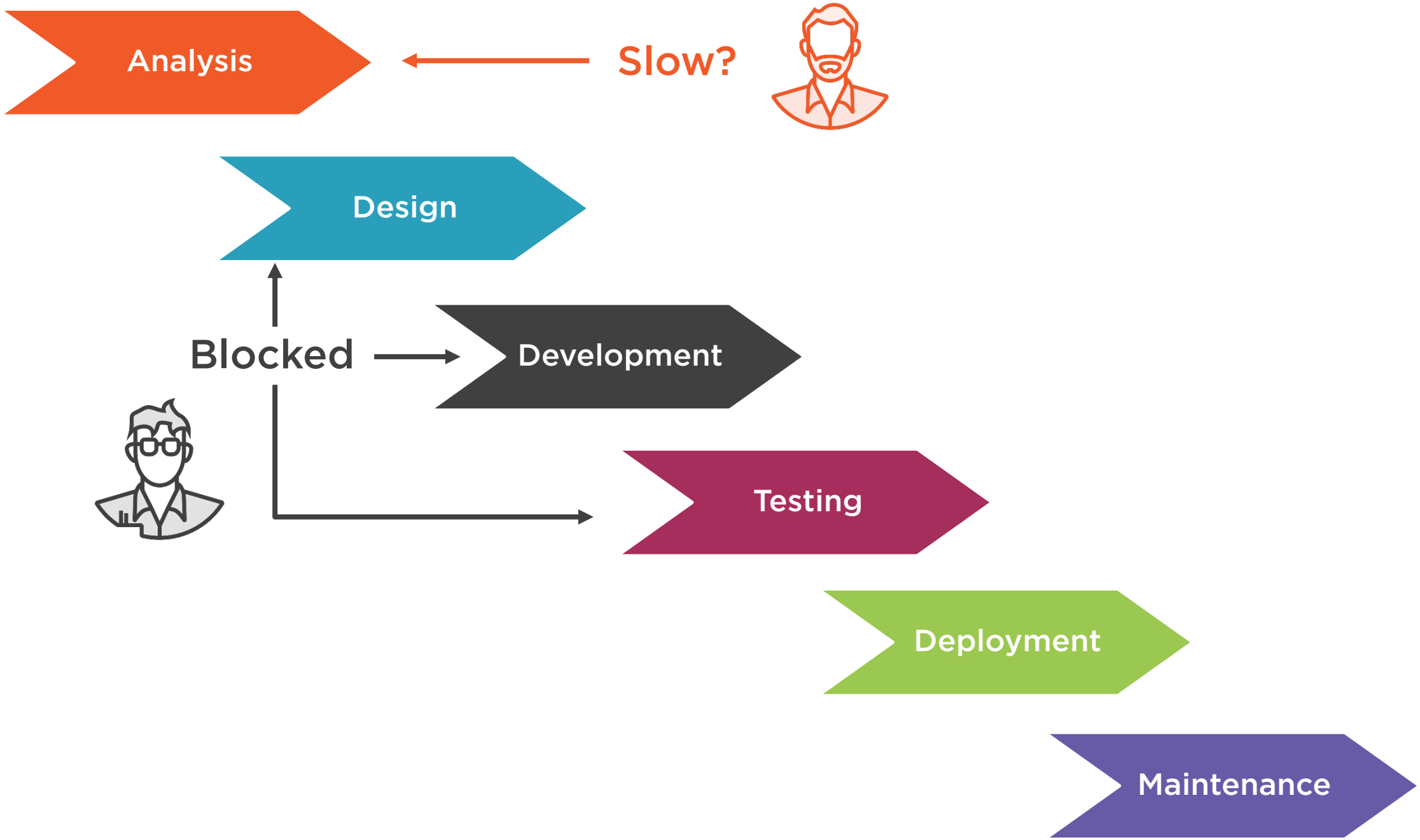
Deployment

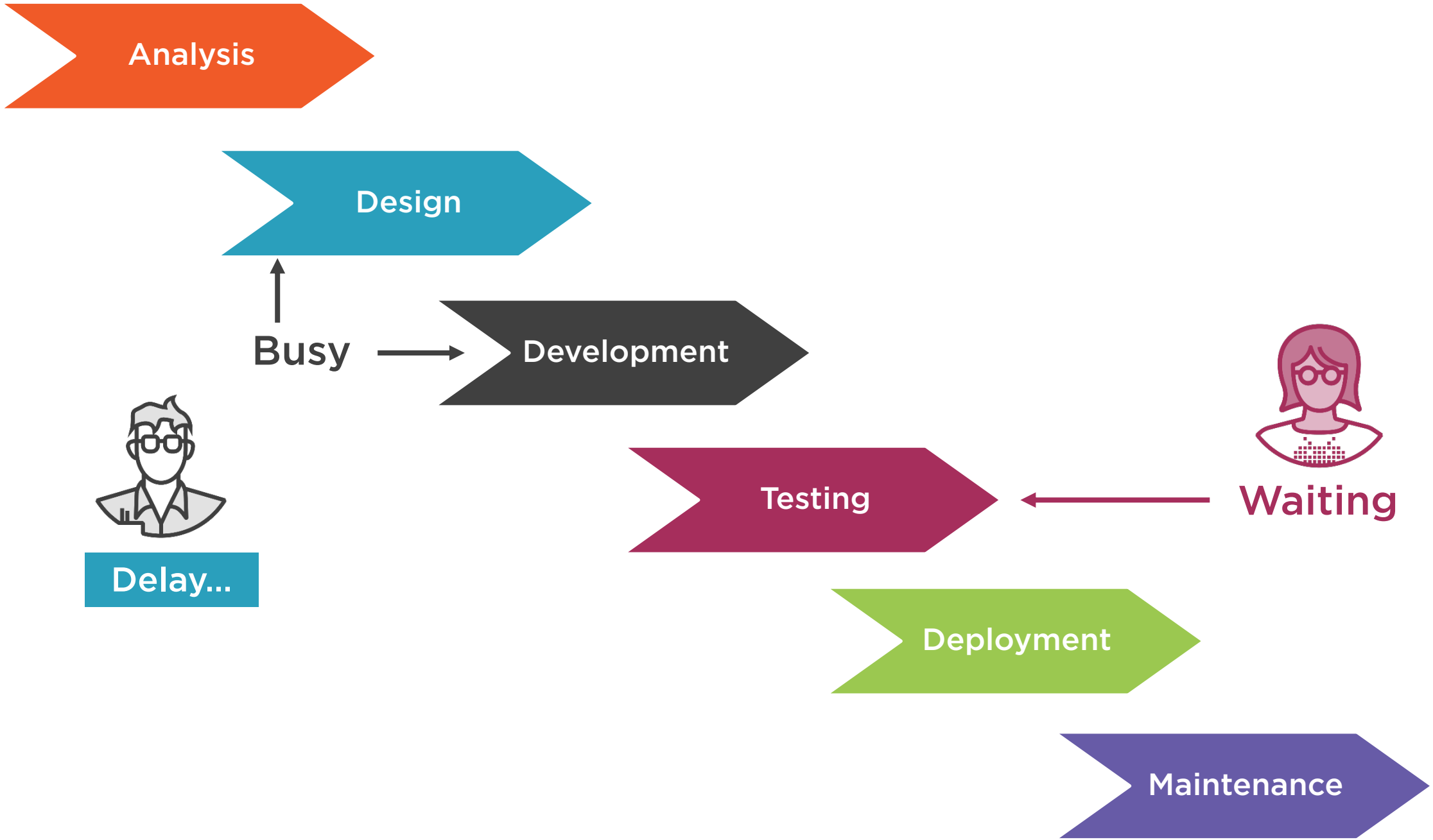
Maintenance

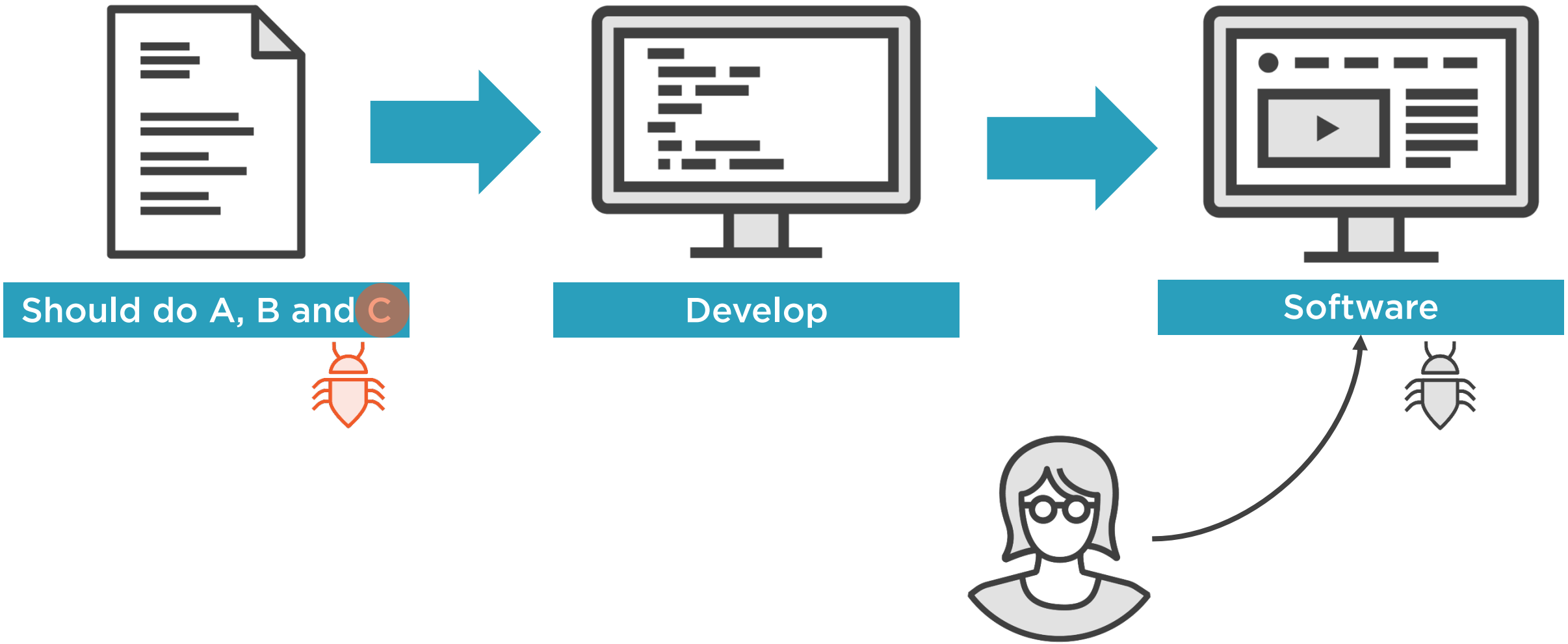

Analysis
Design
Development
Testing
Deployment
Maintenance
