Component Testing

@zmerta

WHAT IS A COMPONENT?

"A component is any well-encapsulated, coherent
and independently replaceable part of a larger system."

–Toby Clemson

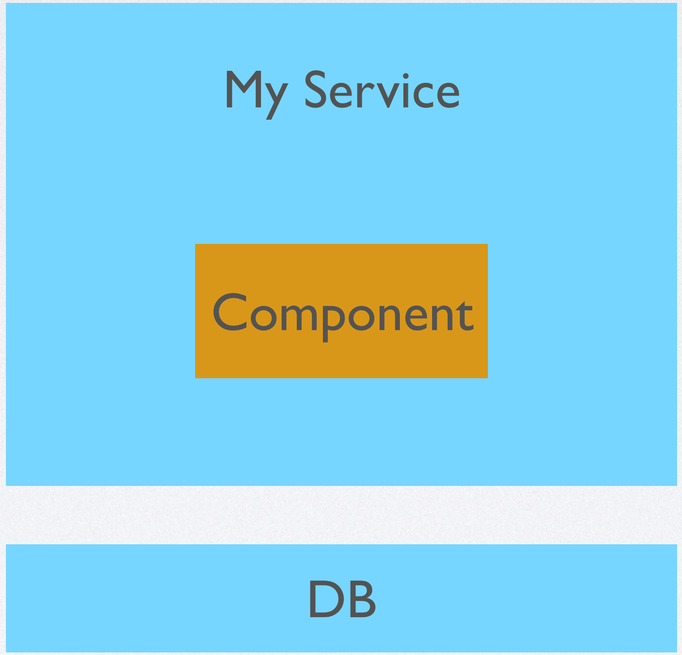
Component testing
in Microservices Architecture demands
that each component is tested in isolation
by replacing external dependencies.

SITUATIONS

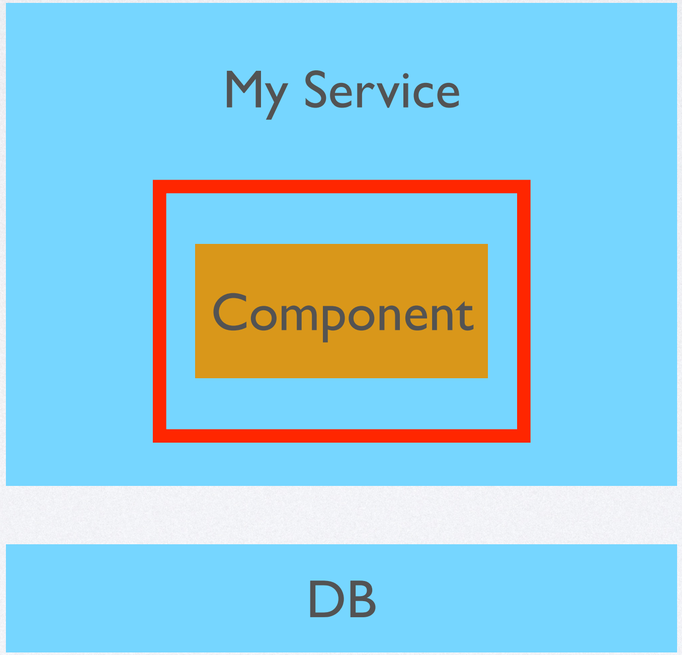
THREE BASIC SITUATIONS

- A component inside a service
- A component inside a service with an external dependency
- A resource component inside a service

