

What To Test, And When

Udi Dahan – The Software Simplist
.NET Development Expert & SOA Specialist
Microsoft Solutions Architect MVP



What are we paid to do?



Create business value

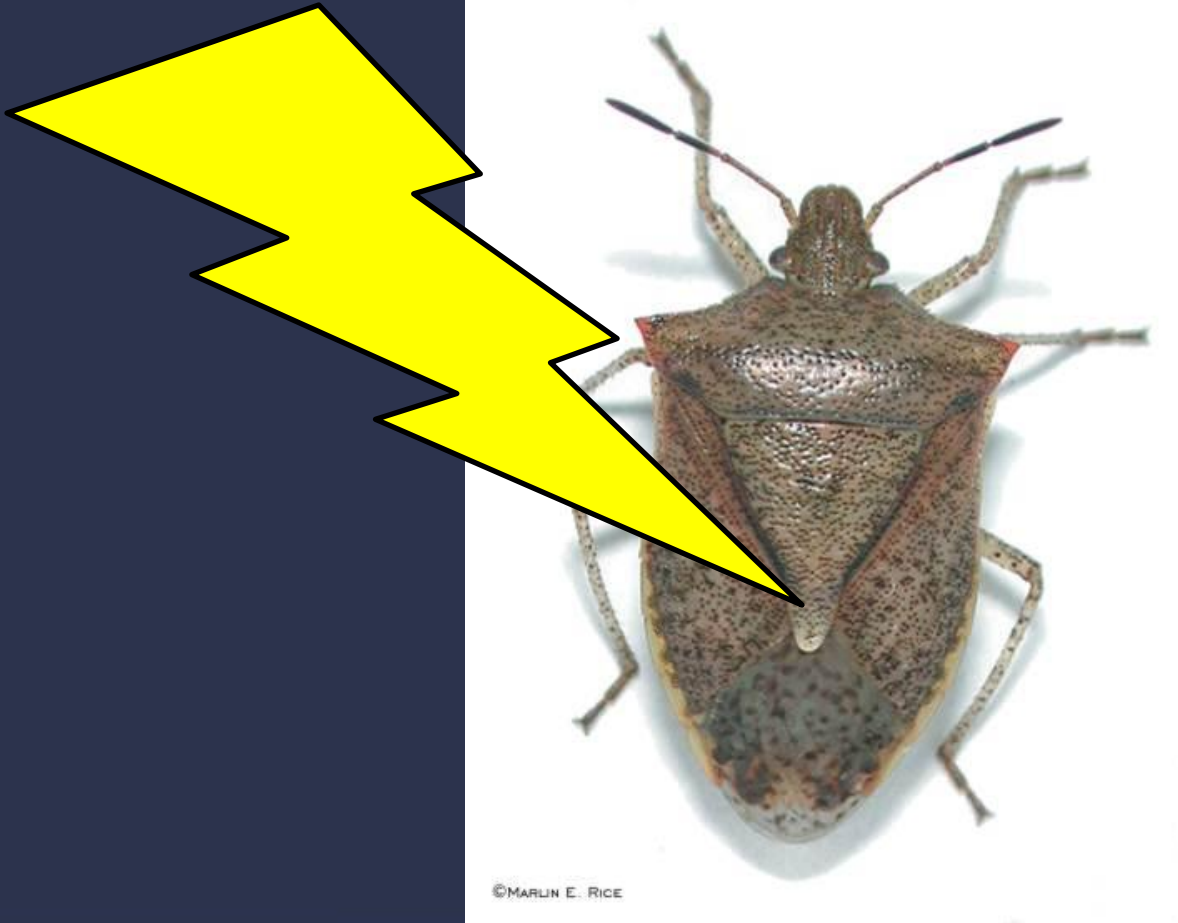
Does finding bugs count as creating value?



©MARLIN E. RICE

NO.

Does **fixing** bugs count as creating value?