Requirements
Simple

Waterfall Disadvantages

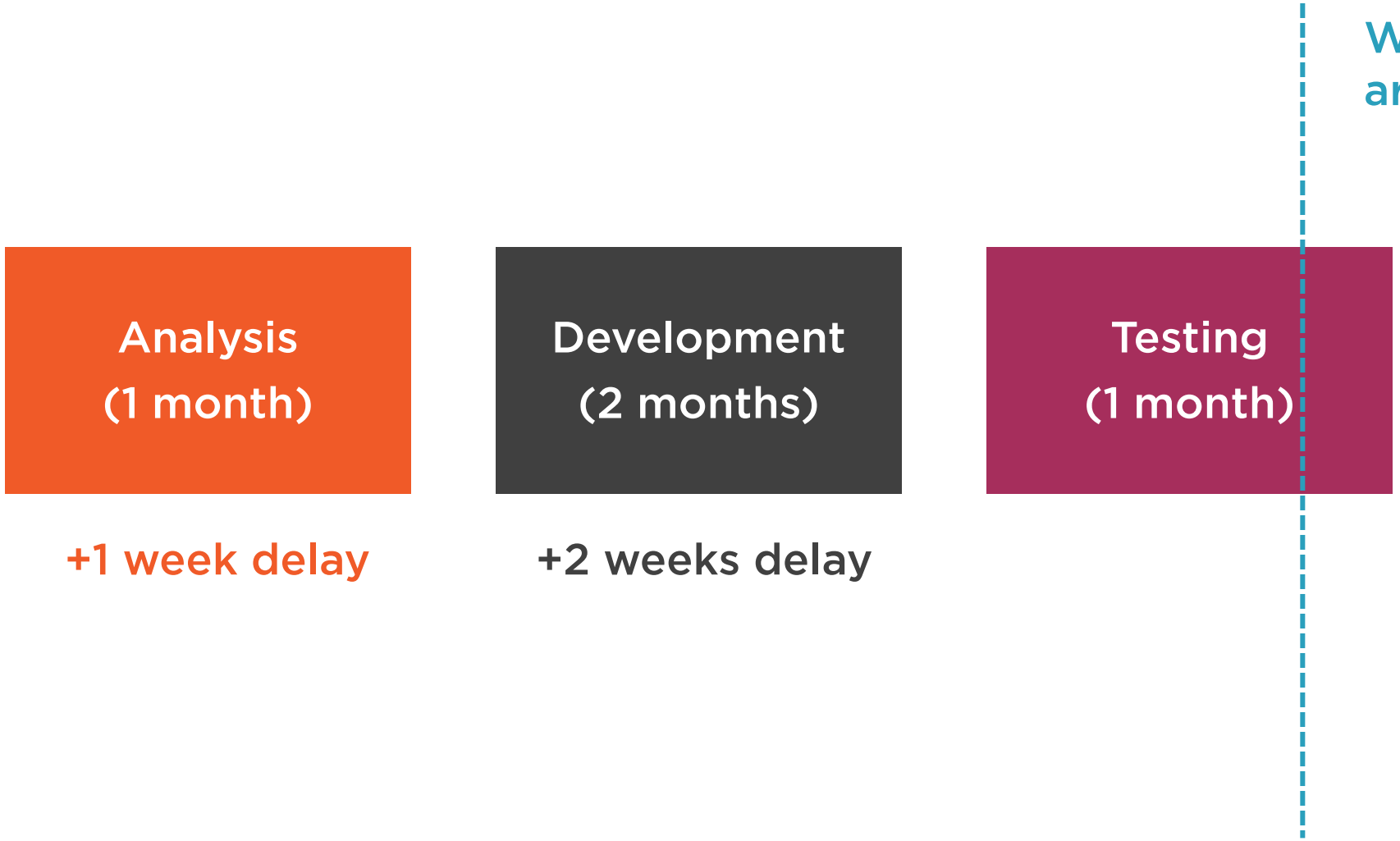
- Can be slow and inefficient
- People wait for others to finish
- Any phase can become a bottleneck
- (Very) late feedback
- Additional pressure on testers


Analysis
Slow?
Design
Blocked
Development
Testing
Deployment
Maintenance


Analysis
Design
Busy
Development
Testing
Waiting
Deployment
Maintenance
Delay...