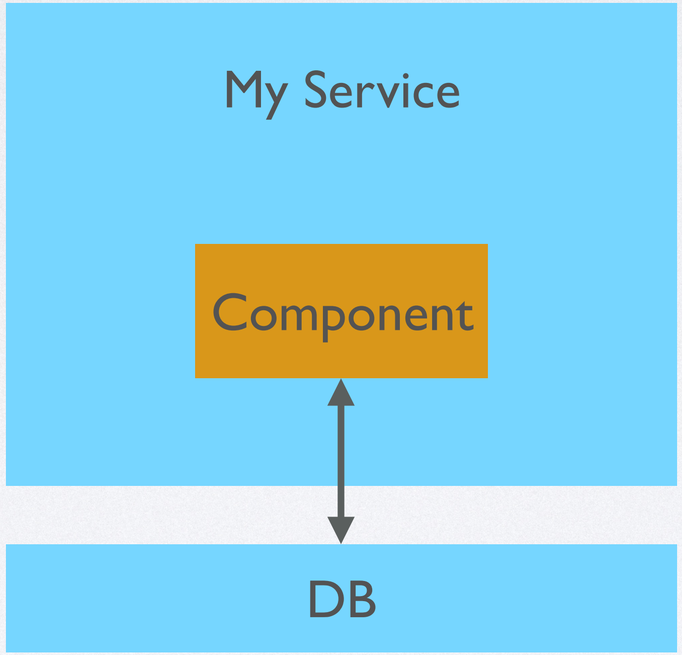
A COMPONENT INSIDE A SERVICE



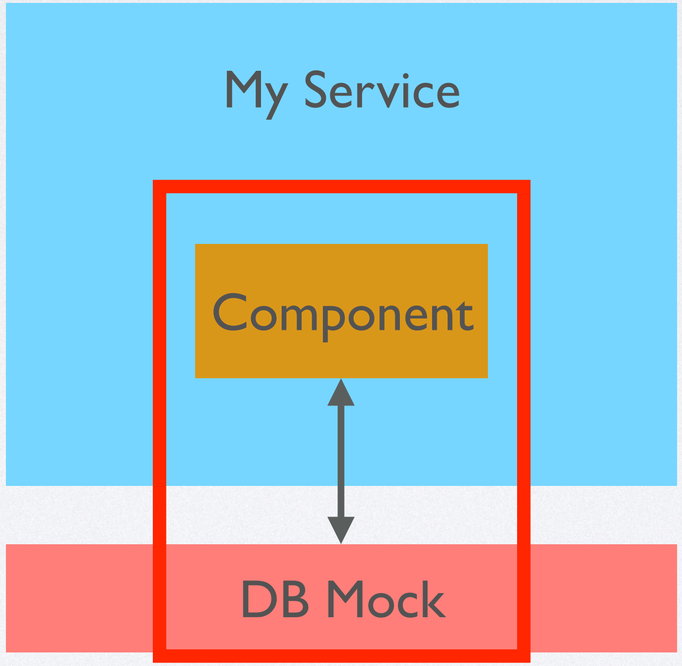
A COMPONENT INSIDE A SERVICE



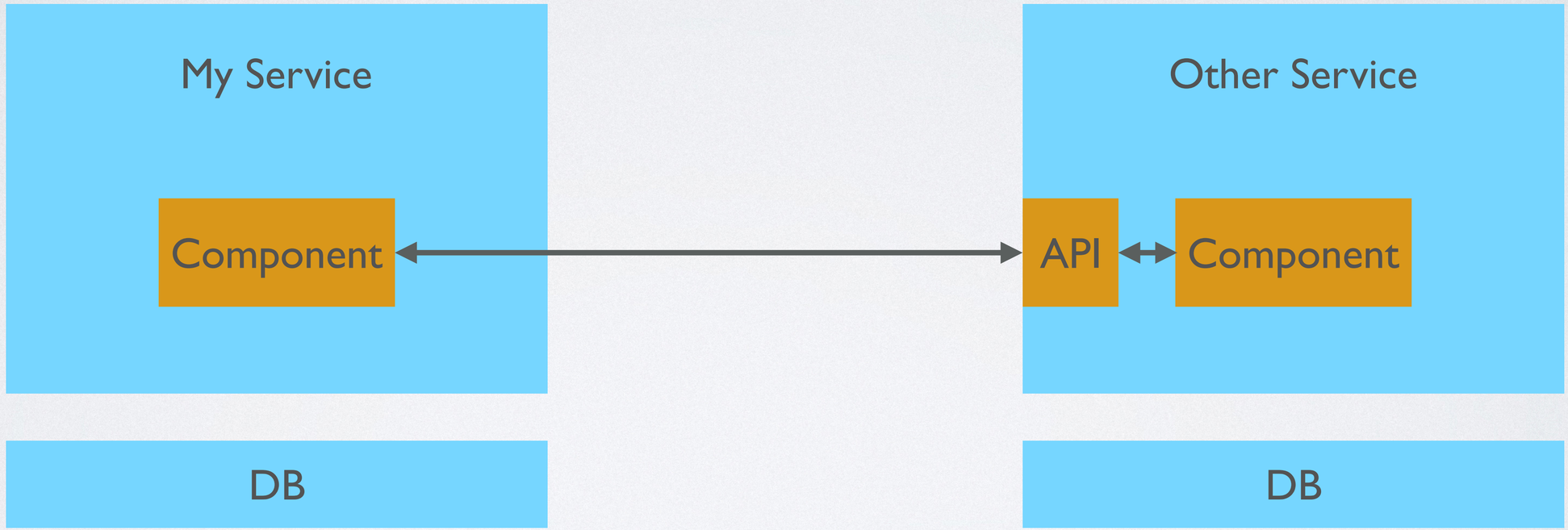
A COMPONENT INSIDE A SERVICE WITH AN EXTERNAL DEPENDENCY I



