

Microservices





Somkiat Puisungnoen

Update Info 1 View Activity Log 10+ ...

Timeline About Friends 3,138 Photos More

When did you work at Opendream? X

... 22 Pending Items

Post Photo/Video Live Video Life Event

What's on your mind?

Public Post

Intro

Software Craftsmanship

Software Practitioner at สยามชั่นนาฎกิจ พ.ศ. 2556

Agile Practitioner and Technical at SPRINT3r

Somkiat Puisungnoen 15 mins · Bangkok · ...

Java and Bigdata



somkiat.cc

Page Messages Notifications 3 Insights Publishing Tools Settings Help ▾

somkiat.cc
@somkiat.cc

Home Posts Videos Photos

Liked Following Share ... + Add a Button



Agenda

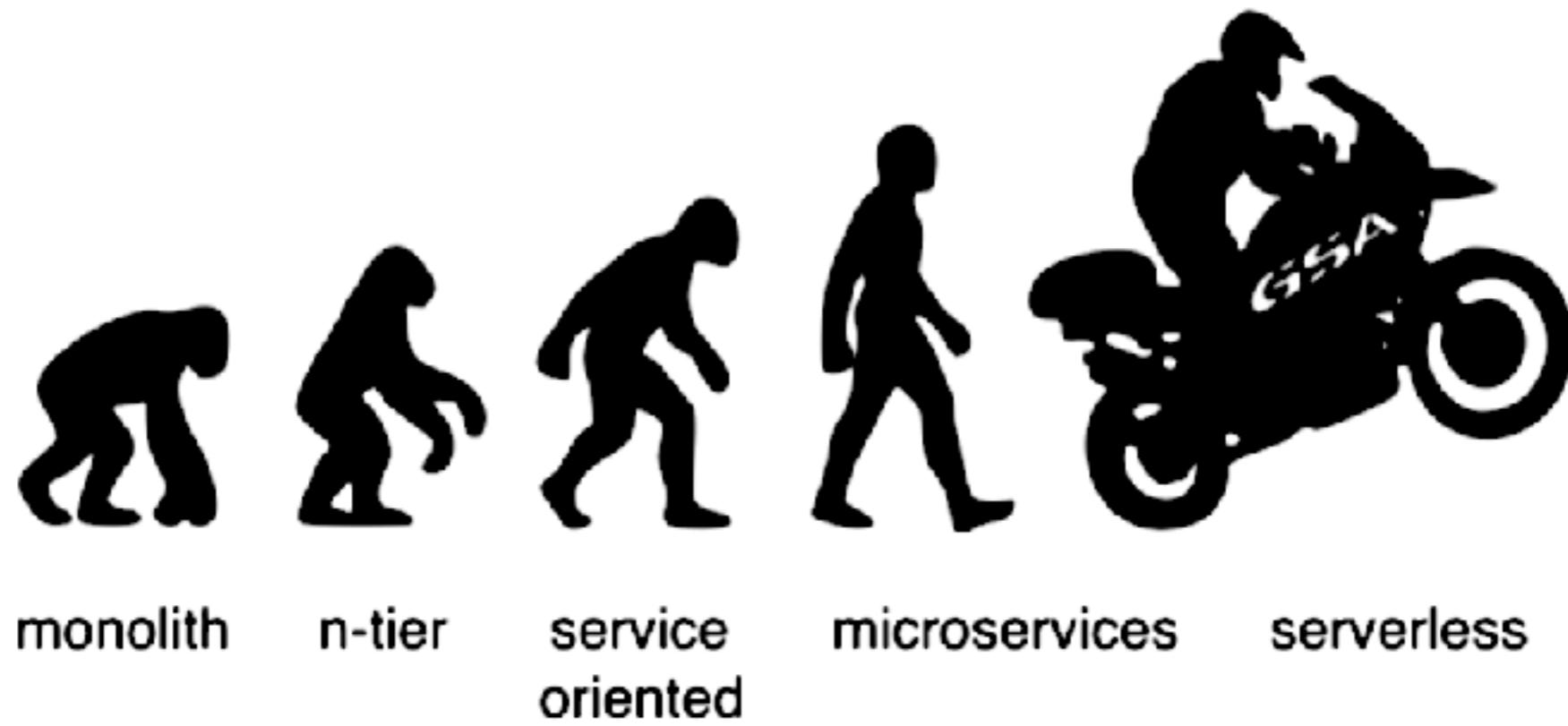
- Cloud Native Application
- Microservices and DevOps
- The architecture of Microservices
- How to model Microservices
- Integrating multiple Microservices



Agenda

- Testing and Developing Microservices
- Monitoring Microservices
- Deploy and Scale Microservices



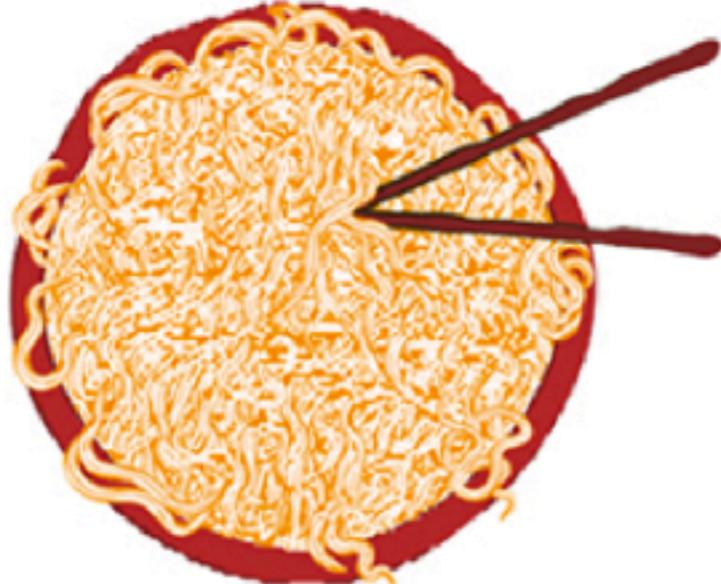


Evolution of Architecture



1990s and earlier

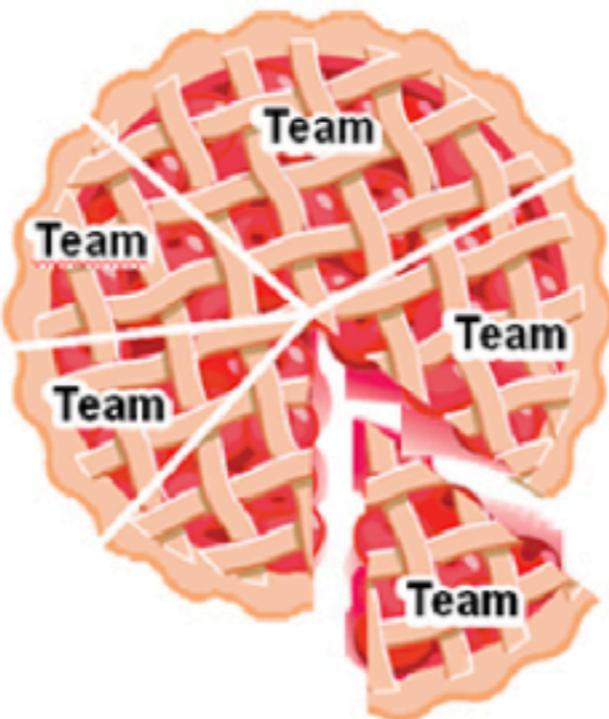
Pre-SOA (monolithic)
Tight coupling



For a monolith to change, all must agree on each change. Each change has unanticipated effects requiring careful testing beforehand.

2000s

Traditional SOA
Looser coupling



Elements in SOA are developed more autonomously but must be coordinated with others to fit into the overall design.

2010s

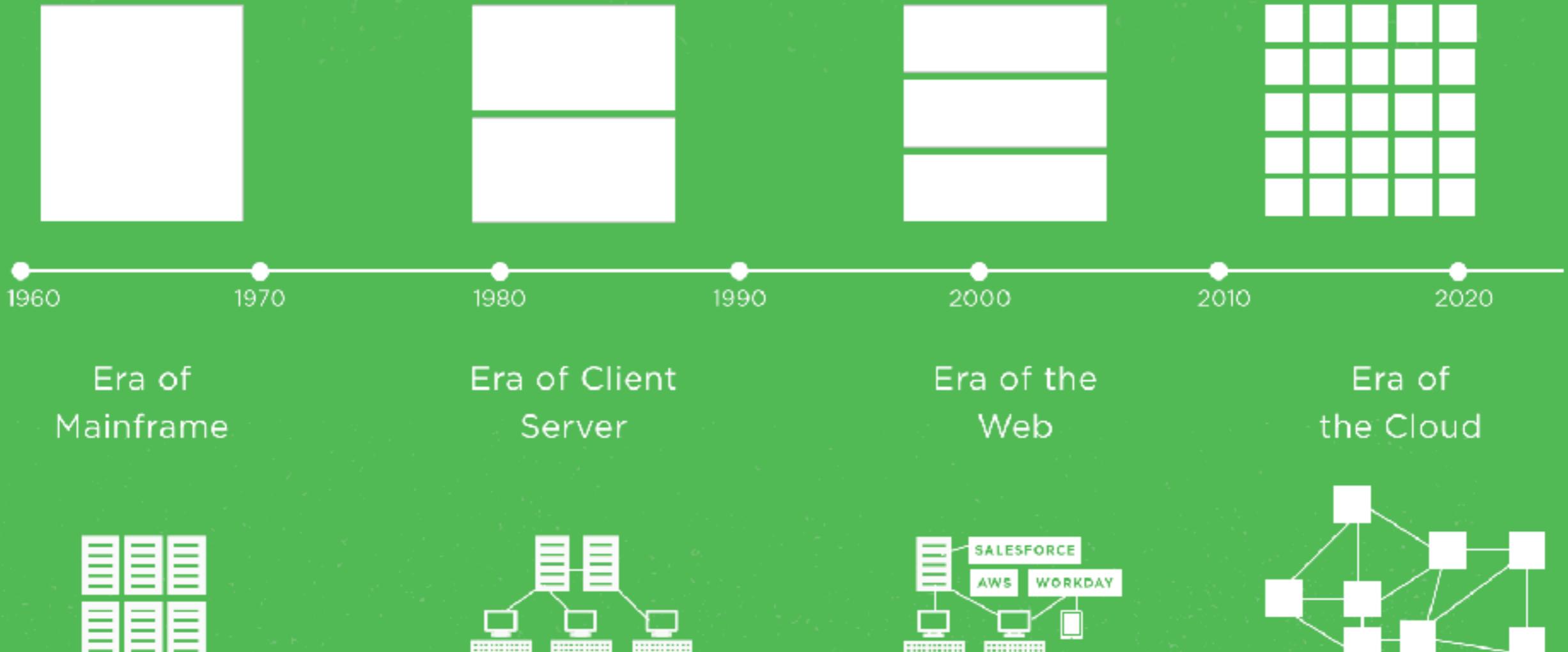
Microservices
Decoupled



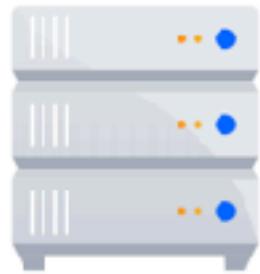
Developers can create and activate new microservices without prior coordination with others. Their adherence to MSA principles makes continuous delivery of new or modified services possible.