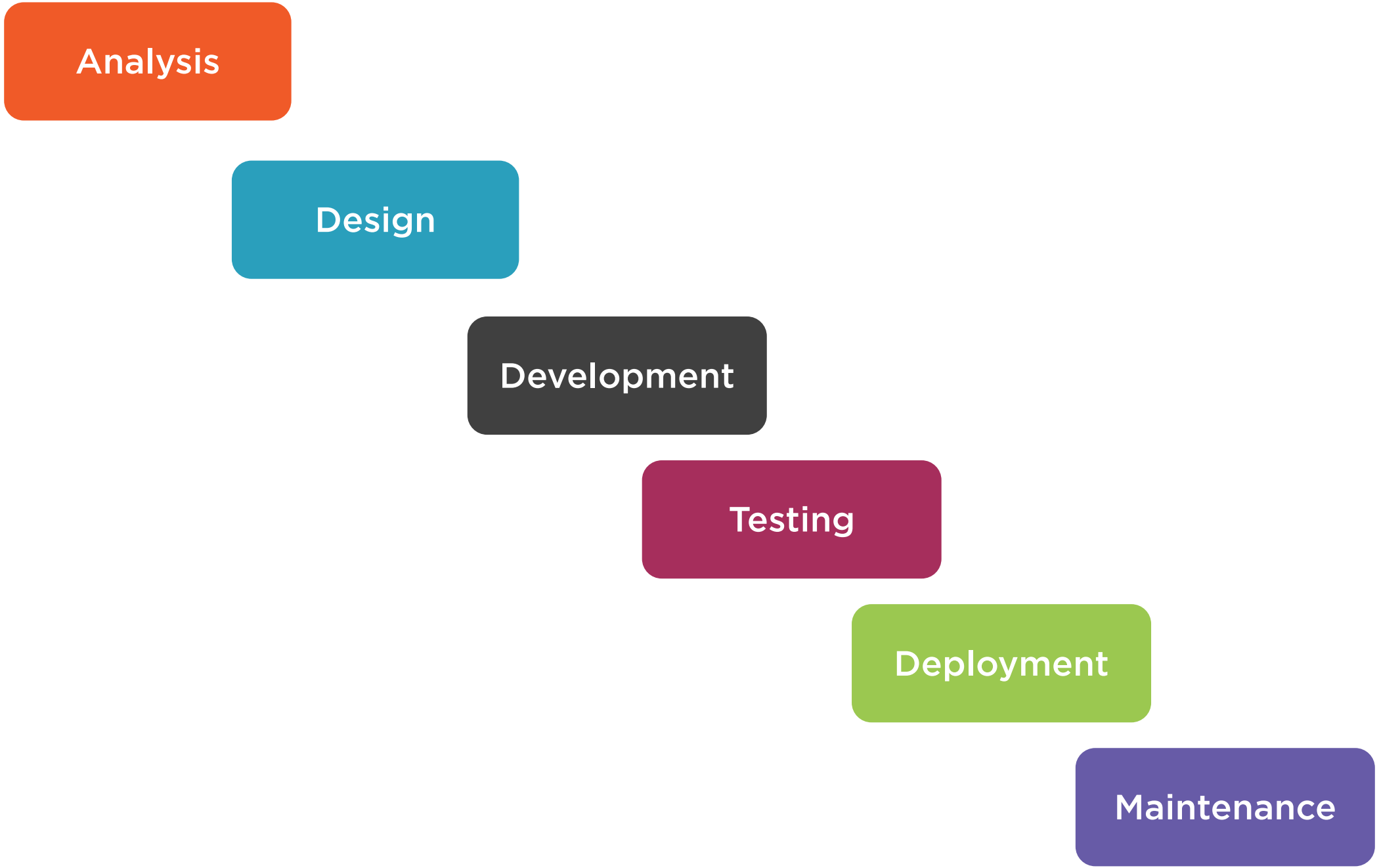

Should do A, B and C
Develop
Software

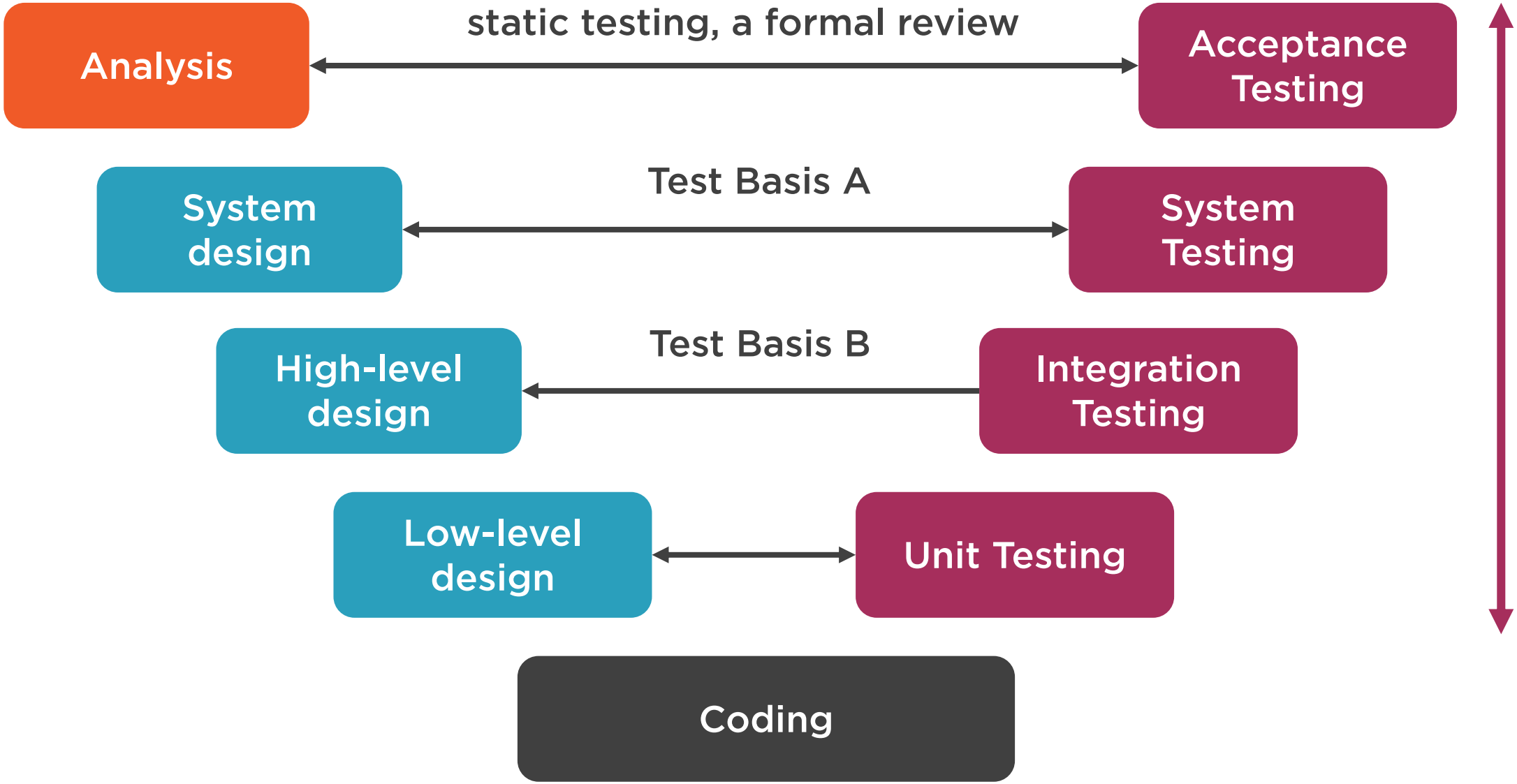
The sooner you discover an issue, the better.