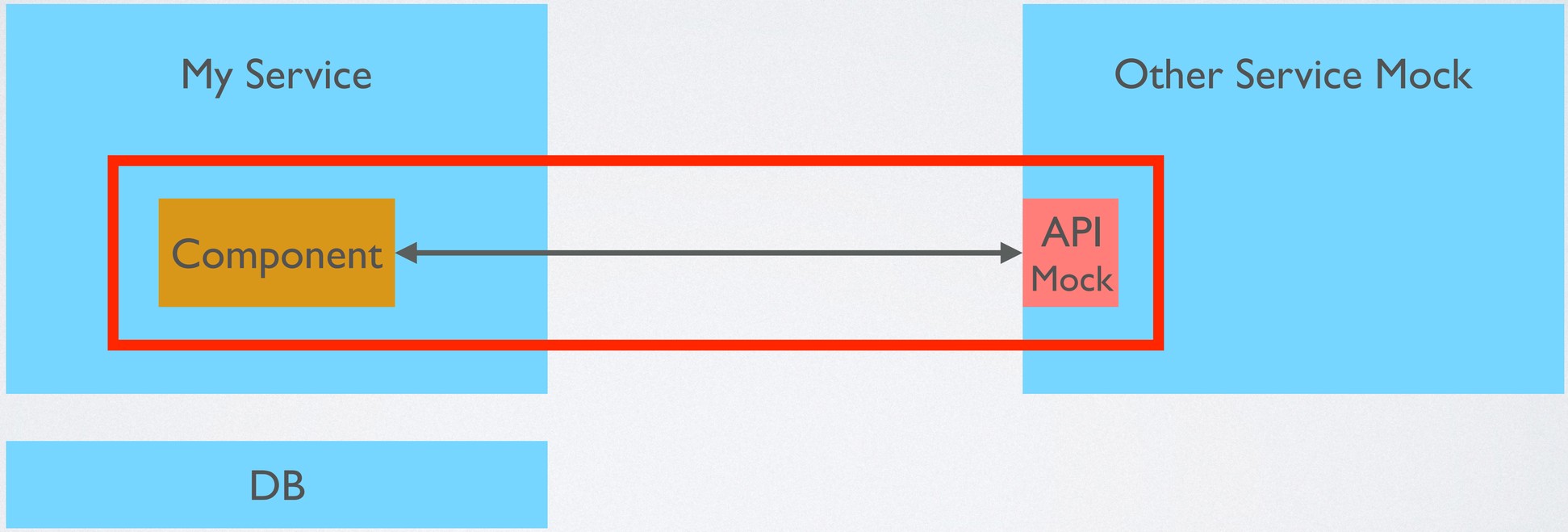
A COMPONENT INSIDE A SERVICE WITH AN EXTERNAL DEPENDENCY I



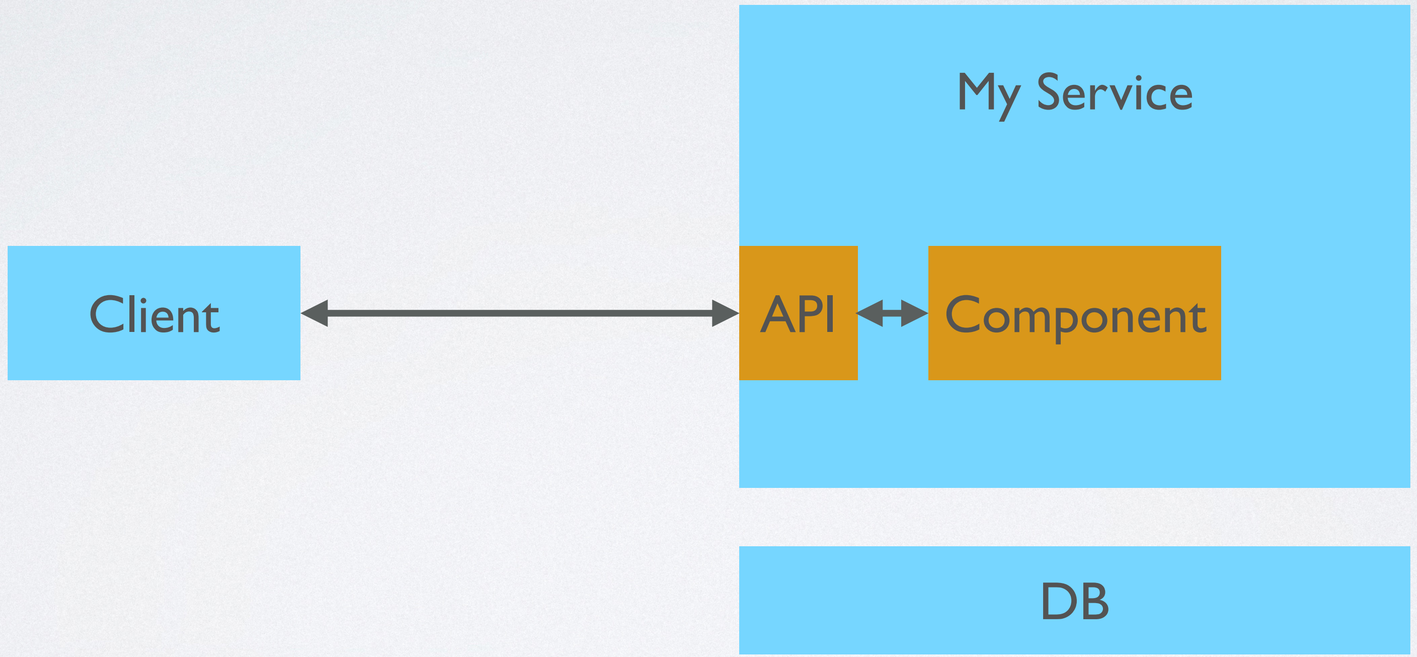
A COMPONENT INSIDE A SERVICE WITH AN EXTERNAL DEPENDENCY 2