ABSTRACTION IN THE ENTERPRISE



Monolith issues



Grow Fat

Code base grows. All the things slow down.



Age

Your code base will become a jurassic park introducing new tech becomes hard



Ownership

Who is responsible for which part and more important: who has the pager



Economies of Scale

The bigger the team the more they interrupt each other

<https://trello.com/>



We need Change





We need Change



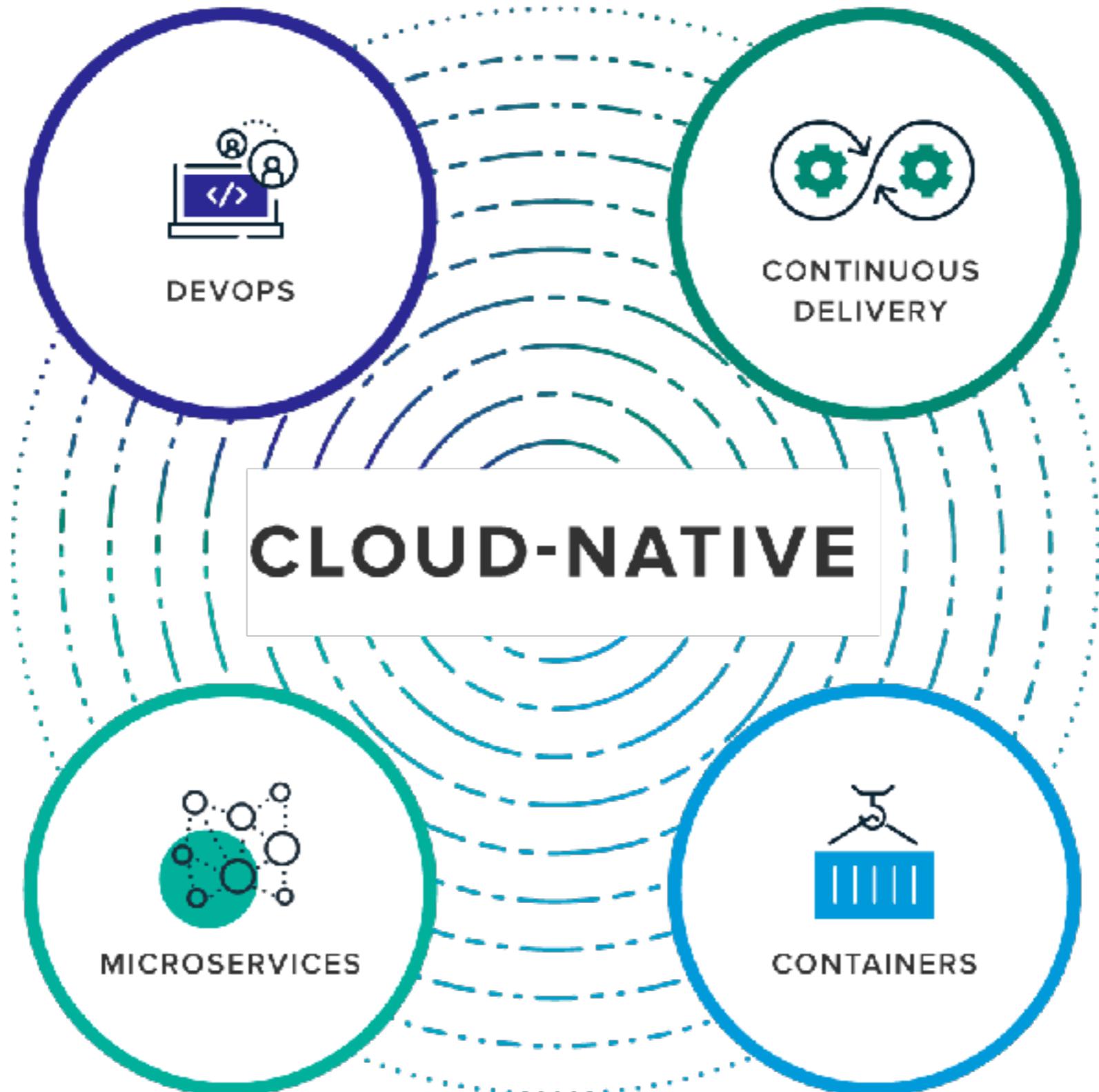
**“Remember, we are not all
webscale”**



“Optimize for rapid and sustainable flow of value”