What testers are left with
Analysis (1 month)
Development (2 months)
Testing (1 month)
+1 week delay
+2 weeks delay

Can we extend the deadline?

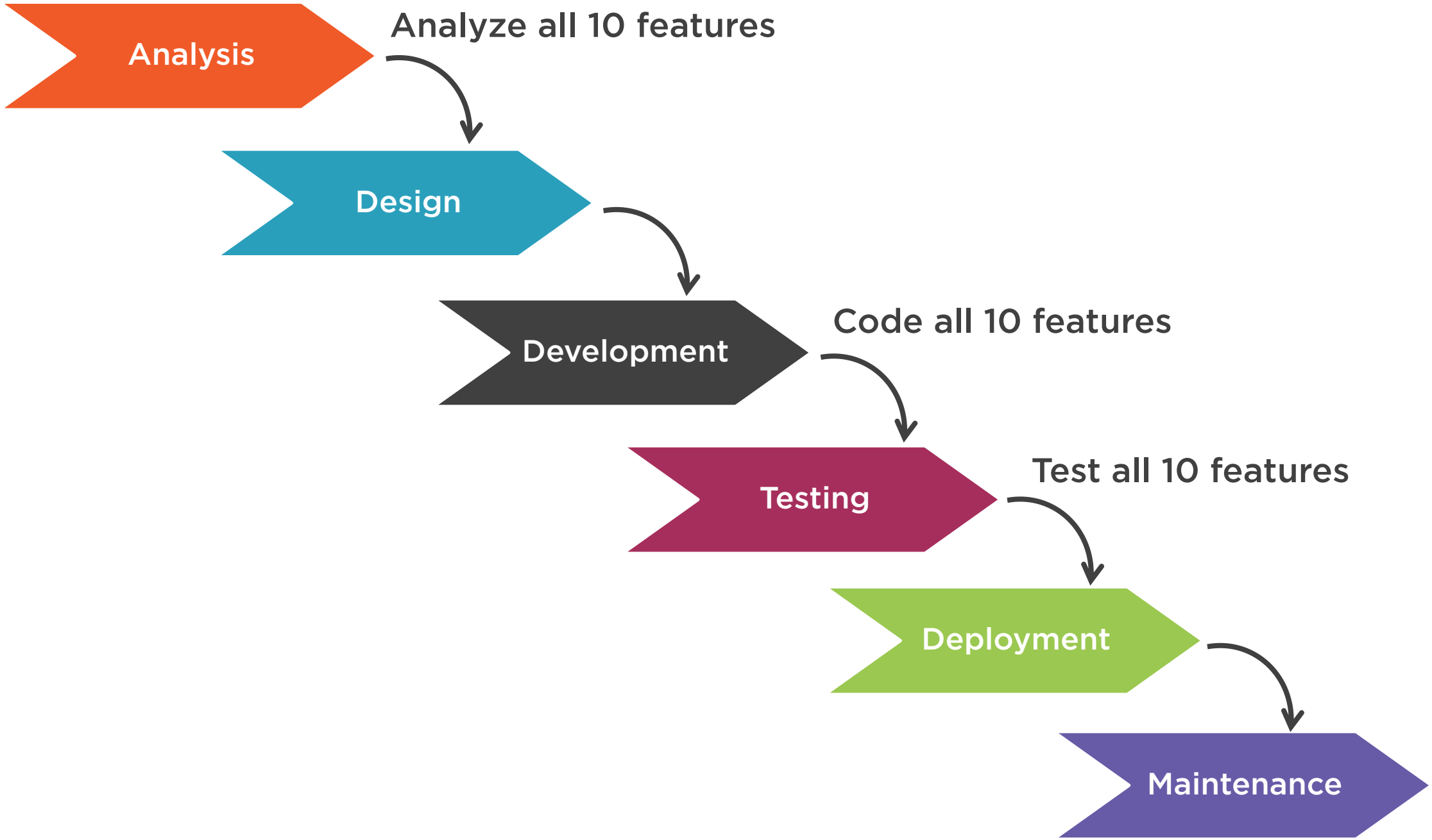
Ha ha! No!