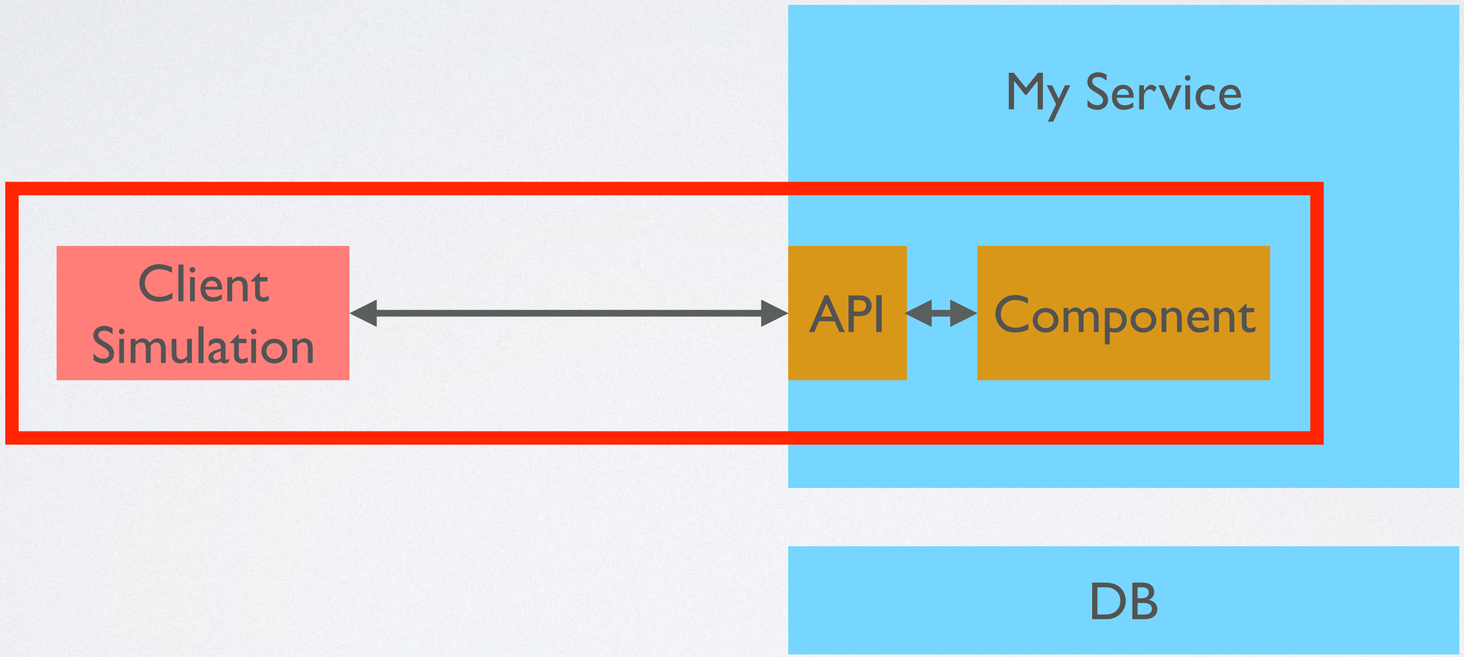
A COMPONENT INSIDE A SERVICE WITH AN EXTERNAL DEPENDENCY 2



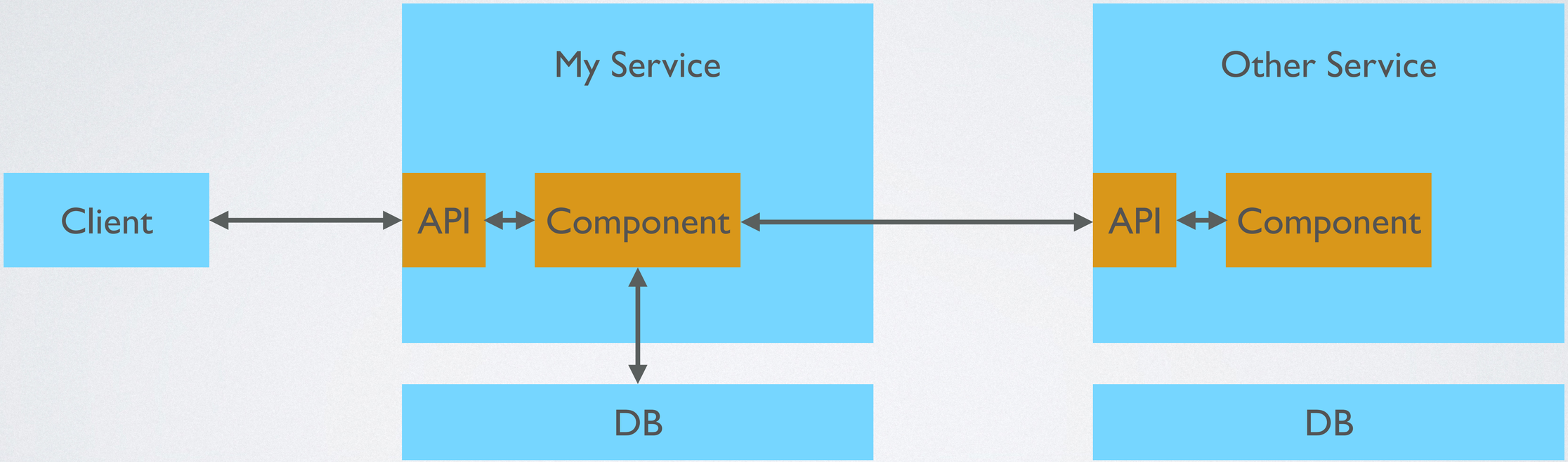
A RESOURCE COMPONENT INSIDE A SERVICE



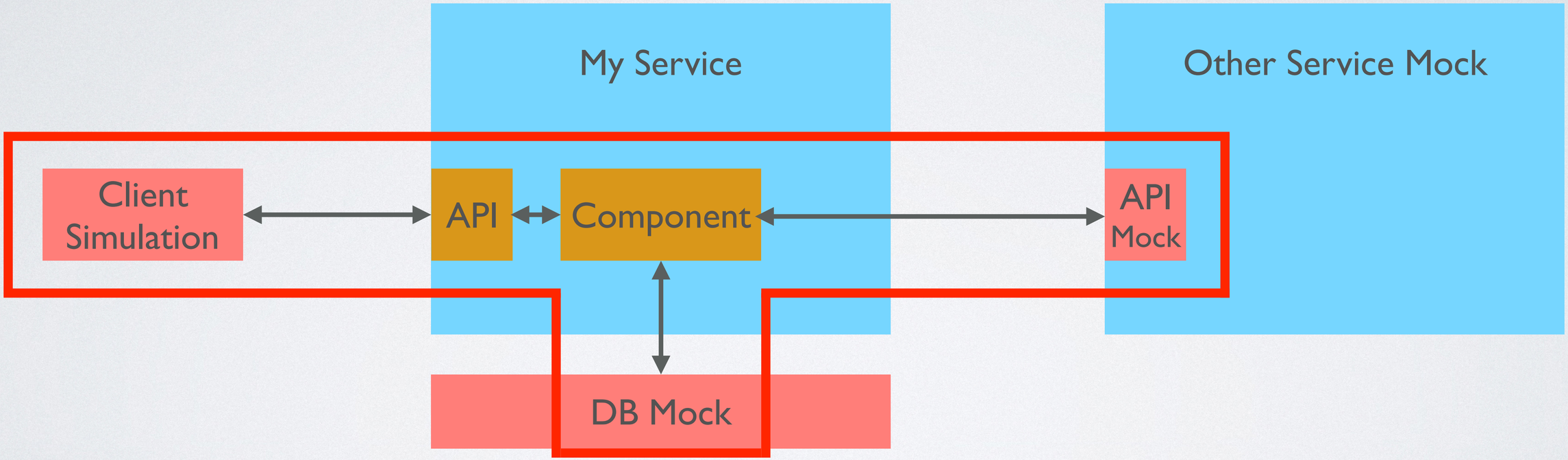
A RESOURCE COMPONENT INSIDE A SERVICE



A COMBINATION



A COMBINATION



TESTING TOOLS

TESTING FRAMEWORKS

Classical Frameworks

- jUnit
- Spock

BDD/ATDD/SBE

- cucumber
- JBehave

Which testing framework should I use?