Yes.

Does preventing bugs count as creating value?



Absolutely.

What kinds of tests are there?

Automate

Business Facing

Support Programming

Acceptance Tests	Usability Testing
Business Intent Design of the Product	Exploratory Testing
Unit Tests	Property Tests
Developer Intent Design of Code	Response Times, Security, Scaling

Critique Product

Automate

Technology Facing

Tests tell us where we are



But what should I test, and when?



Risk endangers business value



Handling schedule risk



Handling schedule risk

Use short iterations



“Iterations” means doing some (or all) of these

Acceptance	Usability Exploratory
Unit	Property

every iteration

Nothing happens outside an iteration

Testing is a lifecycle issue

RUP

Disciplines

Business Modeling

Requirements

Analysis & Design

Implementation

Test

Deployment

Configuration
& Change Mgmt

Project Management

Environment

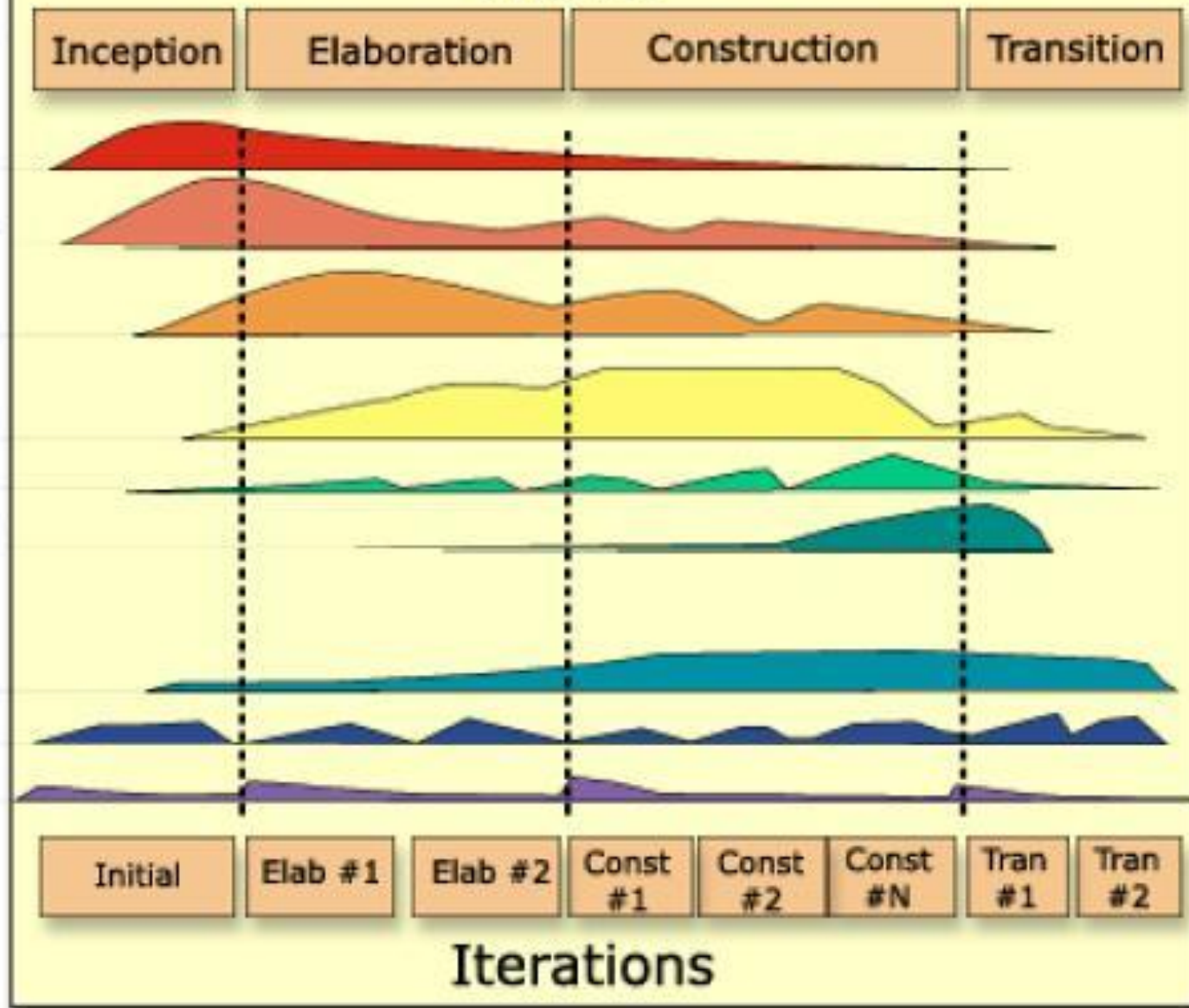
Phases

Inception

Elaboration

Construction

Transition



Iterations

Initial

Elab #1

Elab #2

Const #1

Const #2

Const #N

Tran #1

Tran #2



How the customer explained it



How the Project Leader understood it



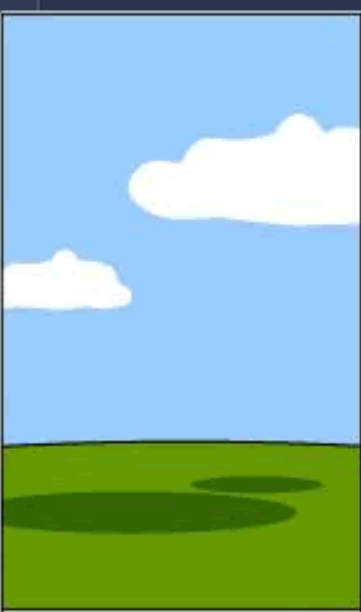
How the Analyst designed it



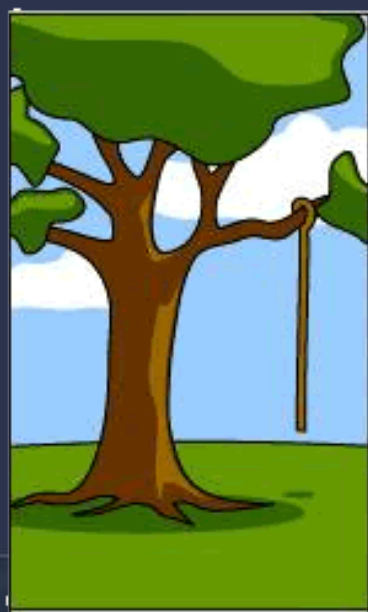
How the Programmer wrote it



How the Business Consultant described it



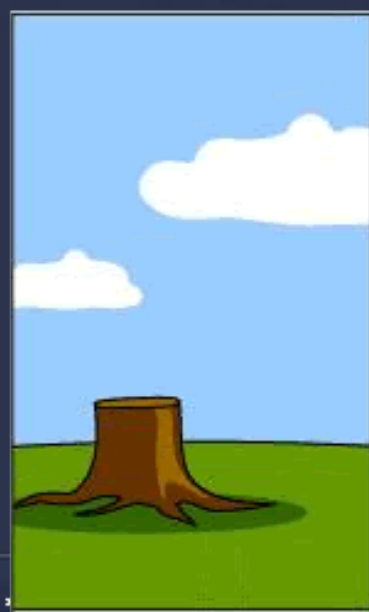
How the project was documented



What operations installed



How the customer was billed



How it was supported



What the customer really needed

So what should I test, and when?