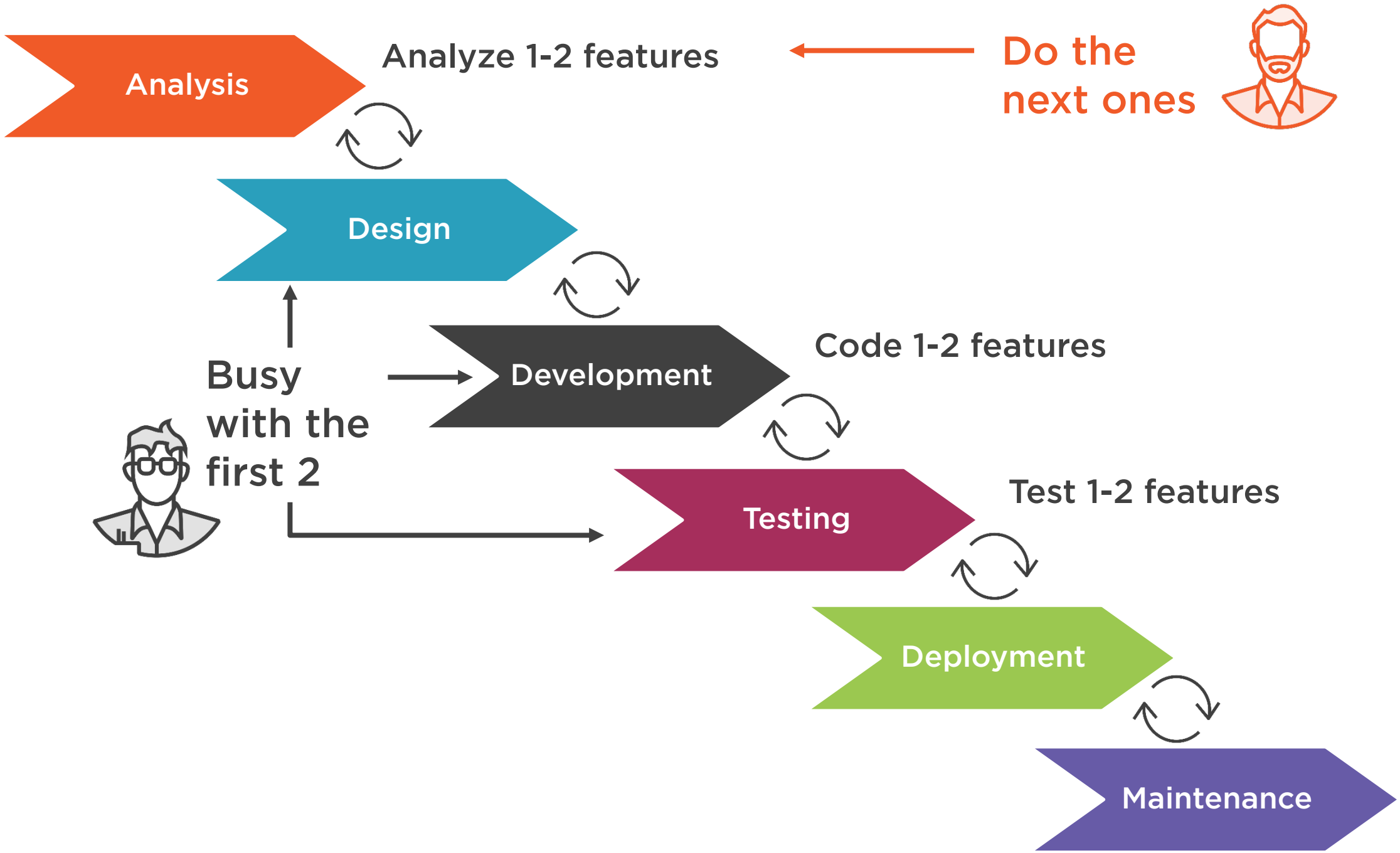

Analysis
Design
Development
Testing
Deployment
Maintenance


Analysis
static testing, a formal review
Acceptance Testing
System design
Test Basis A
System Testing
High-level design
Test Basis B
Integration Testing
Low-level design
Unit Testing
Coding

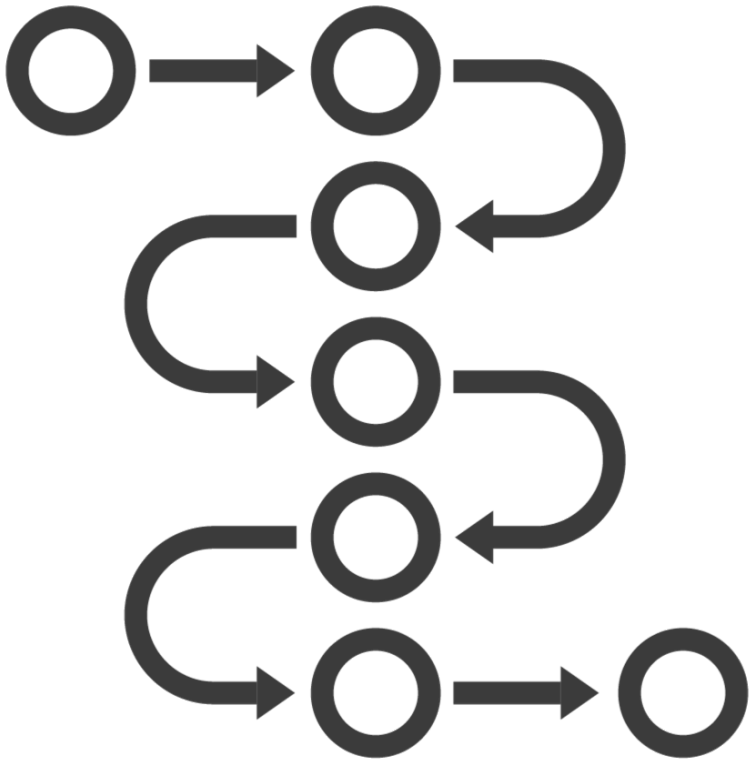
Test Basis

Material that you use to derive tests from – formal requirements, technical specifications, code, etc.