Are you testing an API?

Are you testing an API?

Are you testing a Technical Component?

Are you testing an API?

Are you testing a Technical Component?

Use a classical Testing Framework

Are you testing a Business Logic?

Are you testing a Business Logic?

Do you have a Business Specification?

Are you testing a Business Logic?

Do you have a Business Specification?

Should the tests be readable by Business?

Are you testing a Business Logic?

Do you have a Business Specification?

Should the tests be readable by Business?

Use BDD Testing Framework

MOCKS, STUBS, TEST DOUBLES FRAMEWORKS

- mockito
- PowerMock
- WireMock
- mountebank

END-TO-END TESTING
VS
COMPONENT TESTING

Writing and maintaining end-to-end tests can be very difficult.

END-TO-END TESTING CHALLENGES

- Dynamic environment
- Brittle tests (more reasons to fail)
- Additional cost of maintenance
- Who will create and maitain end-to-end tests?

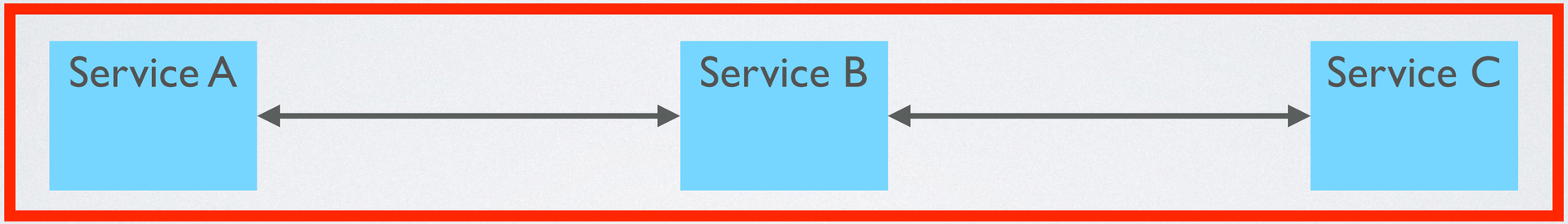
Focus on user journeys.

Divide them into component tests.