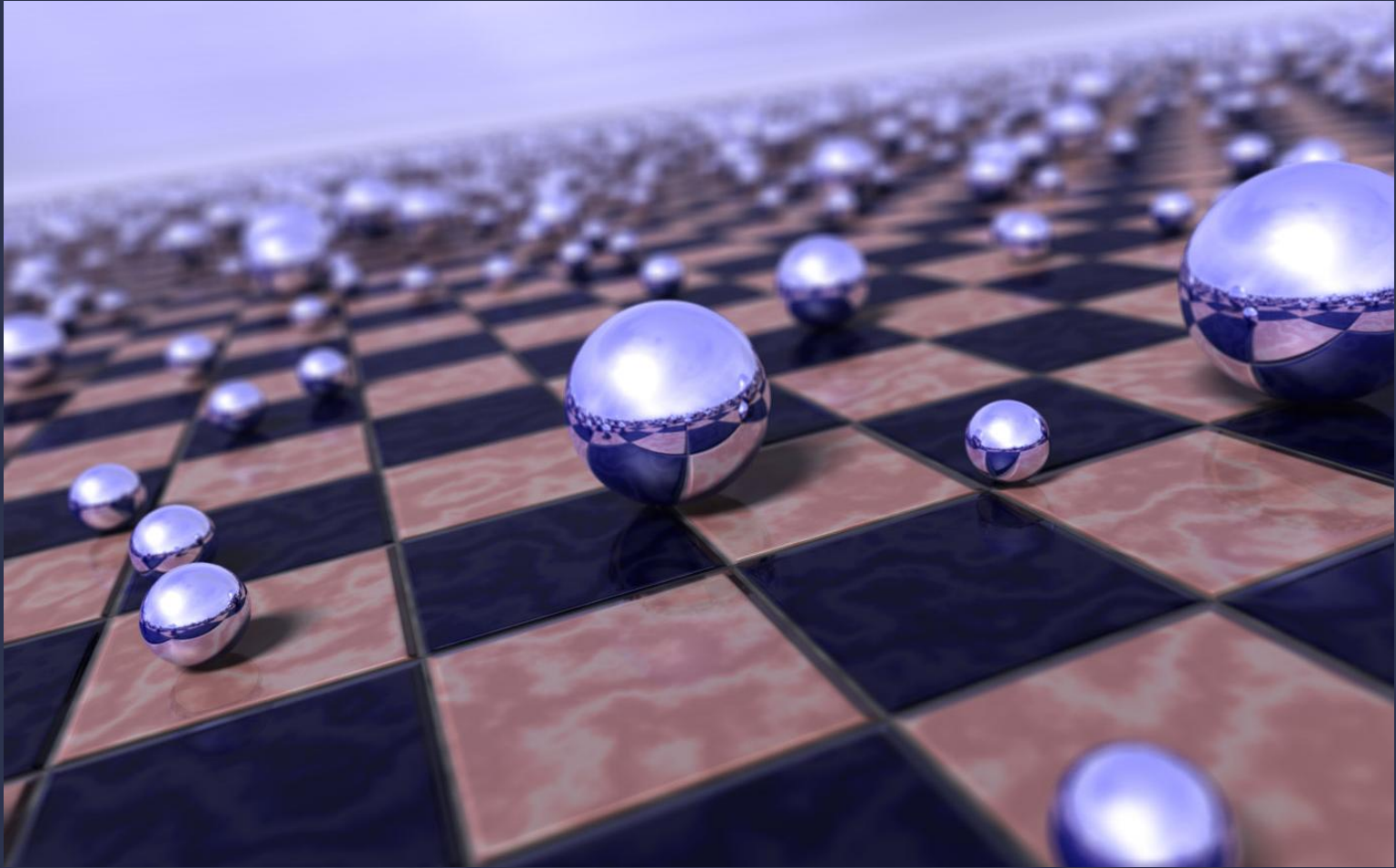
Test what's risky

- Performance?
- Security?
- Complex business logic?
- External interfaces?

