W4156

Cementing

Good News I

Almost every single challenge the teams are encountering are solvable with the techniques we have learnt thus far!

(we just need to diagnose the problem, find the technique & apply)

Good News II

90% of the ‘pain’ you are experiencing is building you into better software engineers

(learning is hard because you need to level up all the skills and all the skills are interdependent.
Once you write better designs ↠ you introduce less bugs & easier to test & easier to find them when they occur)

Good News III

You have absorbed a lot so we will cement today before moving on + more worked examples by request!

Agenda

- ❏ Cementing with a worked example
- ❏ Emphasize Key Outcomes
- ❏ SOLID OO Design Principles

Big Worked Example

1. Course recommendation system for students @ Columbia
2. Chat bot to assist students and lecturers with common FAQs
3. A system to book equipment at the Columbia Gym
4. Automated class attendance tracker
5. Class choice!

(I do want to pick something in *your* problem domain so we spend more time working on the problem than explaining a novel problem domain)

Big Worked Example

1. Lightening PRD
2. Lightening Roadmap
3. Sprint Backlog
4. OOAD

Key Reminders (Roadmap)

1. Roadmap is high level ordering of features & priorities driven by customer
2. Roadmap != sprints/delivery dates or solutions
3. Roadmap is driven by customer value (enable a persona)
4. Debugging infrastructure is debugging!!! Follow the methodical process!!
5. Make small, incremental changes!!!
6. If you have 4 people, how many tasks can you have in progress?

Product Roadmaps

	Demo	MVP	MMP	Grand Vision
Summary (Human language description of what this milestone deliers)				
Theme 1 (Themes are specific to the app but are broad classifications of maturity)				
Theme 2				
Theme 3				

We are still talking features and ordering! We are not talking solutions or dates[1]

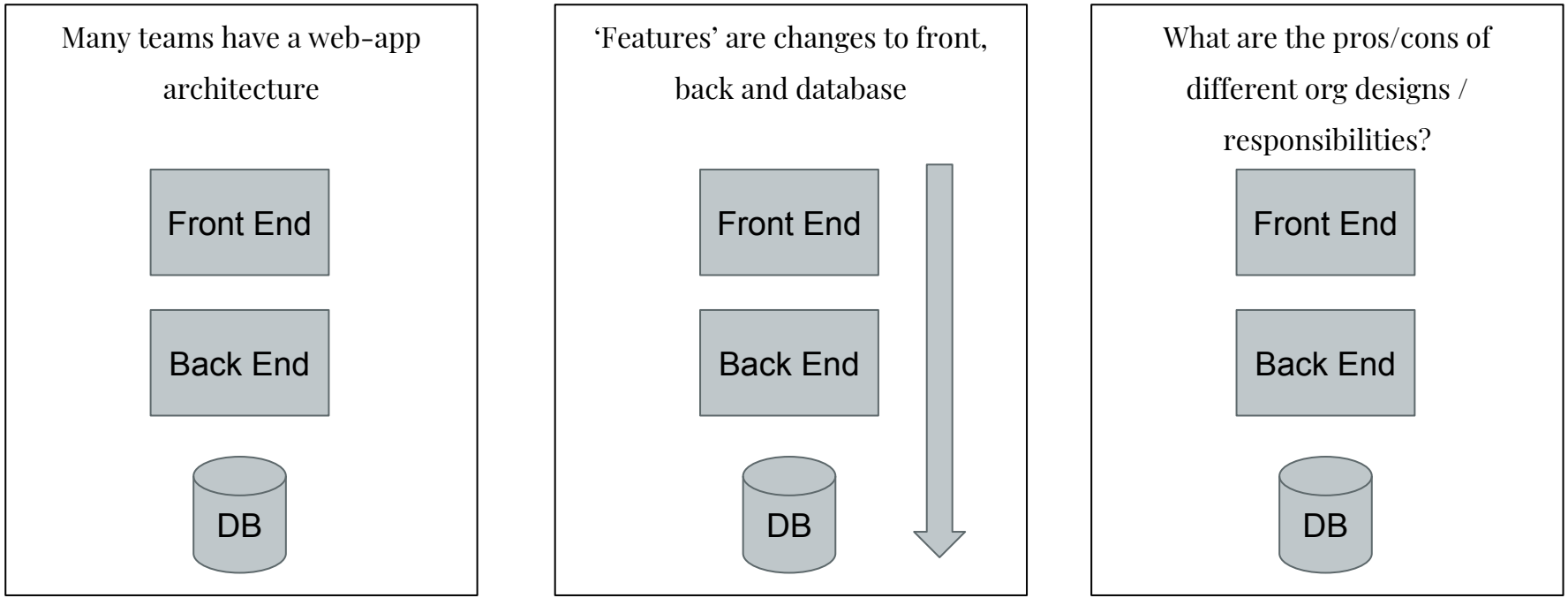
[1] For those who have awareness of agile/scrum I want to emphasize that milestone releases != sprints
(Many roadmap templates and methodologies exist. Given limited time & context I have de-emphasized commercial aspects of roadmap planning)

Key Reminders (Debugging)

1. Debugging infrastructure is debugging!!! Follow the methodical process!!
2. Make small, incremental changes!!!

Thoughts (not covered yet)