Dan Nord





<https://pivotal.io/cloud-native>



Microservices



Microservices



Small

The size will be reasonable and manageable



Independent lifecycle

Nothing will hold the team back. Go as fast as you can



Optimise for the problem

Pick solution and tech based on the problem at hand



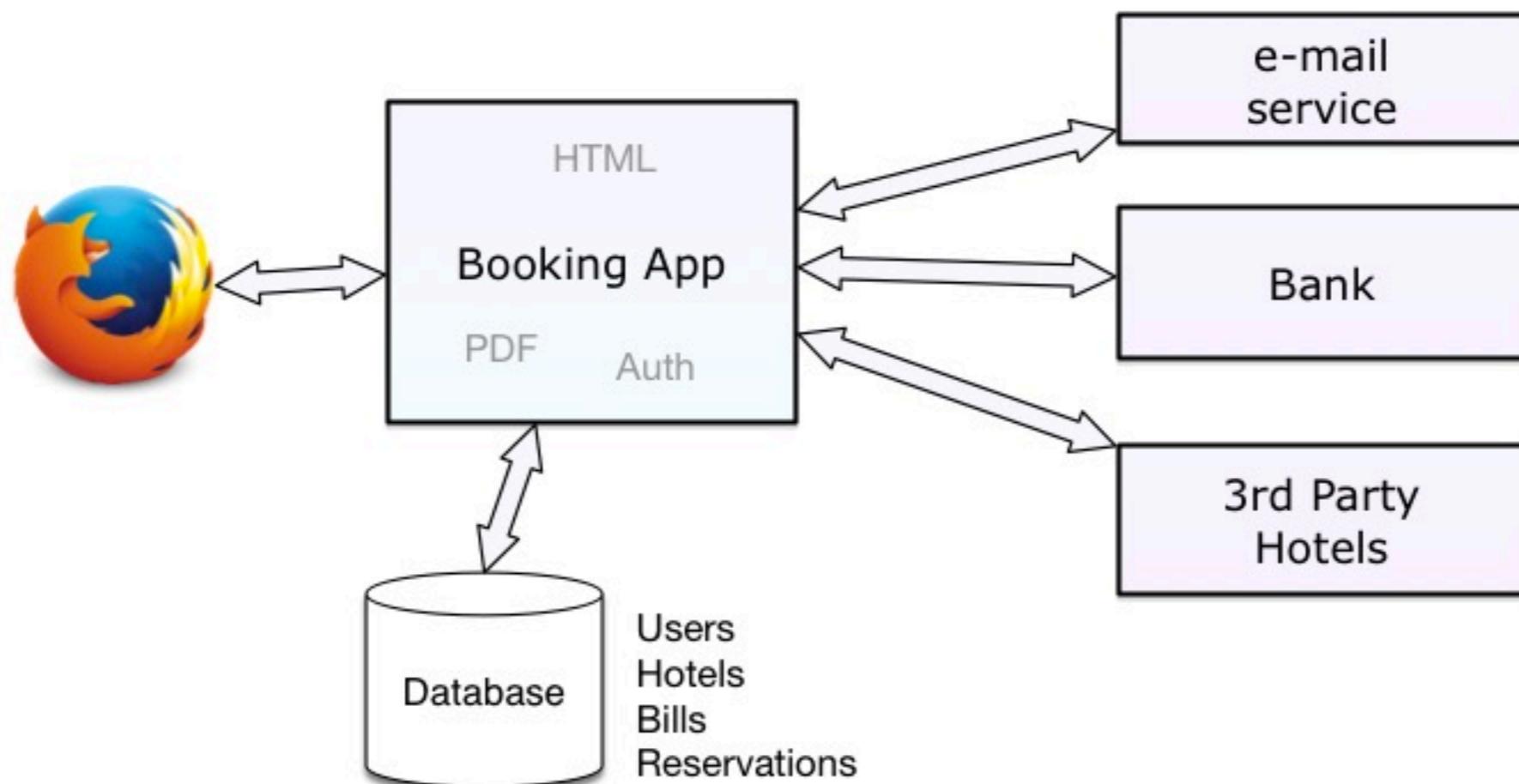
Replaceable

It is easier to replace if there is a need for it

<https://trello.com/>



Monolithic



Monolithic



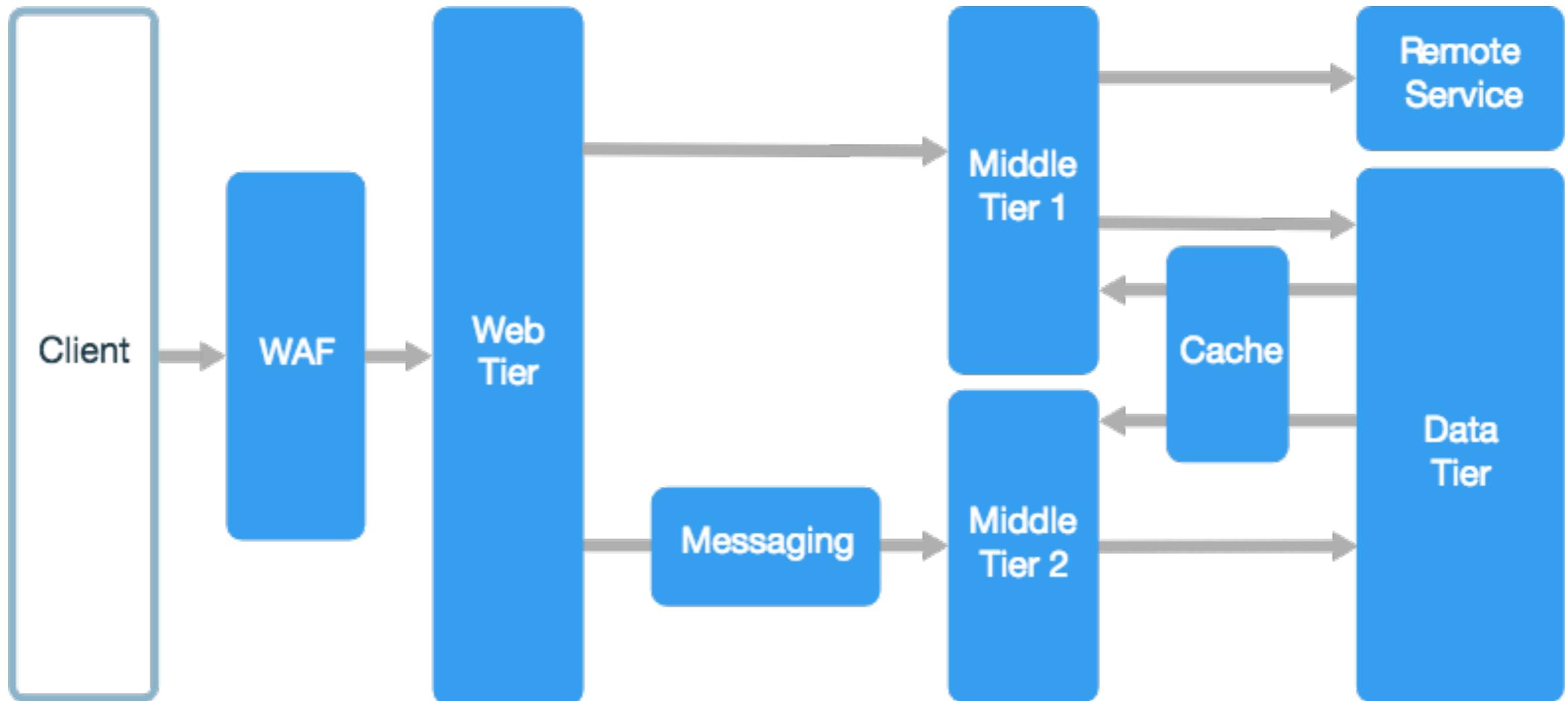
Monolithic



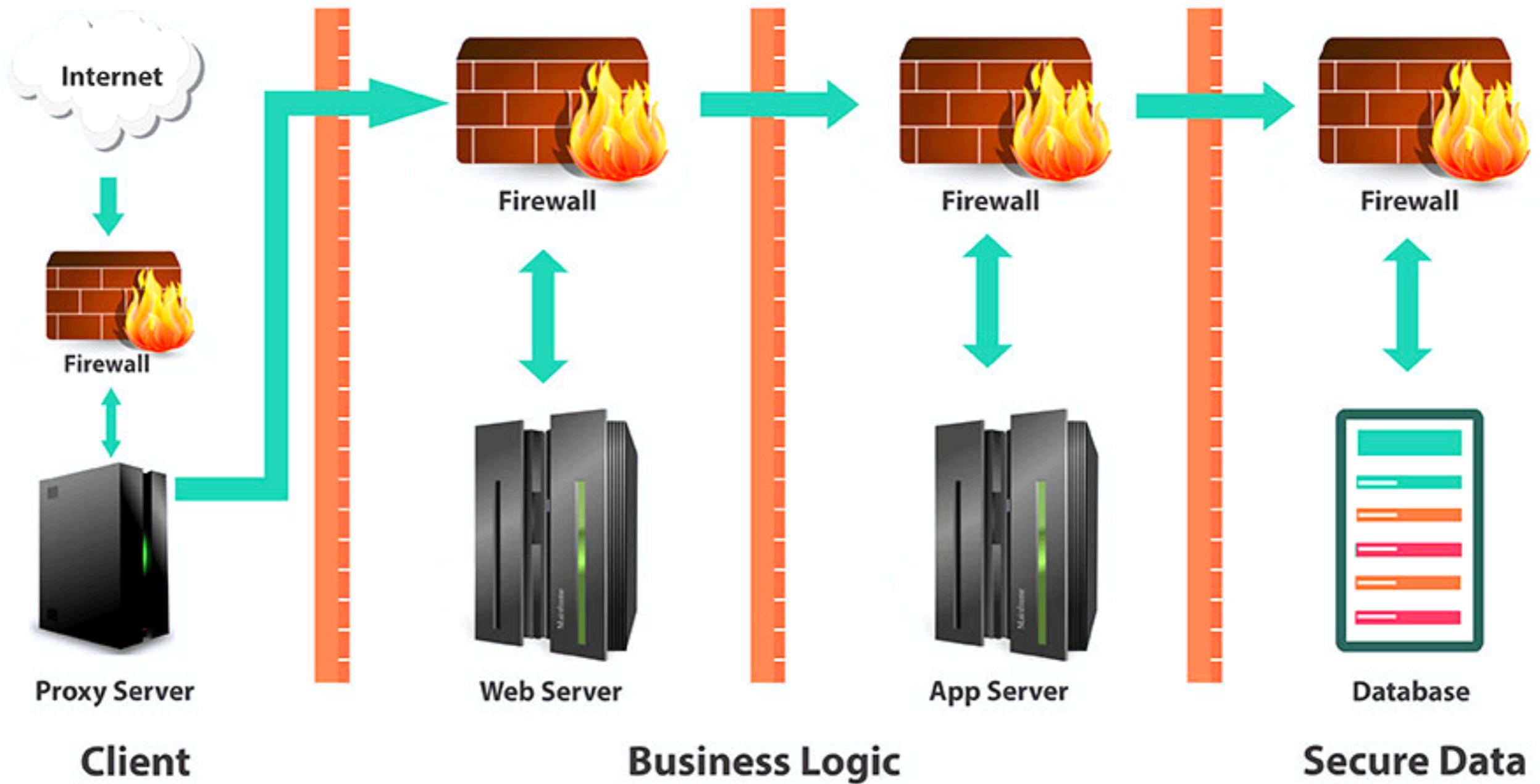
Monolithic



N-Tier



N-Tier



Problem ?



SOA



SOA

Focus on business/service reusability



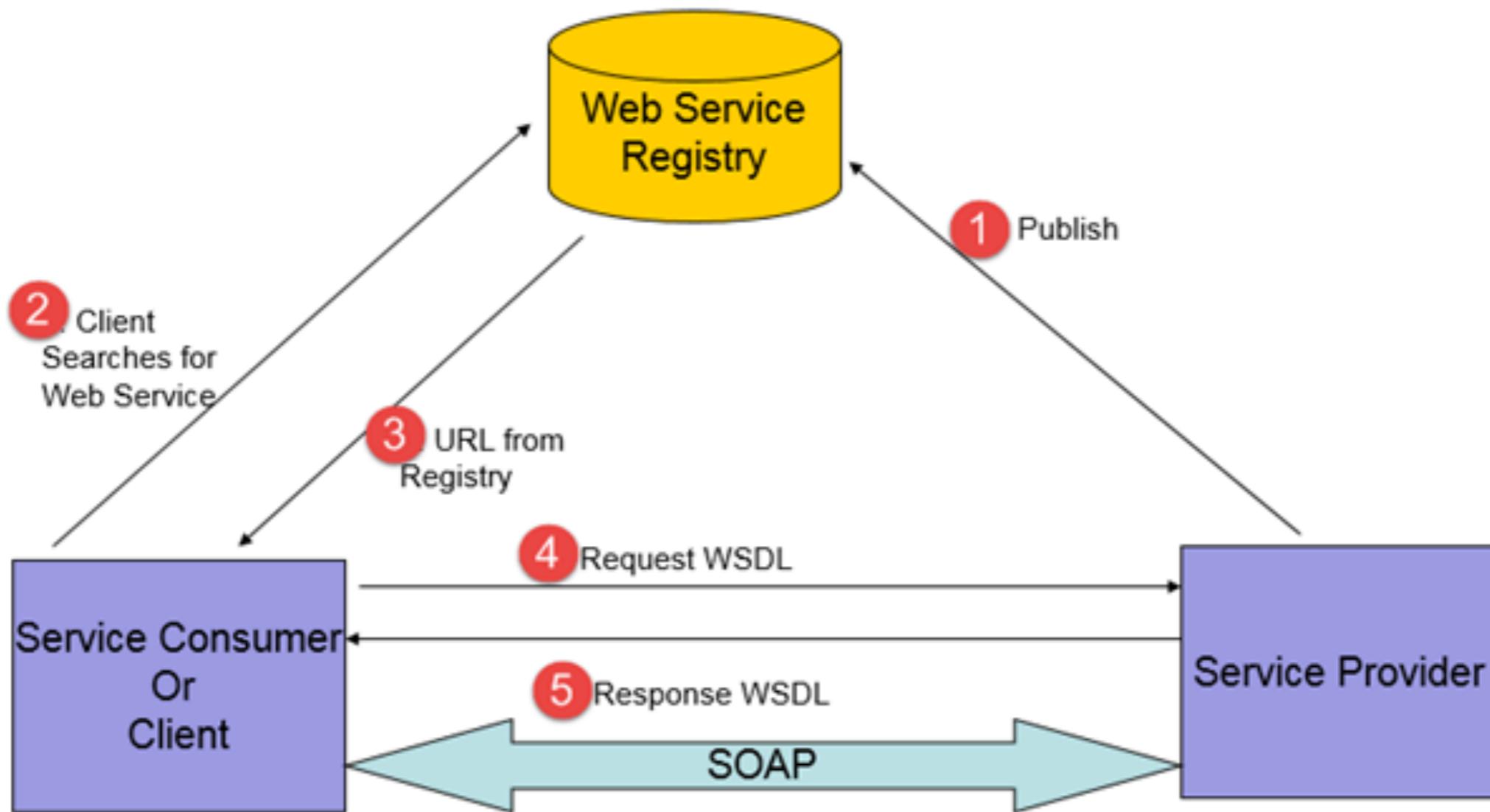
SOA

For communication it use ESB



WebService

WebService is a implementation of SOA



Home >

OPINION

Top 10 Reasons Why People are Making SOA Fail

SOA is not something you buy, it is something you do. Research shows us that very few companies are doing well. But the reasons for so many failures are usually people issues, not technology issues.



By [Mike Kavis](#)

CIO | JUL 18, 2008 8:00 AM PT

2008

<https://www.cio.com/article/2434865/service-oriented-architecture/top-10-reasons-why-people-are-making-soa-fail.html>



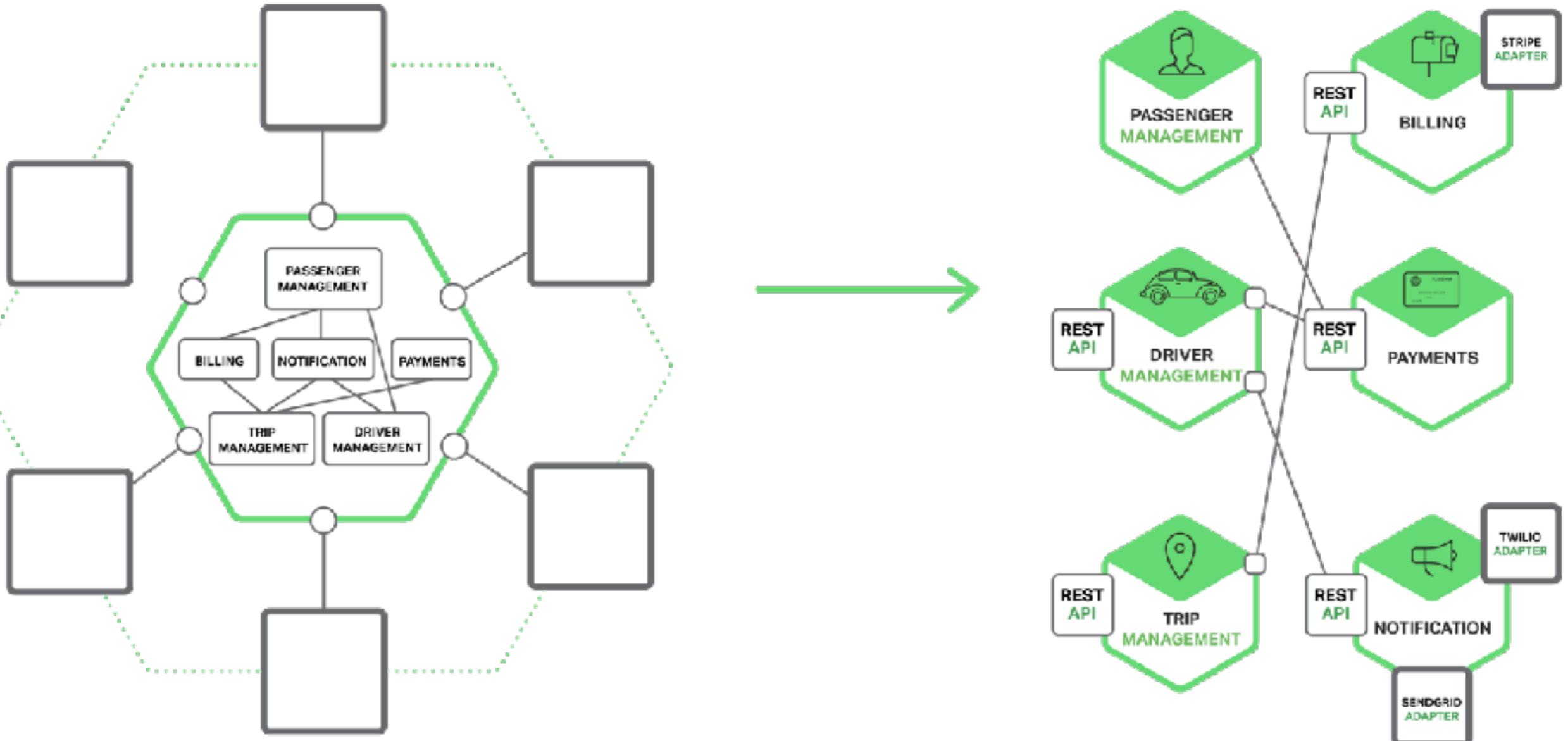
Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Microservices



Microservices



<https://www.nginx.com/>



What are Microservices ?