Feature test all the time

- Unit tests while writing code
- Acceptance tests as soon as part of the feature works

We're going to have a lot of tests



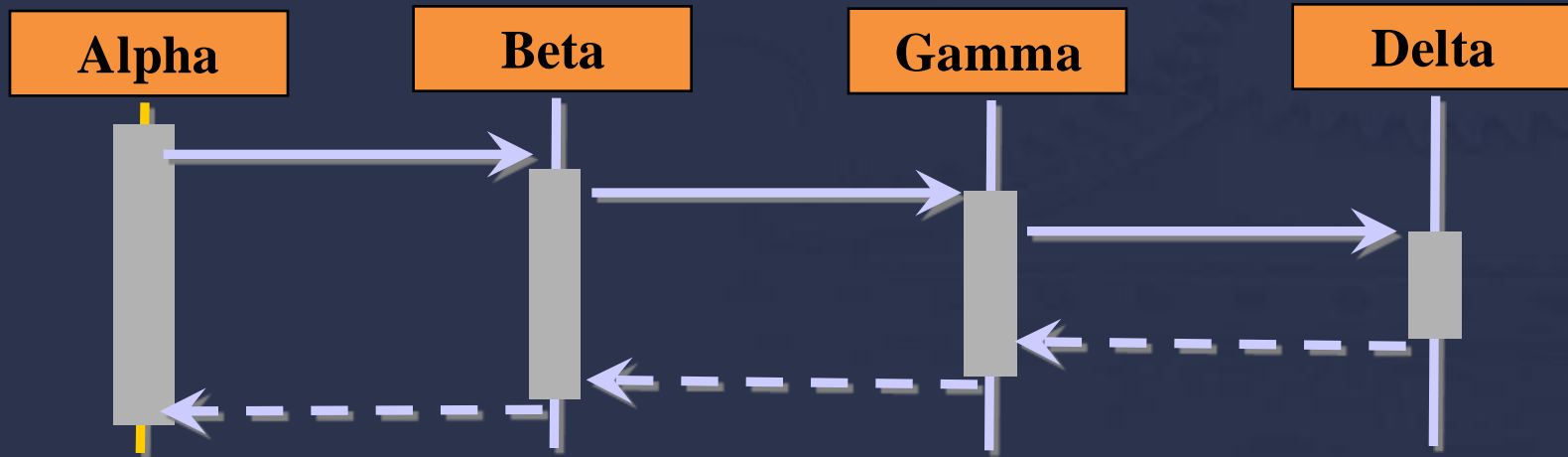
Decide what to run when

Regressions – everyone's nightmare

- January 1st: Bug A found
- January 15th: Bug A fixed
- February 1st: Bug B found
- February 15th: Bug B fixed
- March 1st: Bug A found again

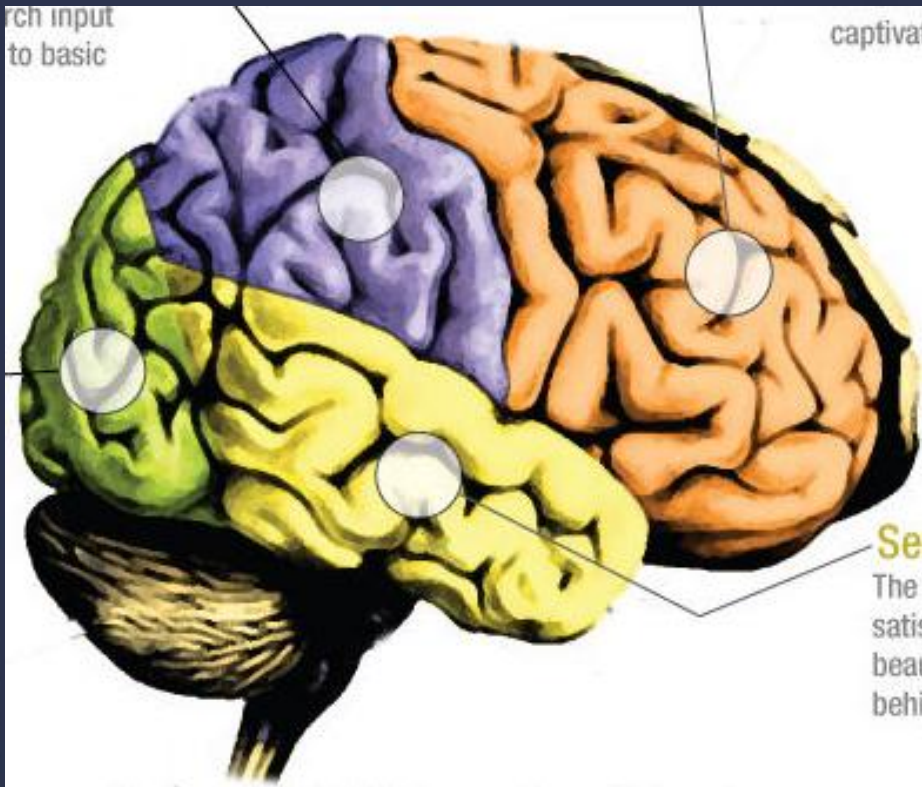
Why don't bugs stay fixed?