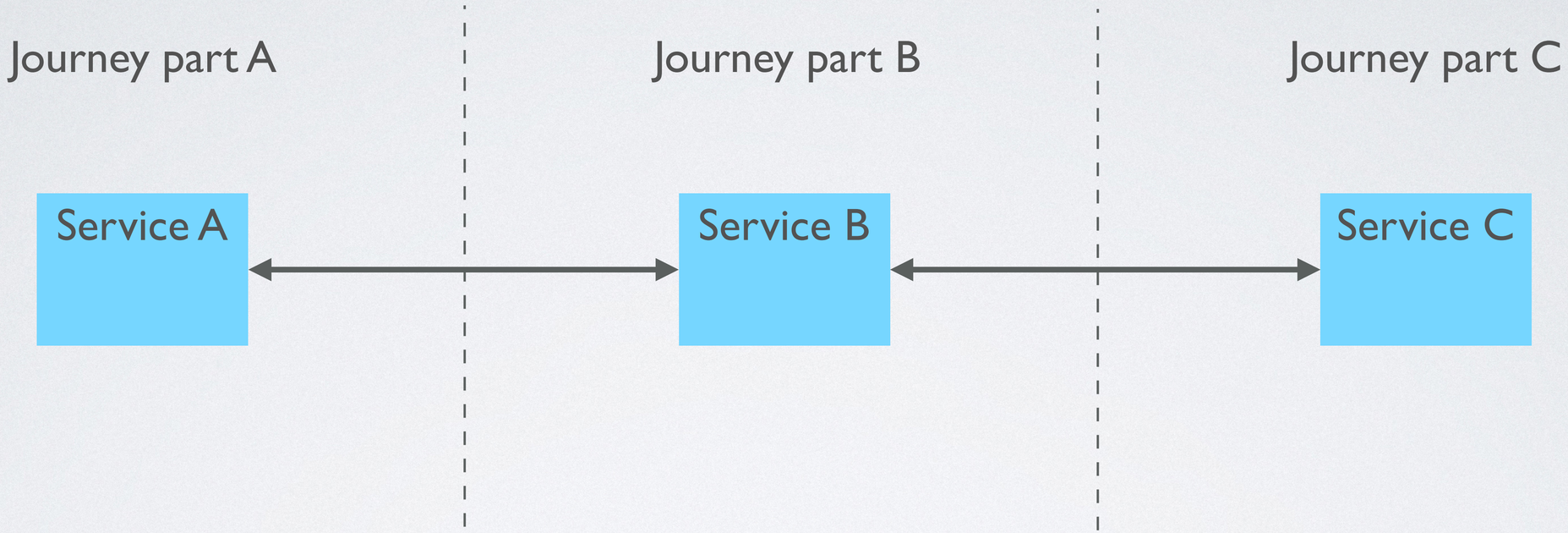
EXAMPLE

TEST LANDSCAPE

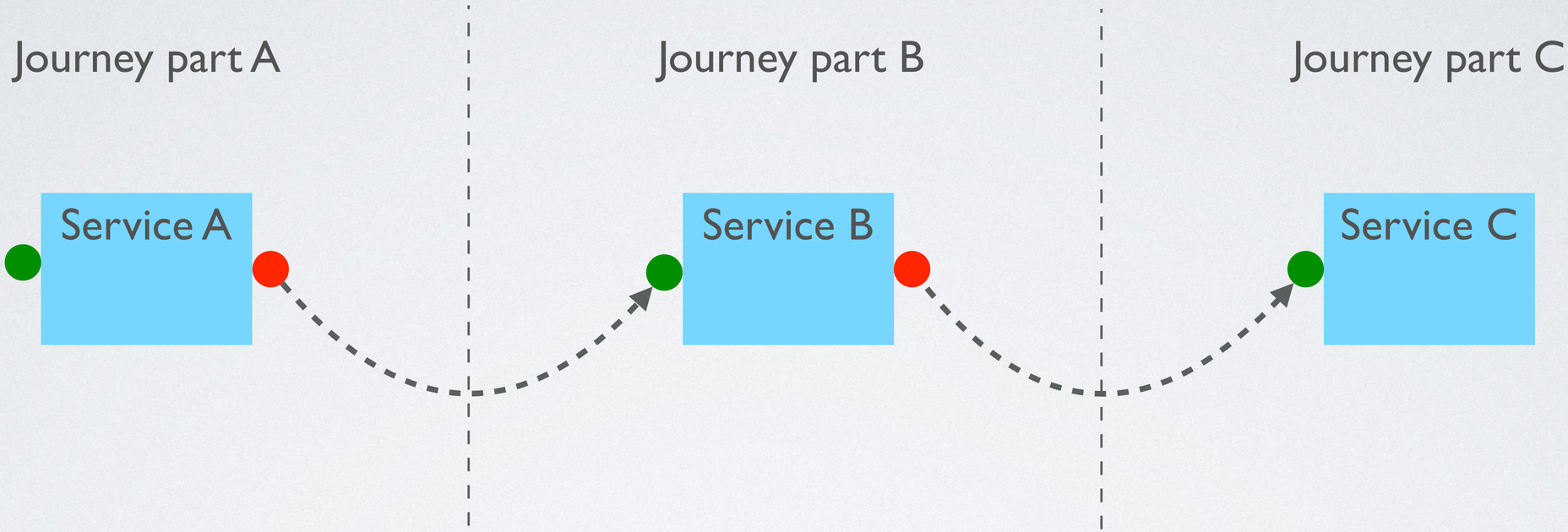
END-TO-END TEST JOURNEY



JOURNEY PARTS

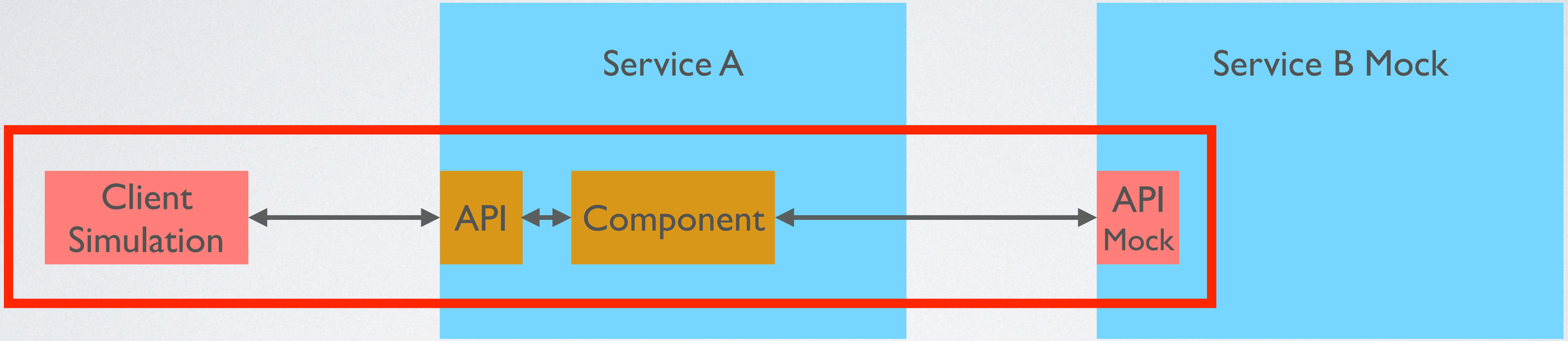


JOURNEY PARTS

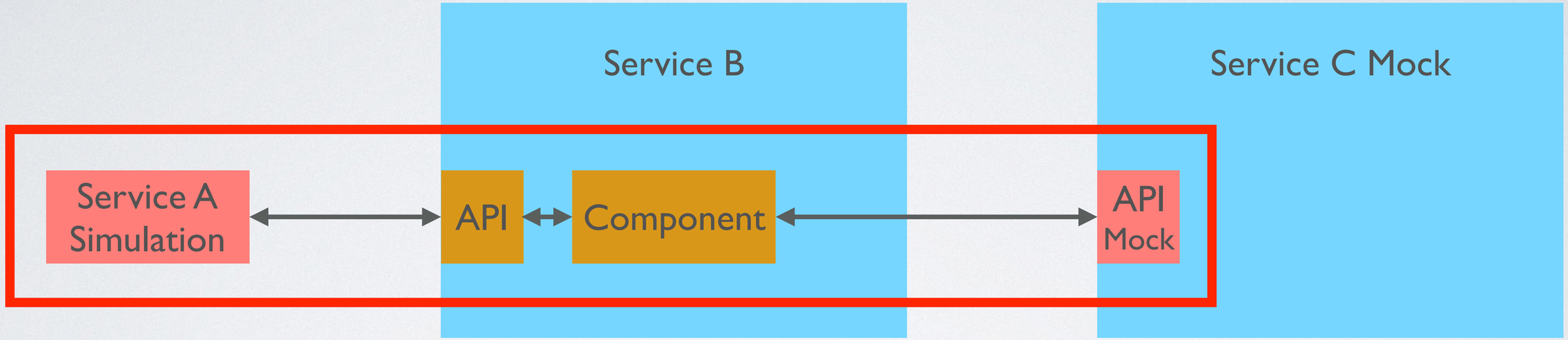