**“Microservices
is about people”**



Microservices

Engineering approach



Microservices

Engineering approach

**Decompose application into single-function
modules**



Microservices

Engineering approach

**Decompose application into single-function
modules**

Well-defined interfaces



Microservices

Engineering approach

**Decompose application into single-function
modules**

Well-defined interfaces

Independently deploy



Microservices

Engineering approach

Decompose application into single-function
modules

Well-defined interfaces

Independently deploy

**Operate by small team who own the lifecycle
of service**



Microservices

Accelerate delivery by minimizing communication and coordination between people

Reduce scope and risk of change



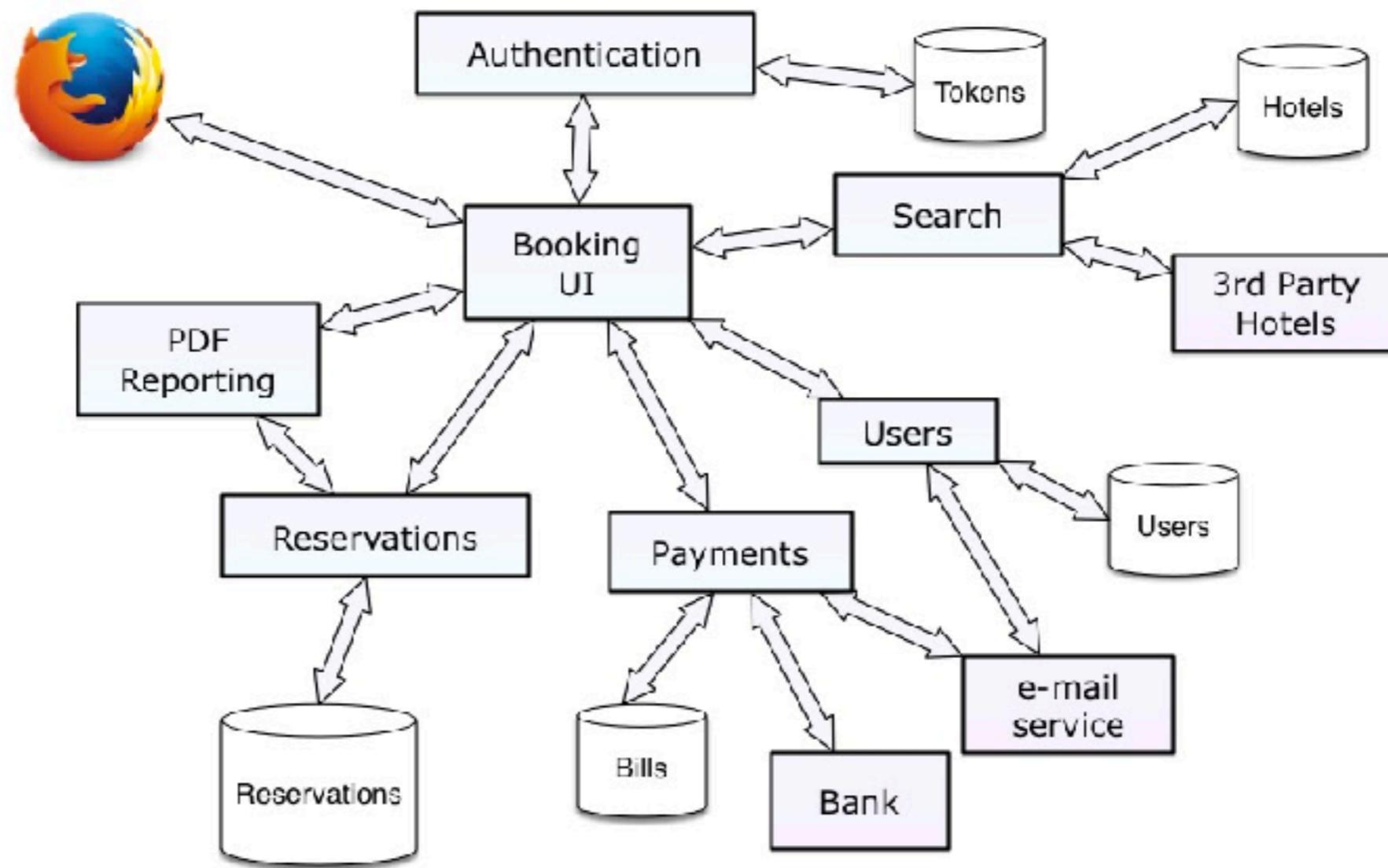
Microservices

Decompose
Single-function
Well-defined interfaces
Independent
Small team
Lifecycle
Minimize communication
Reduce the scope of change