Analysis
Analyze all 10 features
Design
Development
Code all 10 features
Testing
Test all 10 features
Deployment
Maintenance


Analysis
Analyze 1-2 features
Do the
next ones
Design
Busy
with the
first 2
Development
Code 1-2 features
Testing
Test 1-2 features
Deployment
Maintenance

Iterative Approach



Break up the work:

- By feature(s)
- By time cycle (often fixed duration)

What can we achieve in 2 weeks?

Weeks #1-2: two small features

Weeks #3-4: one bigger feature

Sequential vs. Iterative

Sequential

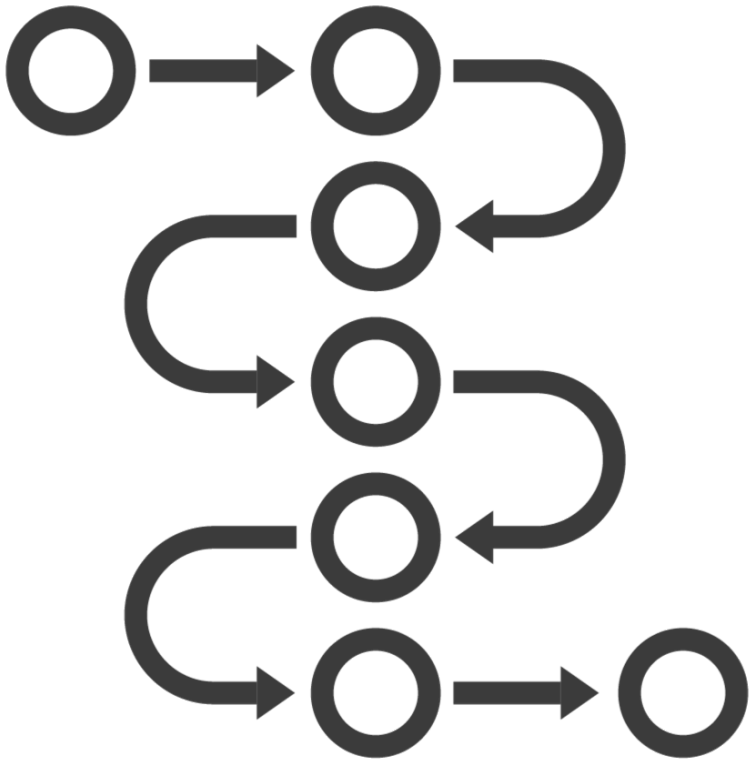
One major activity at a time

Iterative

Everything within a fixed time frame

Deliver something usable fast

Iterative Models



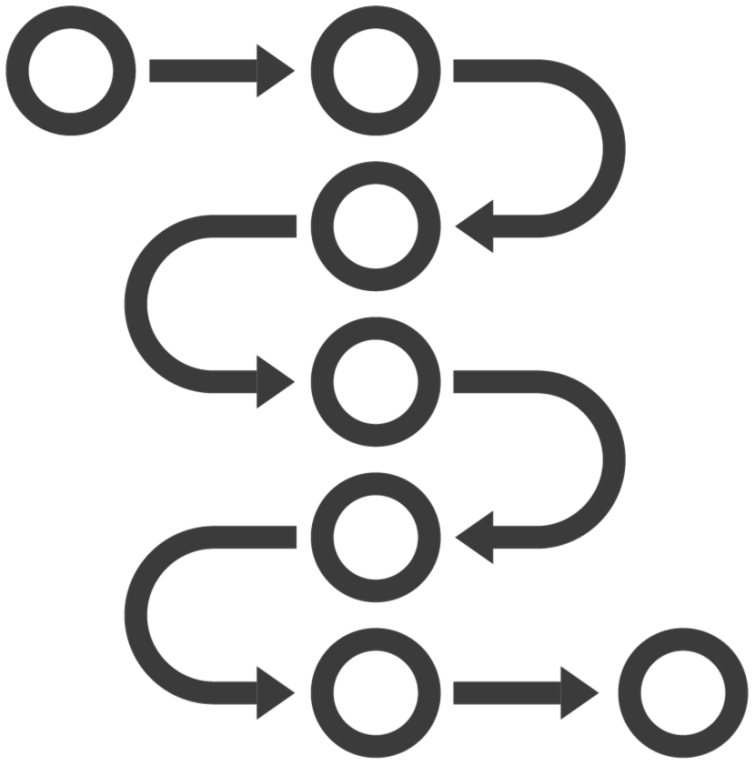
Rational Unified Process

Agile: an umbrella term

- Scrum
- Kanban

Spiral

Iterative Models



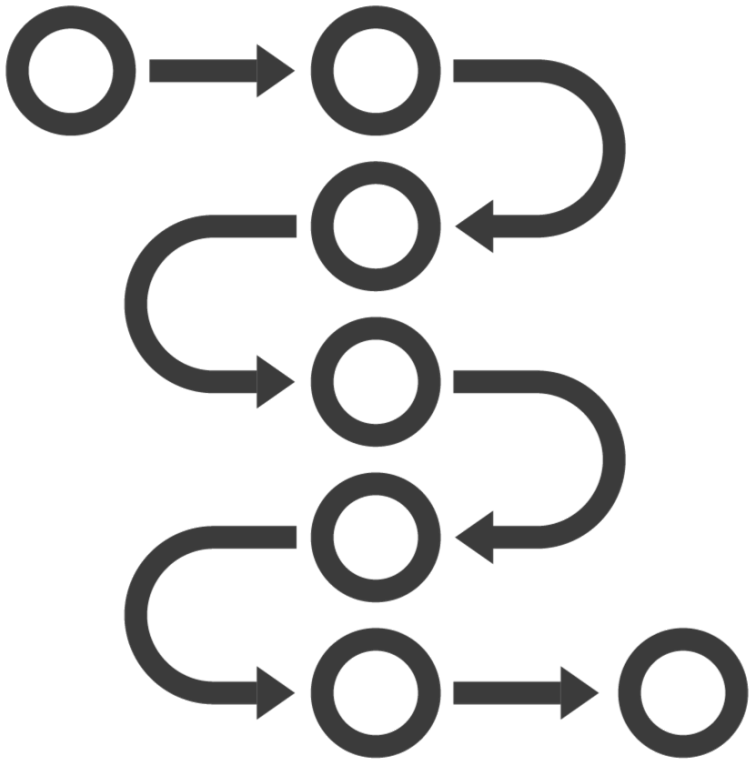
Rational Unified Process

- Relatively long iterations (months)
- Deliver groups of interrelated features

Scrum