Regressions – why they happen



- In order to fix bug A, you change Gamma
- In order to fix bug B, you **What if it's someone else, and not you?**
- Changing Delta might cause Gamma to break

Regressions - An ounce of prevention...



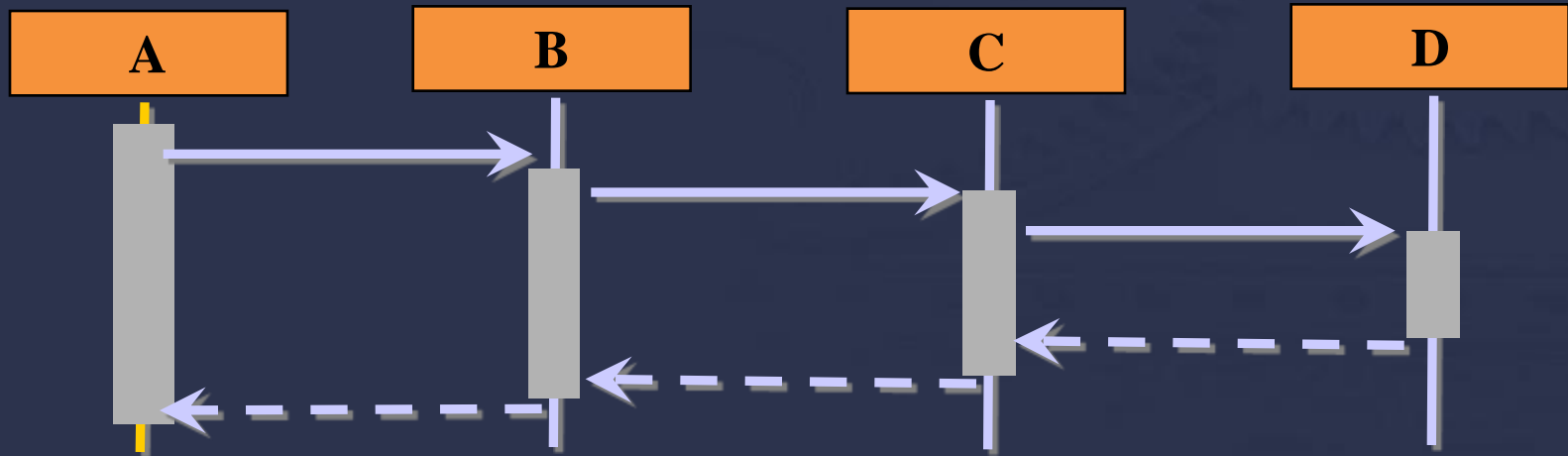
Wouldn't it be great if there was something intelligent that could help?