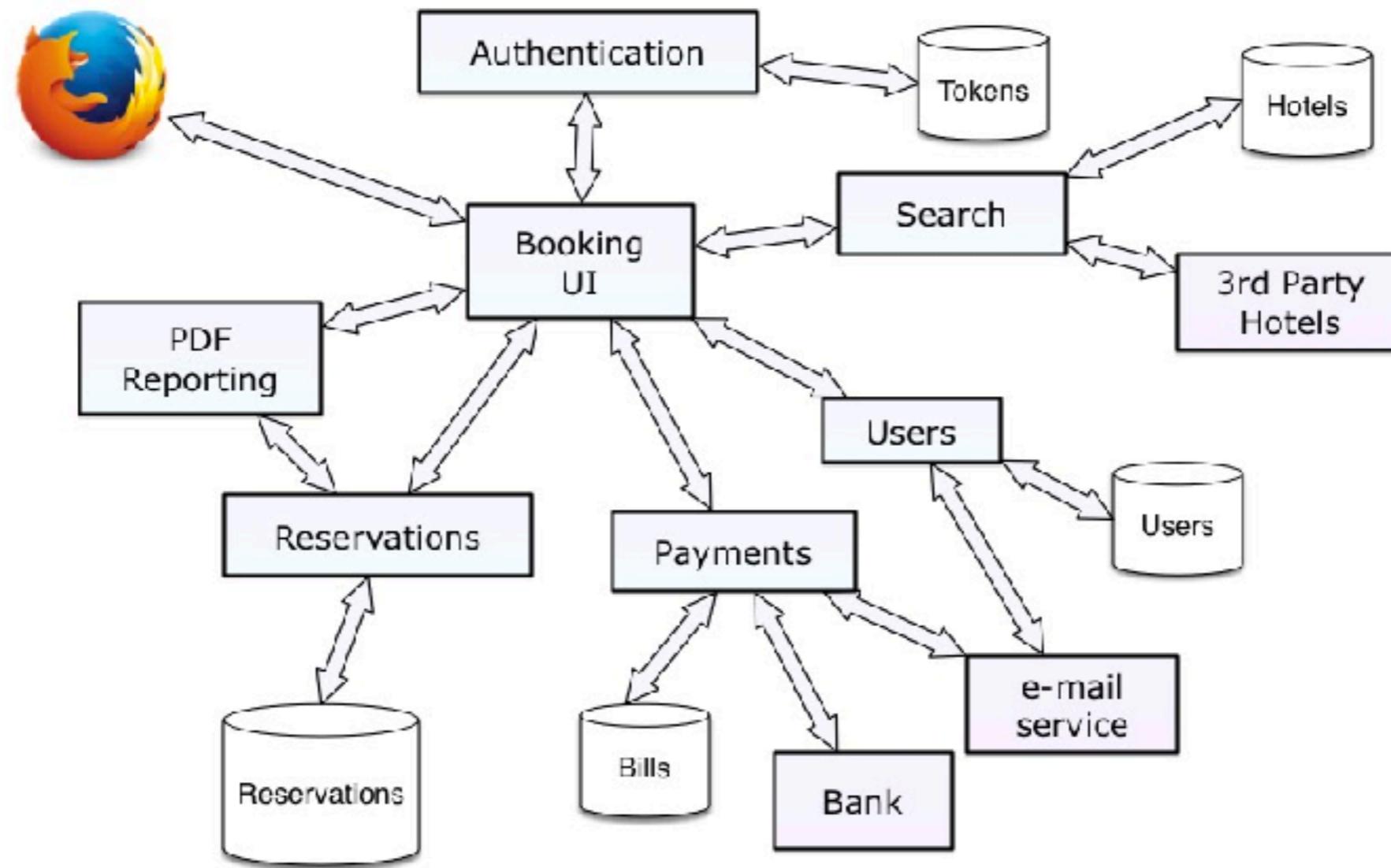
Microservices

Focus on decoupling of services



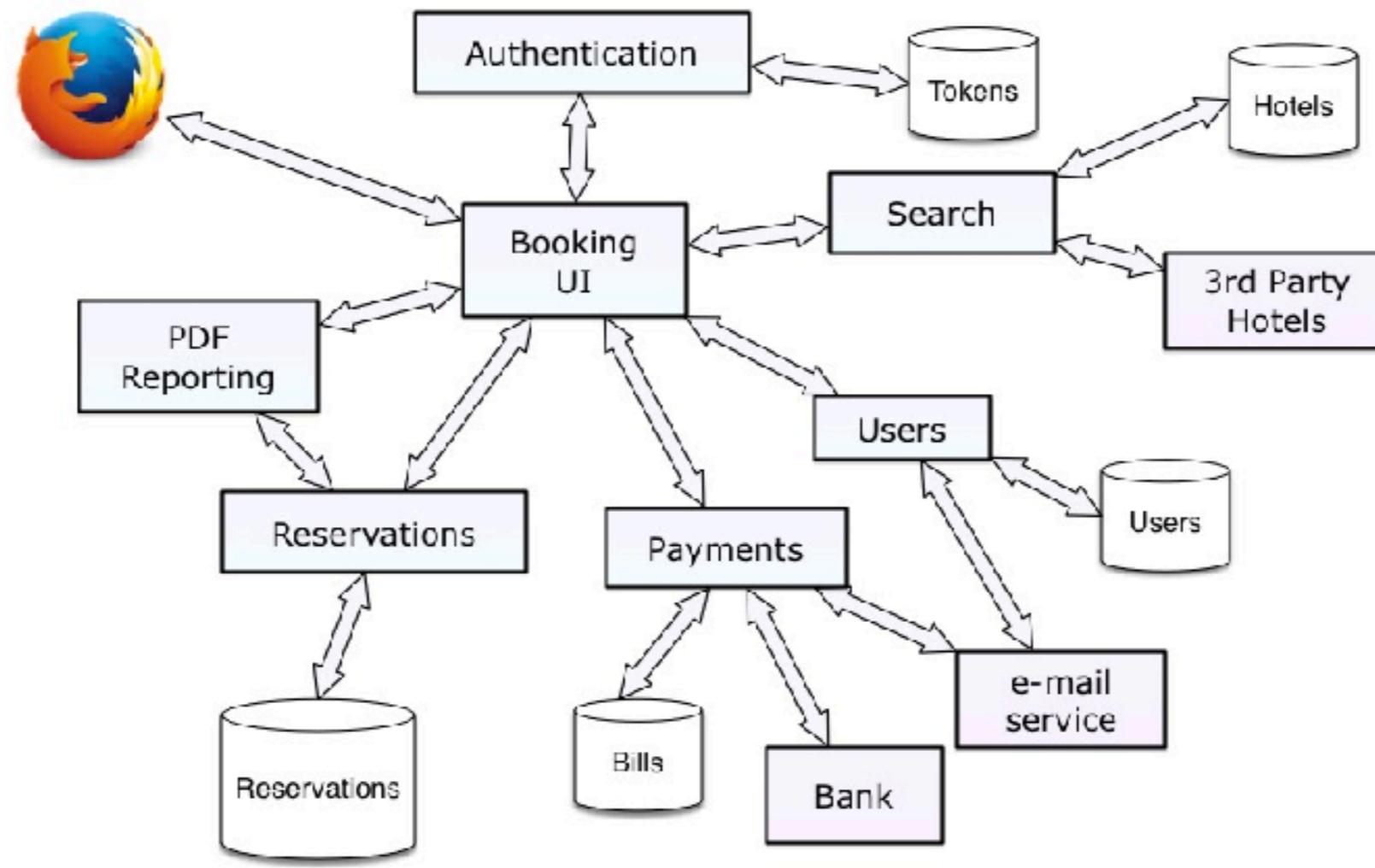
Microservices

Application/Web server is not required



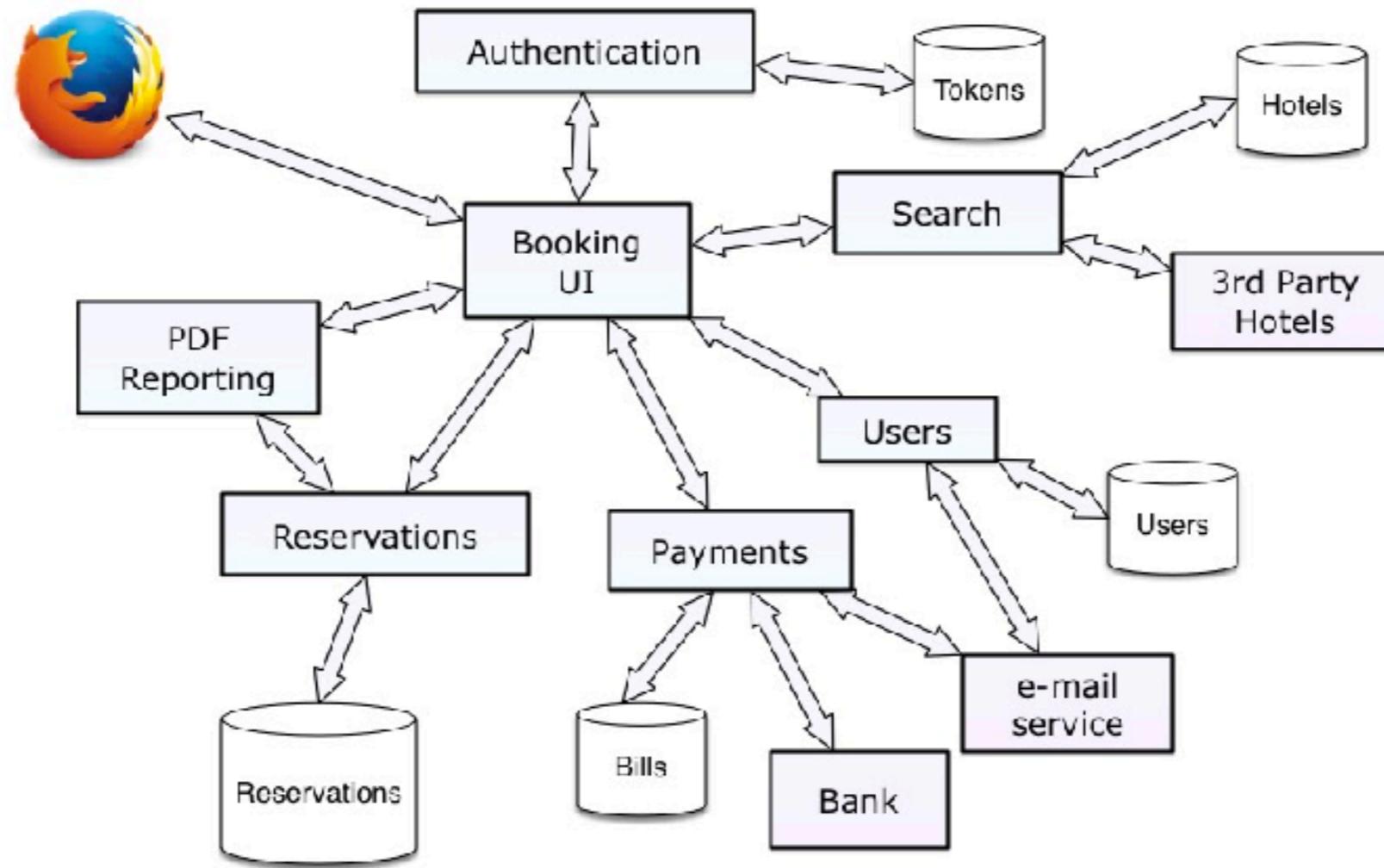
Microservices

Communication use a simple messaging system



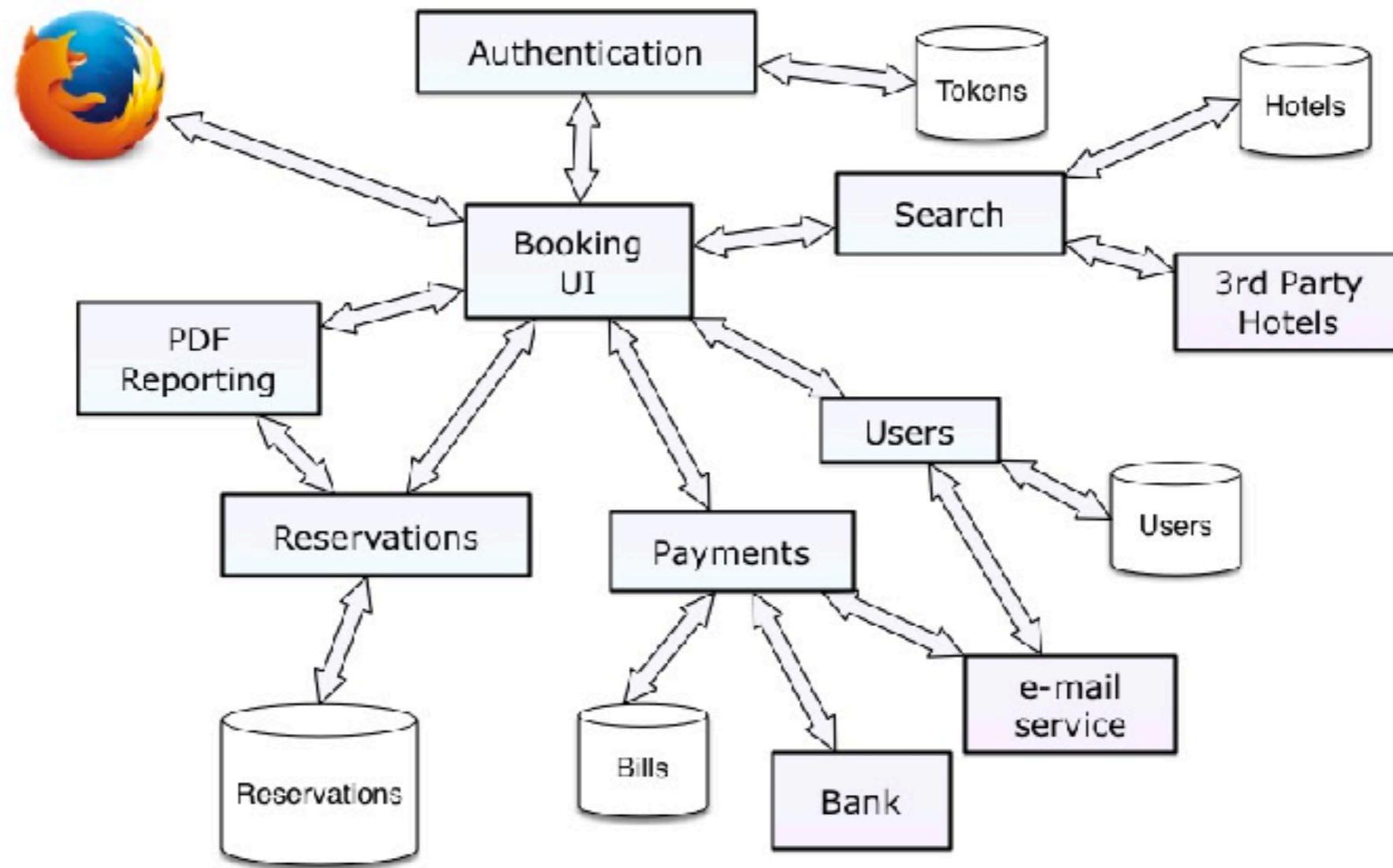
Microservices

Focus on team collaboration and **freedom** of choice



Microservices

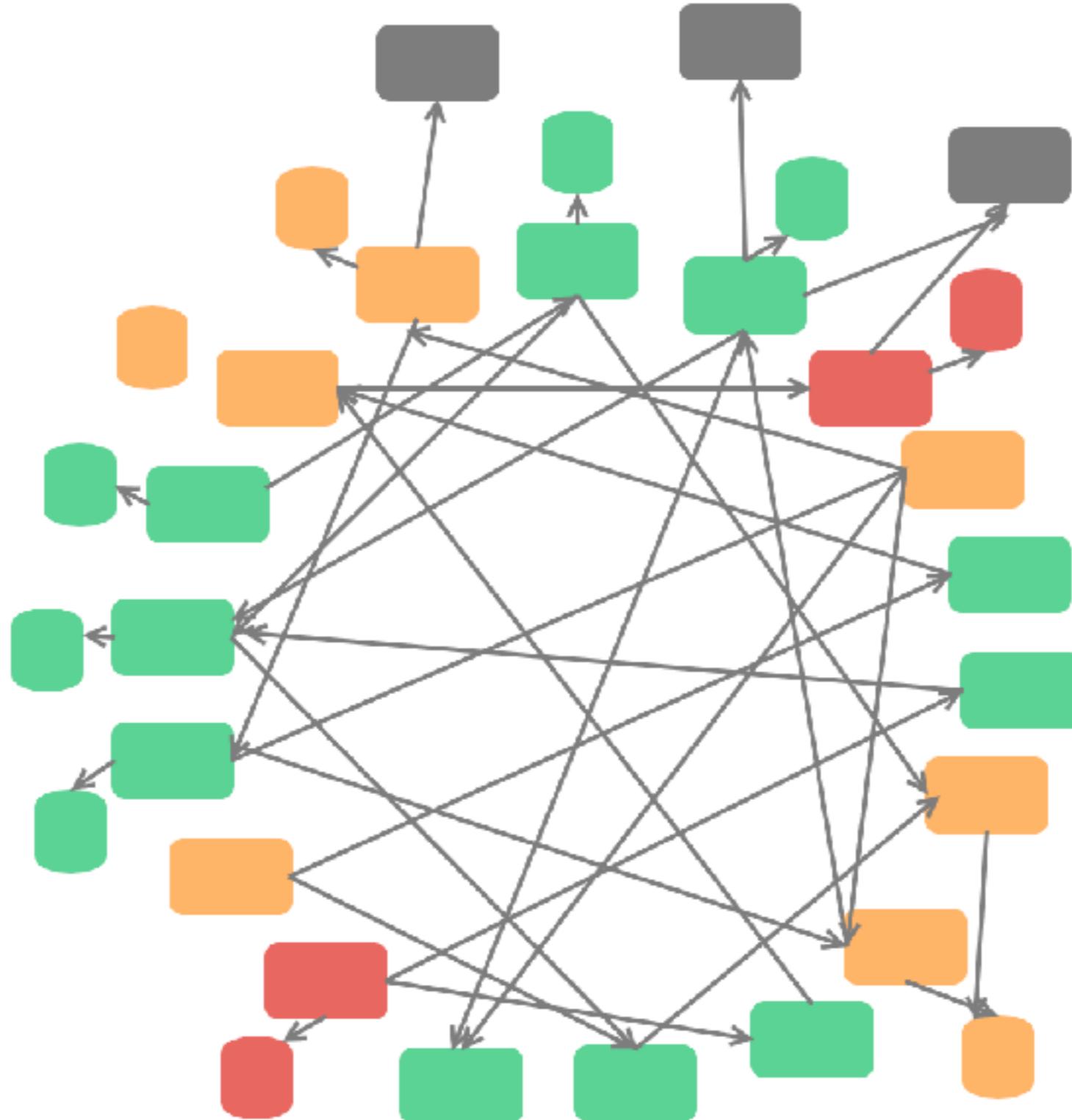