- Short iterations (weeks)
- Deliver fewer things per iteration
- Assigned roles and task owners

Iterative Models

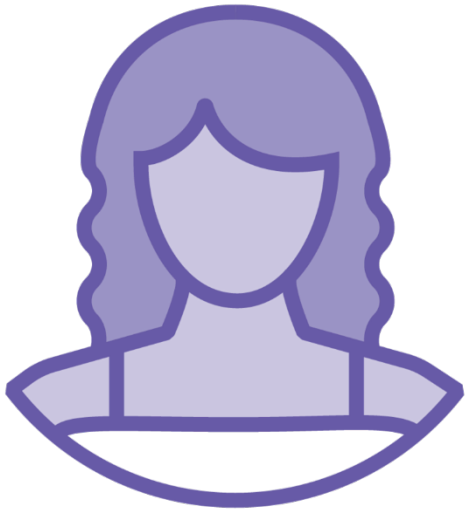


Kanban

- Fluid roles
- Tasks shared by everyone
- Timelines evolve

Spiral

- Experimental
- Most flexible, may integrate other models

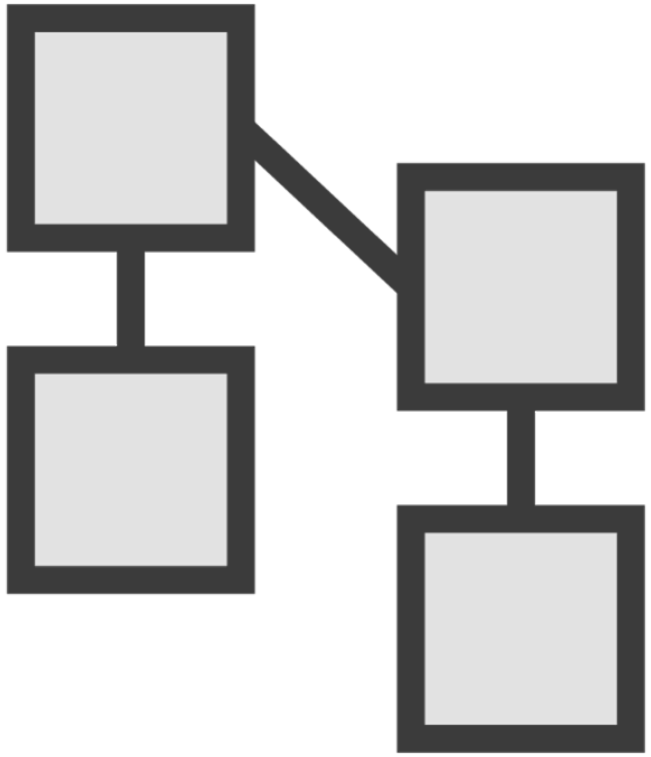


Which SDLC is the best?

Which tool is the best?

Which programming language is the best?

It depends.

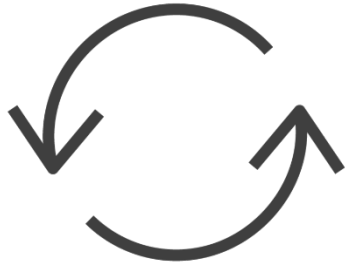


Models are a simplification of reality

A model is like a rough blueprint

Iterative models are NOT always better than sequential

- Context matters
- A robust sequential model may be good enough



Frontend: Scrum

Backend: V-Model

Mixing models is possible
in practice.

Summary

SDLC: Software Development Lifecycle

Two broad categories:

- Sequential: Waterfall, V-Model
- Iterative: timeboxed periods to deliver subsets of functionality
- Iterative examples: RUP, Scrum, Kanban, Spiral

Discovering Test Levels