That's what automated tests do

But no object is an island

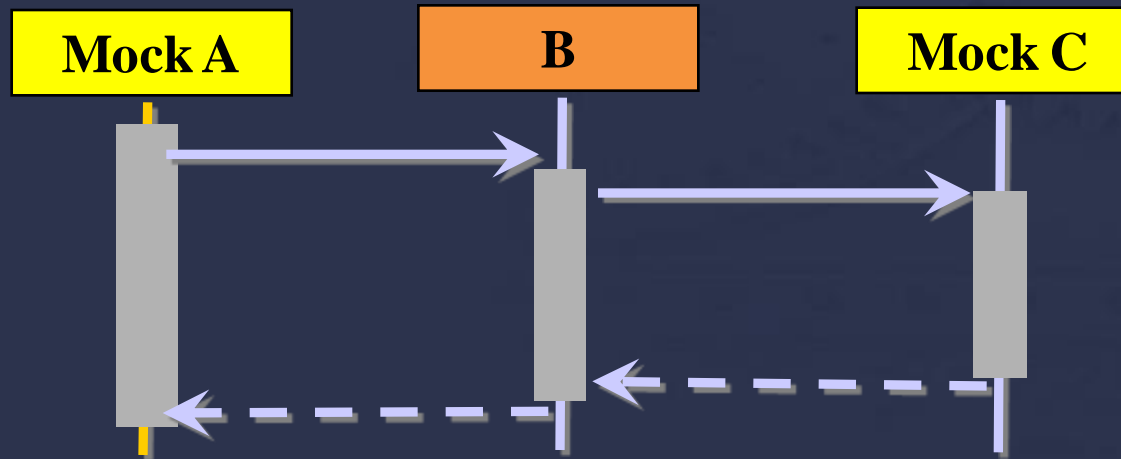


Is there even a “unit” ?