Minimize on sharing



Microservices spaghetti

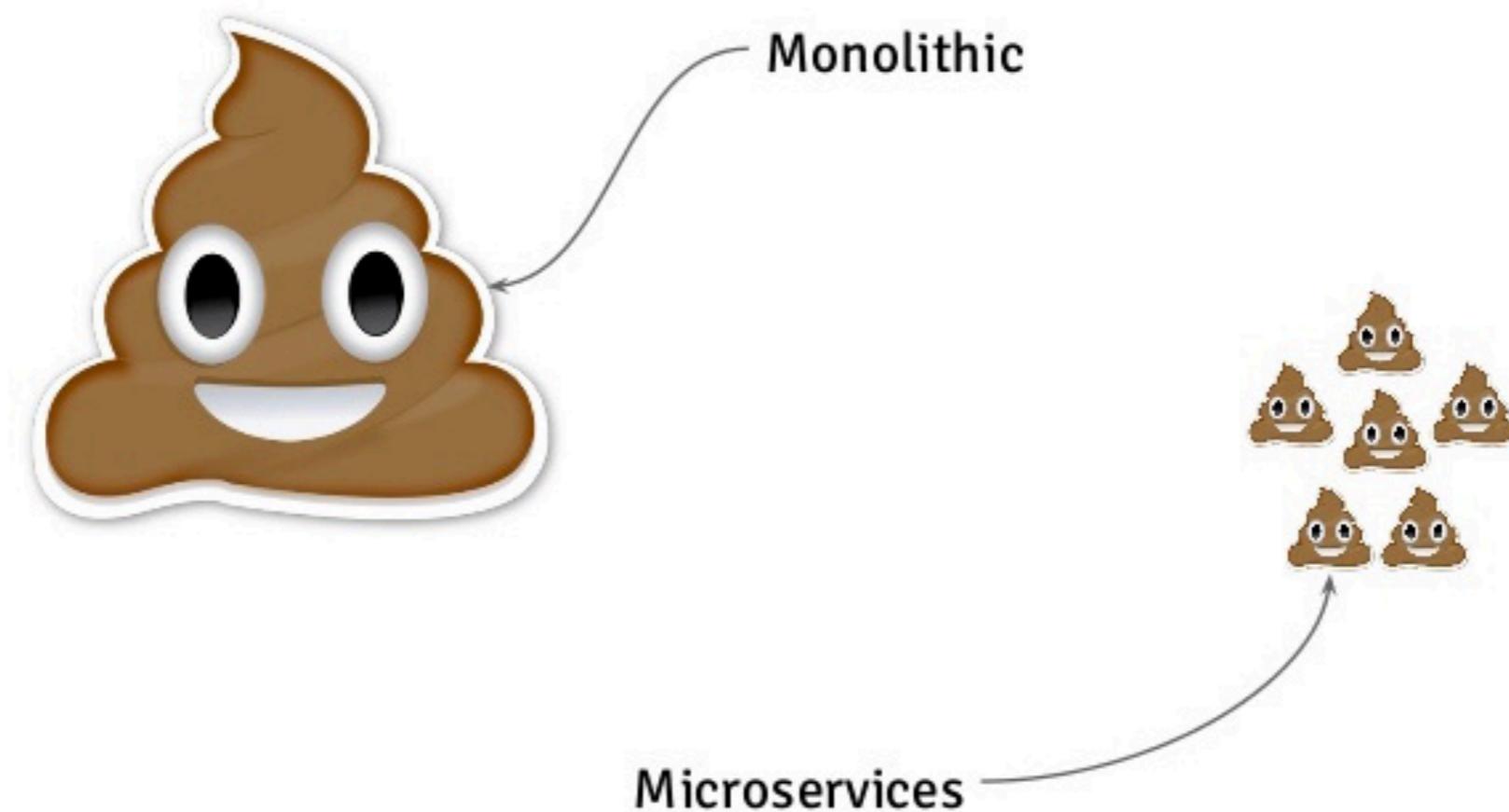


Microservices spaghetti



Microservices spaghetti

Monolithic vs Microservices



SOA vs Microservices



SOA vs Microservices

2000's

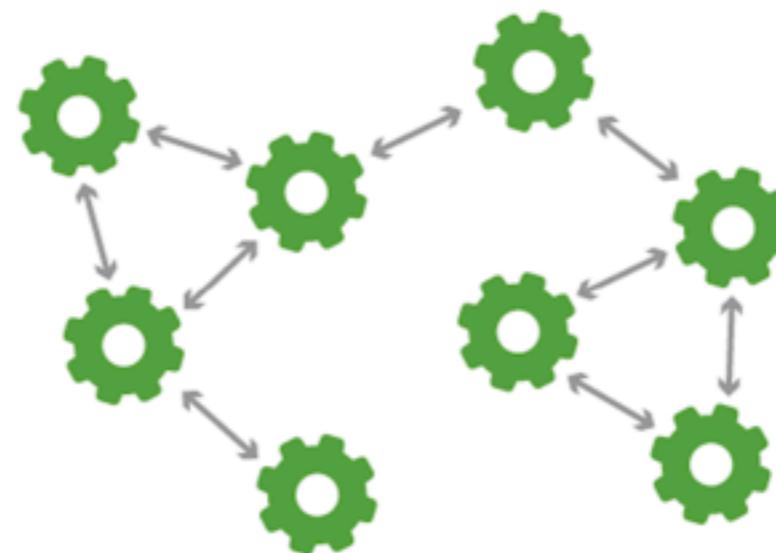
SERVICE ORIENTED ARCHITECTURE



SOA based applications are comprised of more loosely coupled components that use an Enterprise Services Bus messaging protocol to communicate between themselves.