1. The ‘flow’ of value is ideas/features to working code

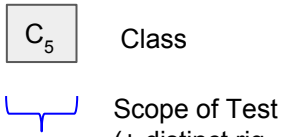


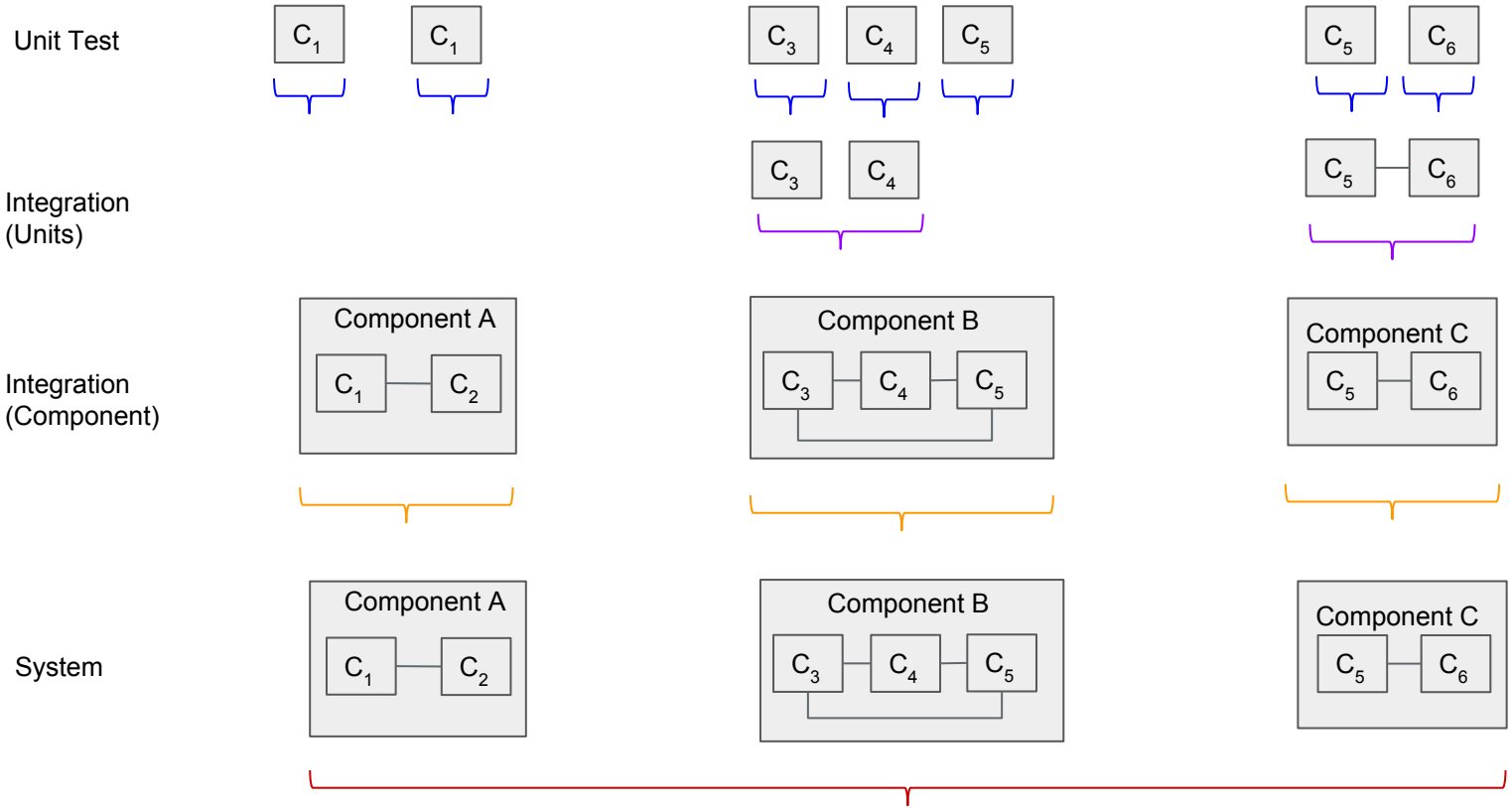
2. If you have 4 people, how many tasks can you have in progress?

Key Reminders (OOAD)

1. OOAD process is a near magical turn-handle which creates your design!
2. Everything is an object (the one I see missed the most is Command)
3. There is an association between unit testing and OOAD
    a. When you decompose you can test individual units

Testing in Industry (Simplified)





Key Reminders (Sprint ➛ Sprint)

1. When you end a sprint the whole thing begins again
2. Feel free to *demo* to classmates
3. Refresh the roadmap incorporating feedback
4. Update the backlog, reload the Trello board
5. You are optimizing for *value*

SOLID

Single Responsibility Principle	Every object should do one thing and one thing only
Open Closed Principle	Objects should be *open* to extension but closed to modification
Liskov Substitution Principle	Where class A calls class B you should be able to *substitute* B with any *subtype* of B while maintaining ‘correctness’ of the behavior of the program
Interface Segregation Principle	A client should only implement interfaces which functionally make sense (should not have ‘empty’ or ‘throw Exception’ interfaces of method). Interfaces should be decomposed to the level where they can faithfully and logically be implemented.
Dependency Inversion Principles	Abstractions should not depend on concrete details && Details should depend on abstractions

Reading

Reading	Optionality
SOLID	Recommended