Maybe we should focus on interactions

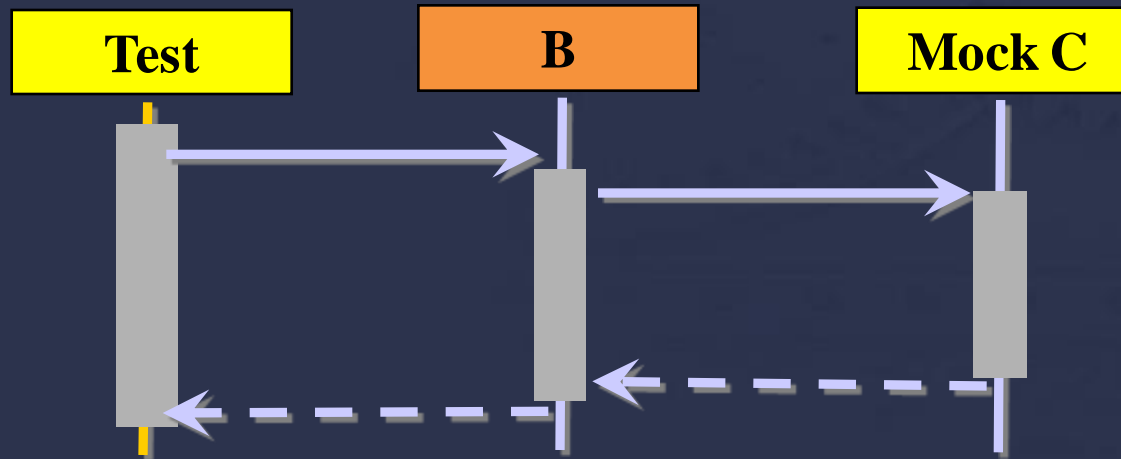
Testing with Mock Objects



Looks like a unit test

But validates state and interactions

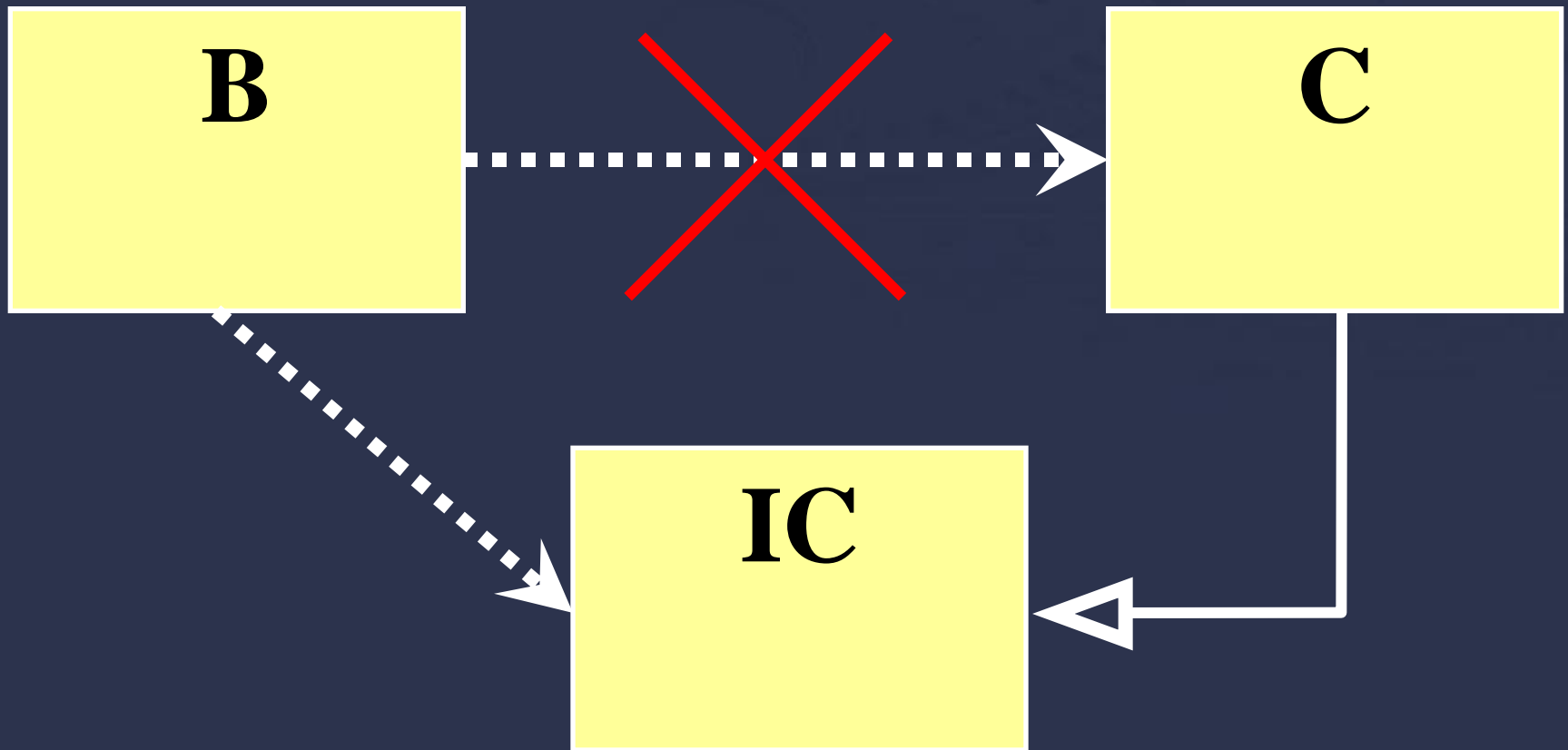
But if we want to test “B”



We can't have it depend on C

We need to replace C

Changing the design might help



Design for testability



Coverage:
how much?

Load

A strong donkey is no replacement for a heavy ox.



Thank you

**Udi Dahan – The Software Simplist
.Net Development Expert & SOA Specialist
Microsoft Solutions Architect MVP**