Test in isolation, use output of a journey part as an input for the next journey part

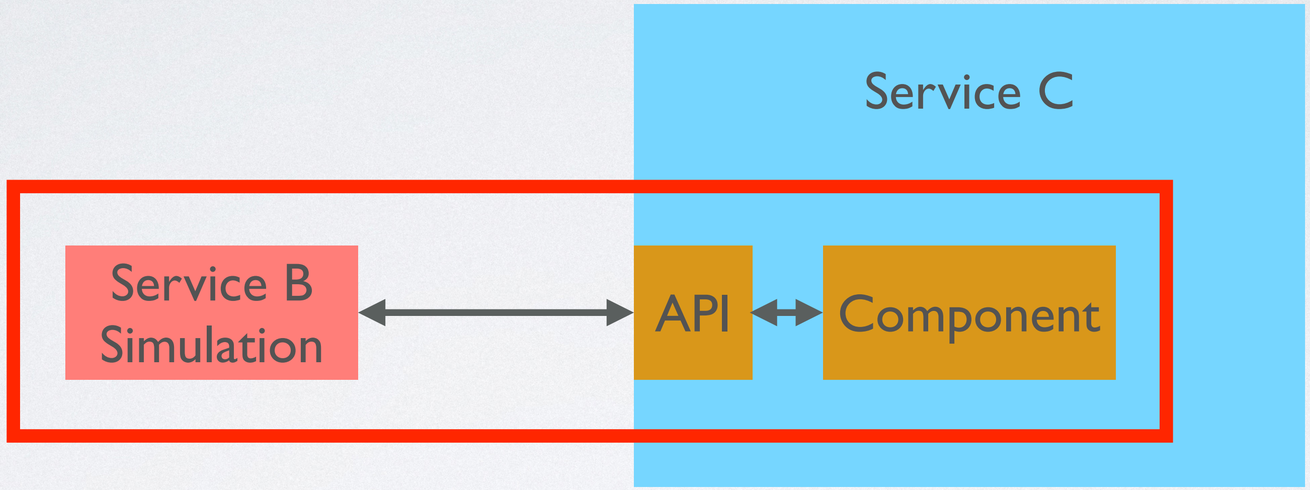
JOURNEY PART A



JOURNEY PART B



JOURNEY PART C



This is the end, beautiful friend

This is the end, my only friend, the end

. . .