2010's

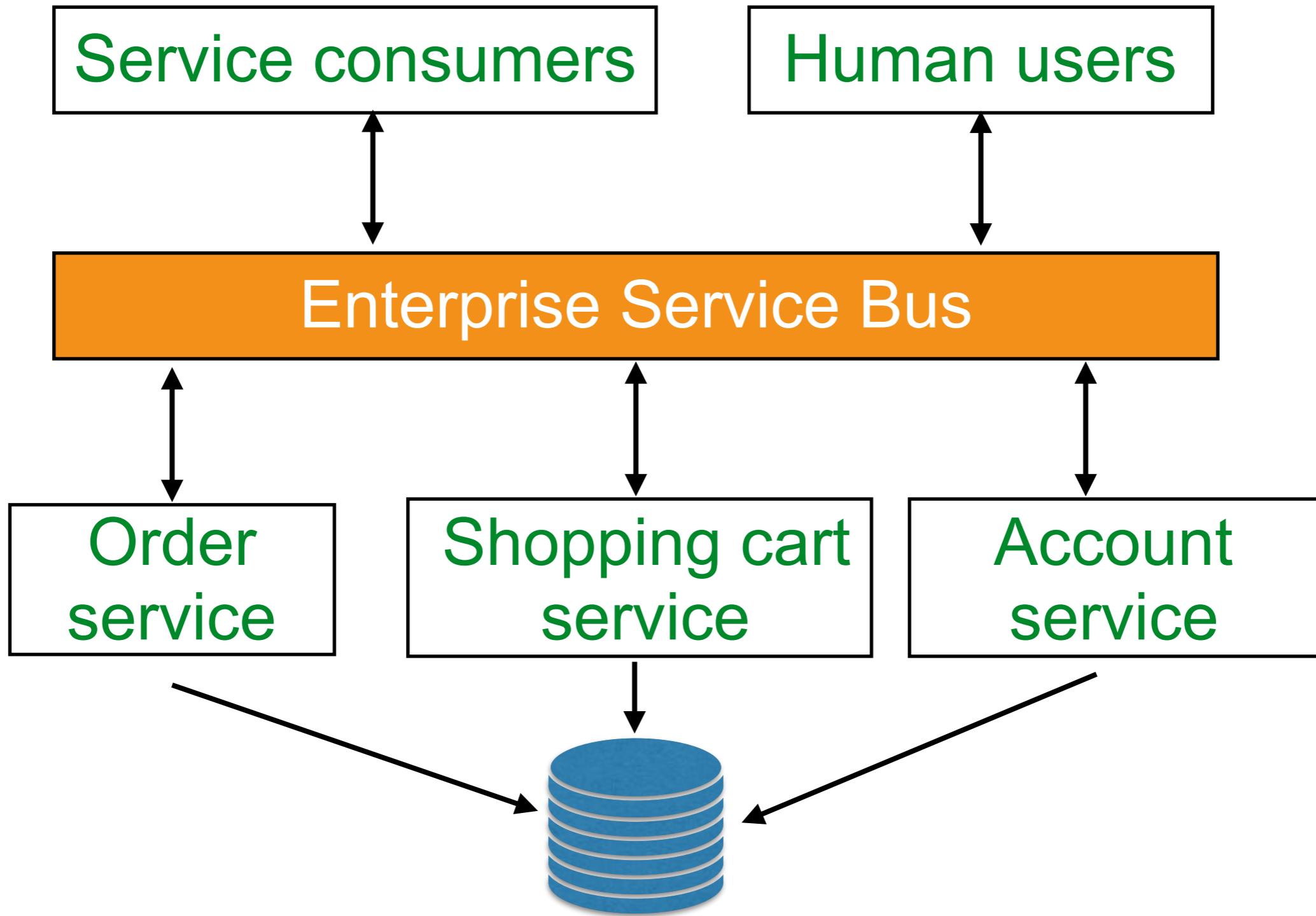
MICROSERVICES ARCHITECTURE



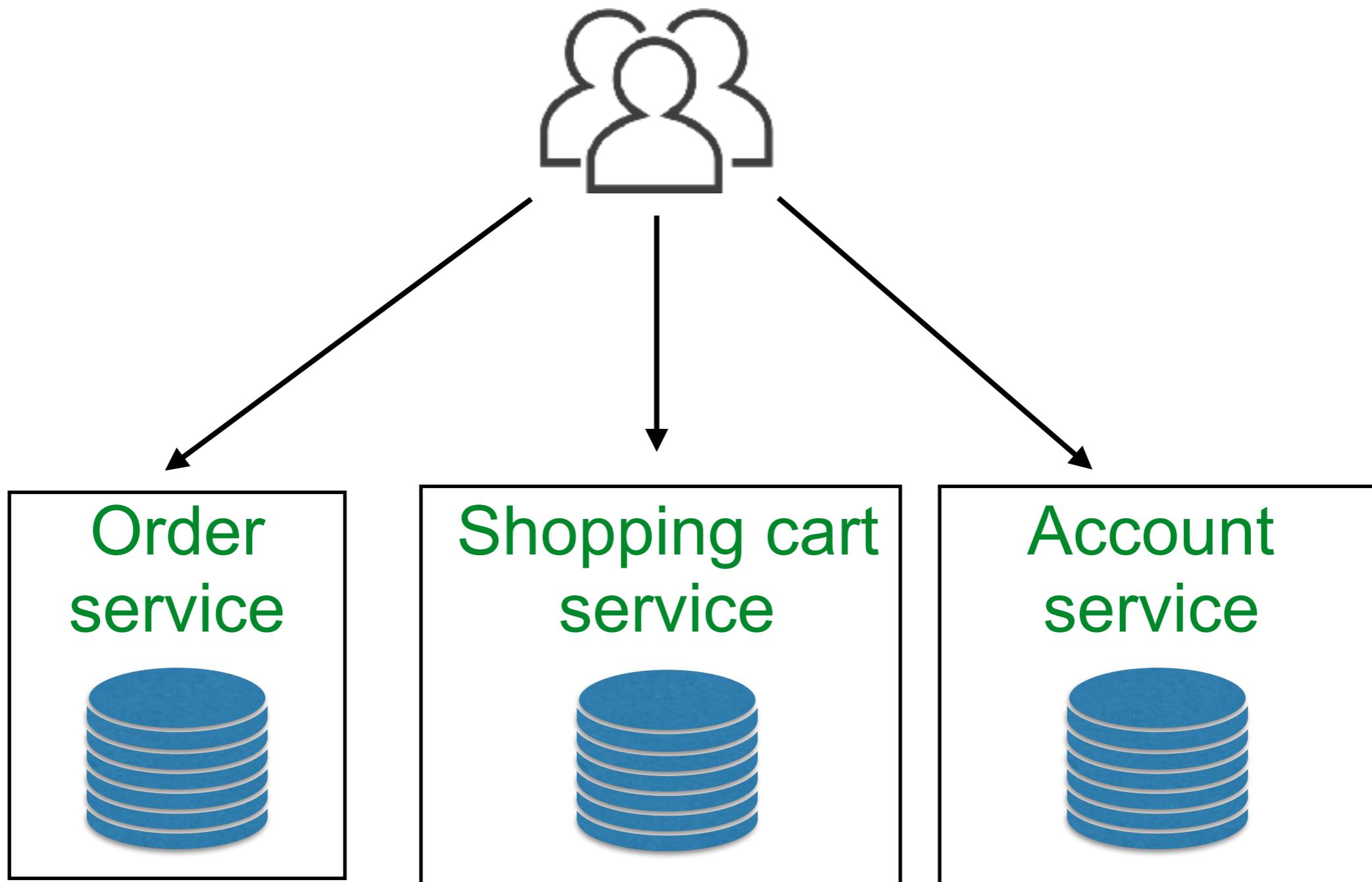
Microservices are a number of independent application services delivering one single functionality in a loosely connected and self-contained fashion, communicating through light-weight messaging protocols such as HTTP, REST or Thrift API.



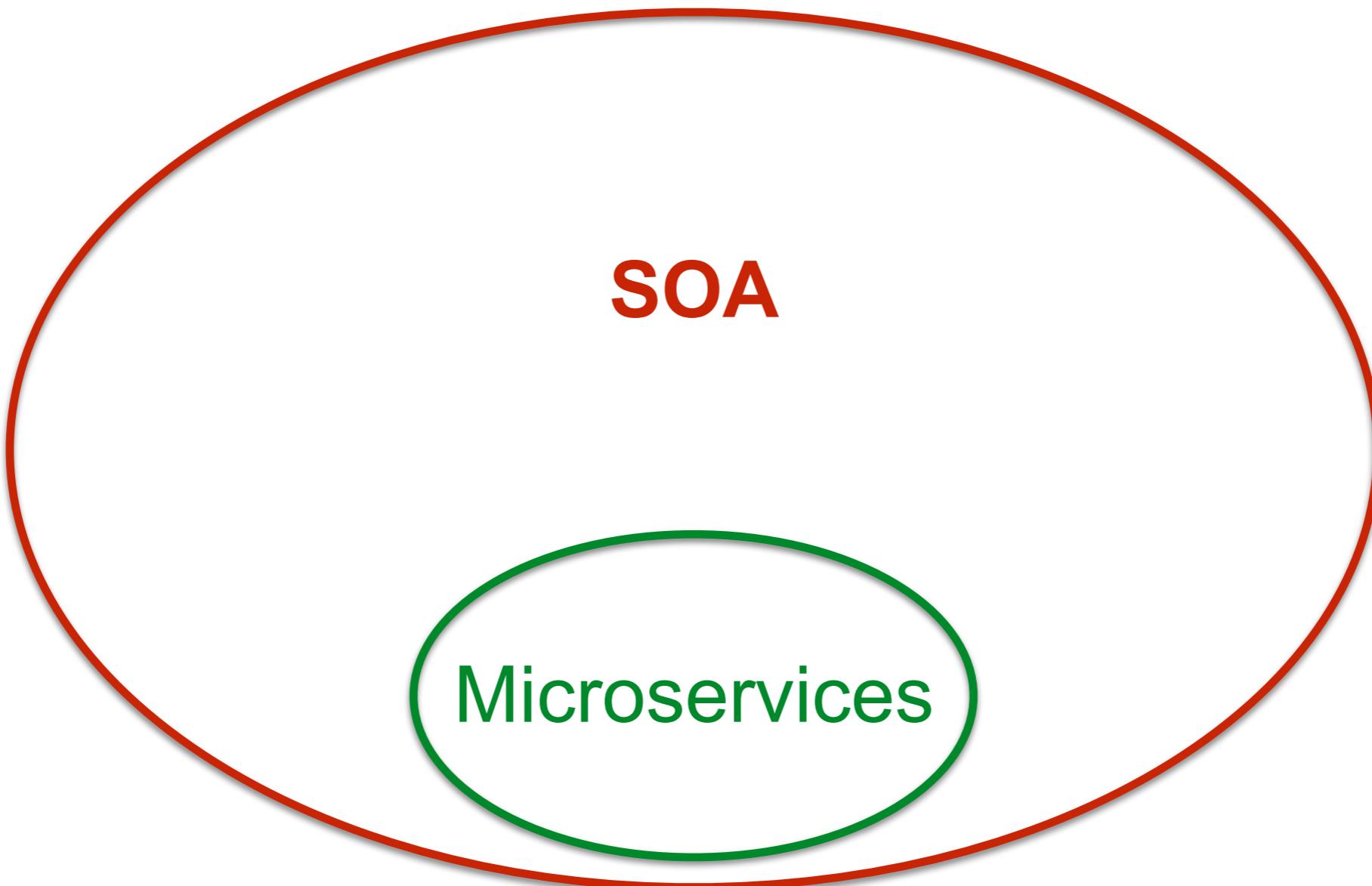
SOA



Microservices



SOA vs Microservices

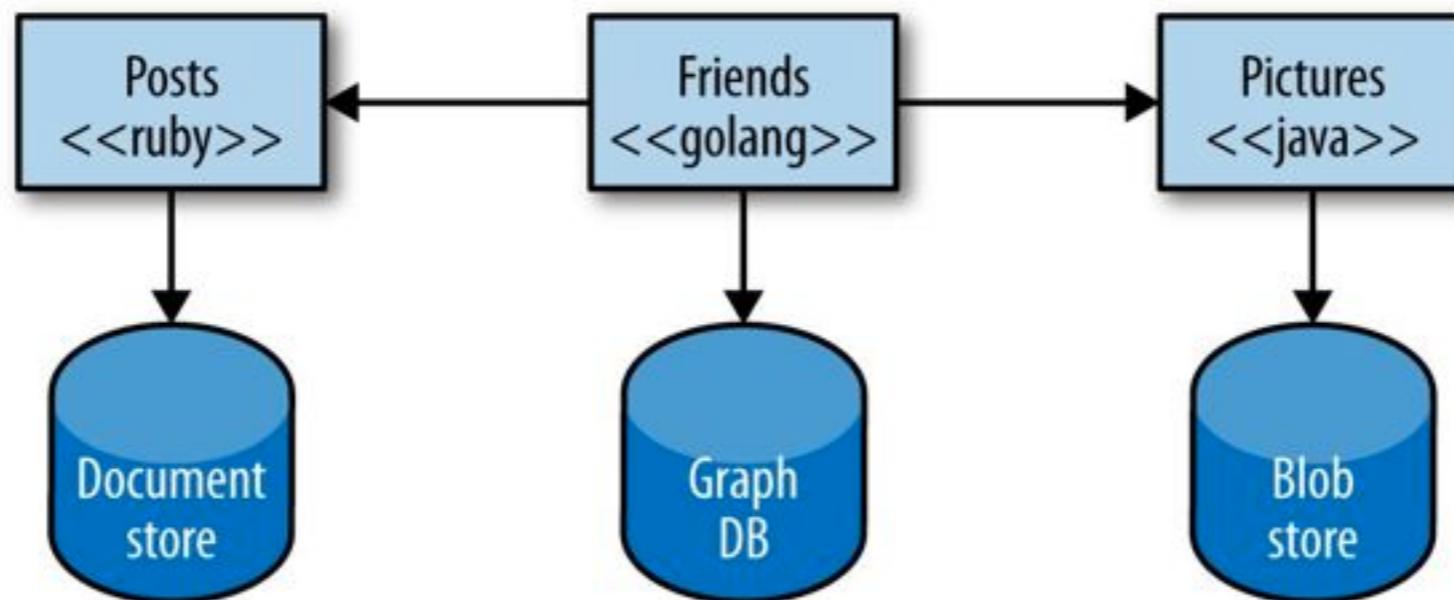


Key Benefits

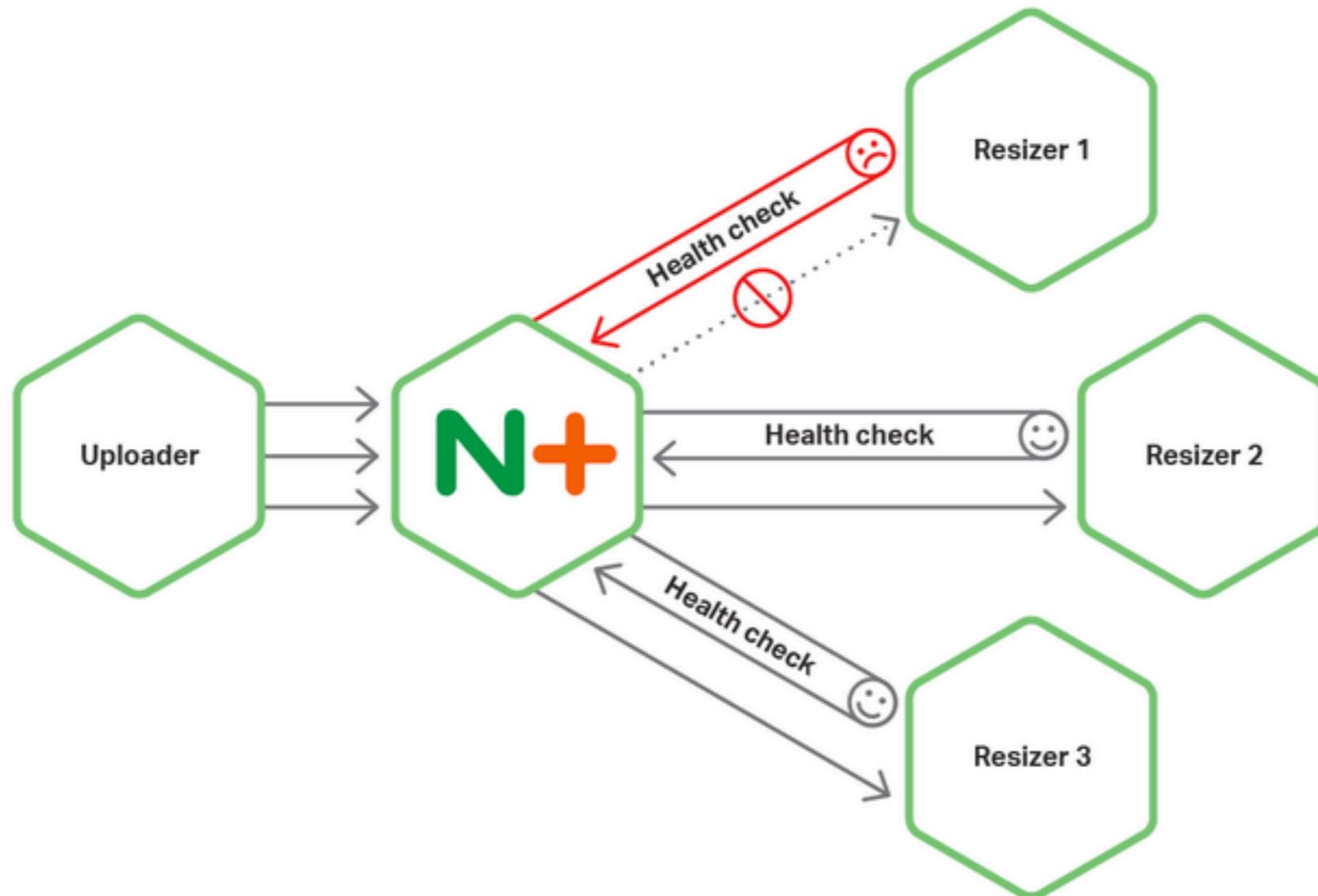


1. Technology heterogeneity

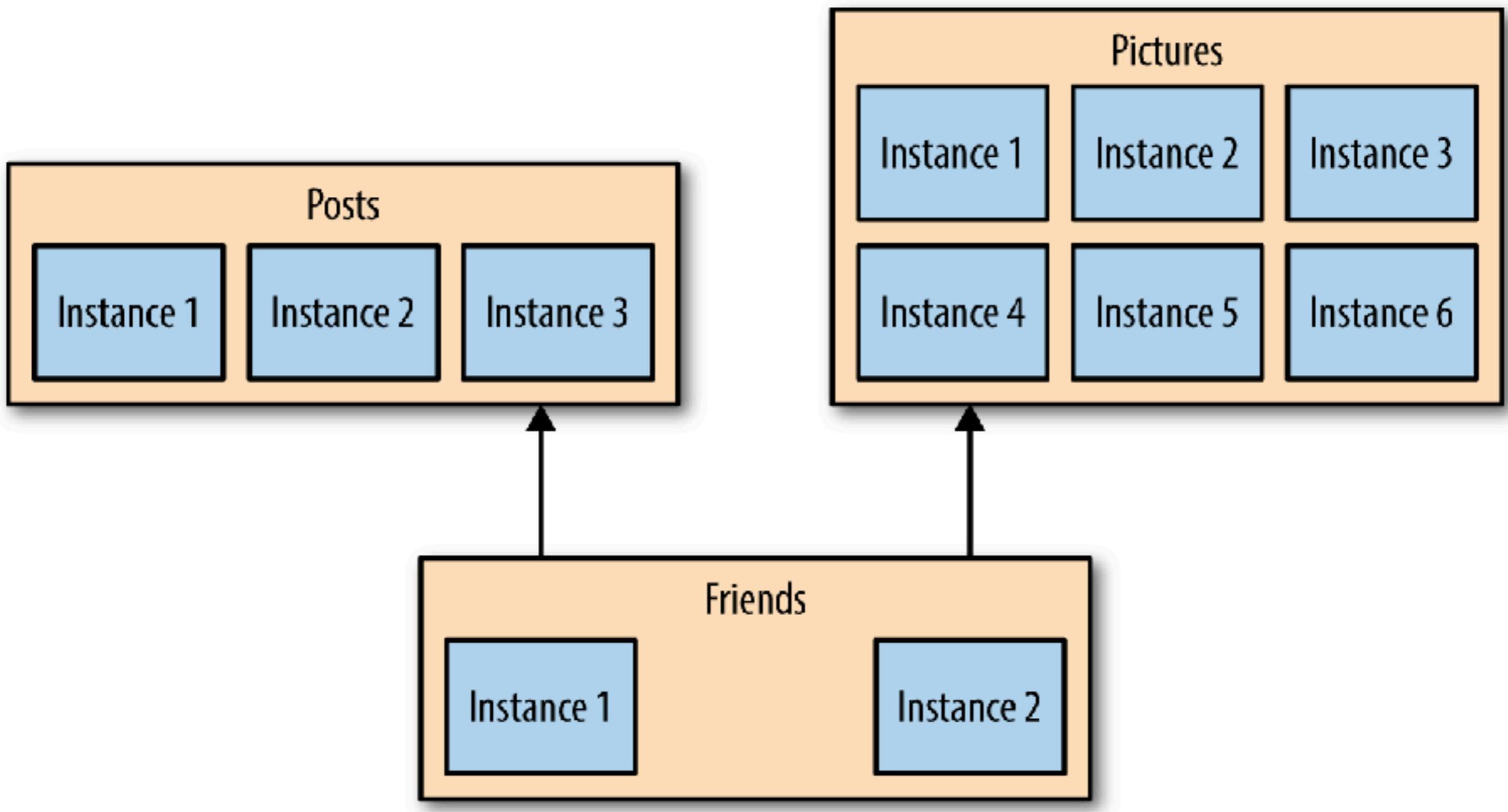
The right tool for each job



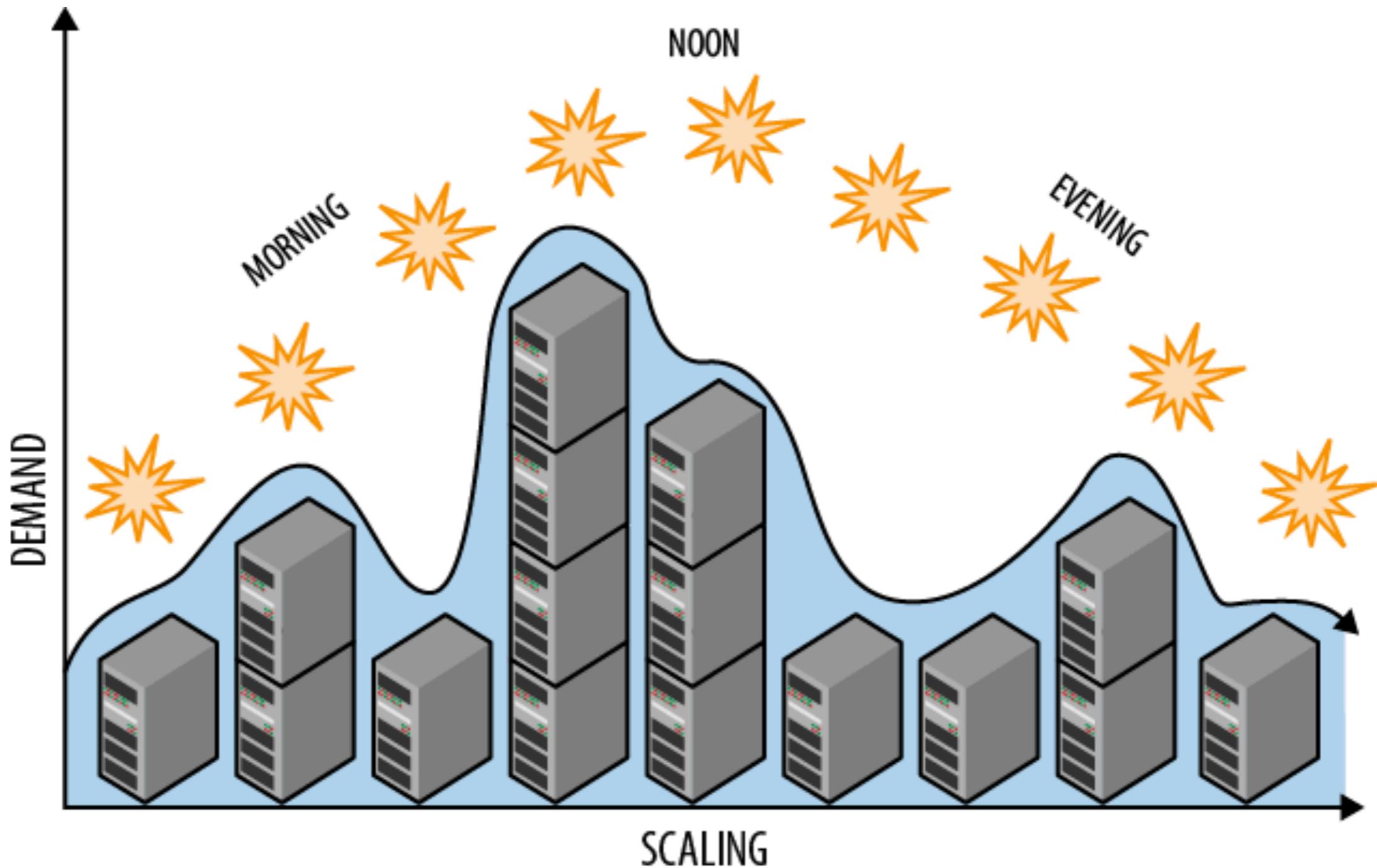
2. Resilience



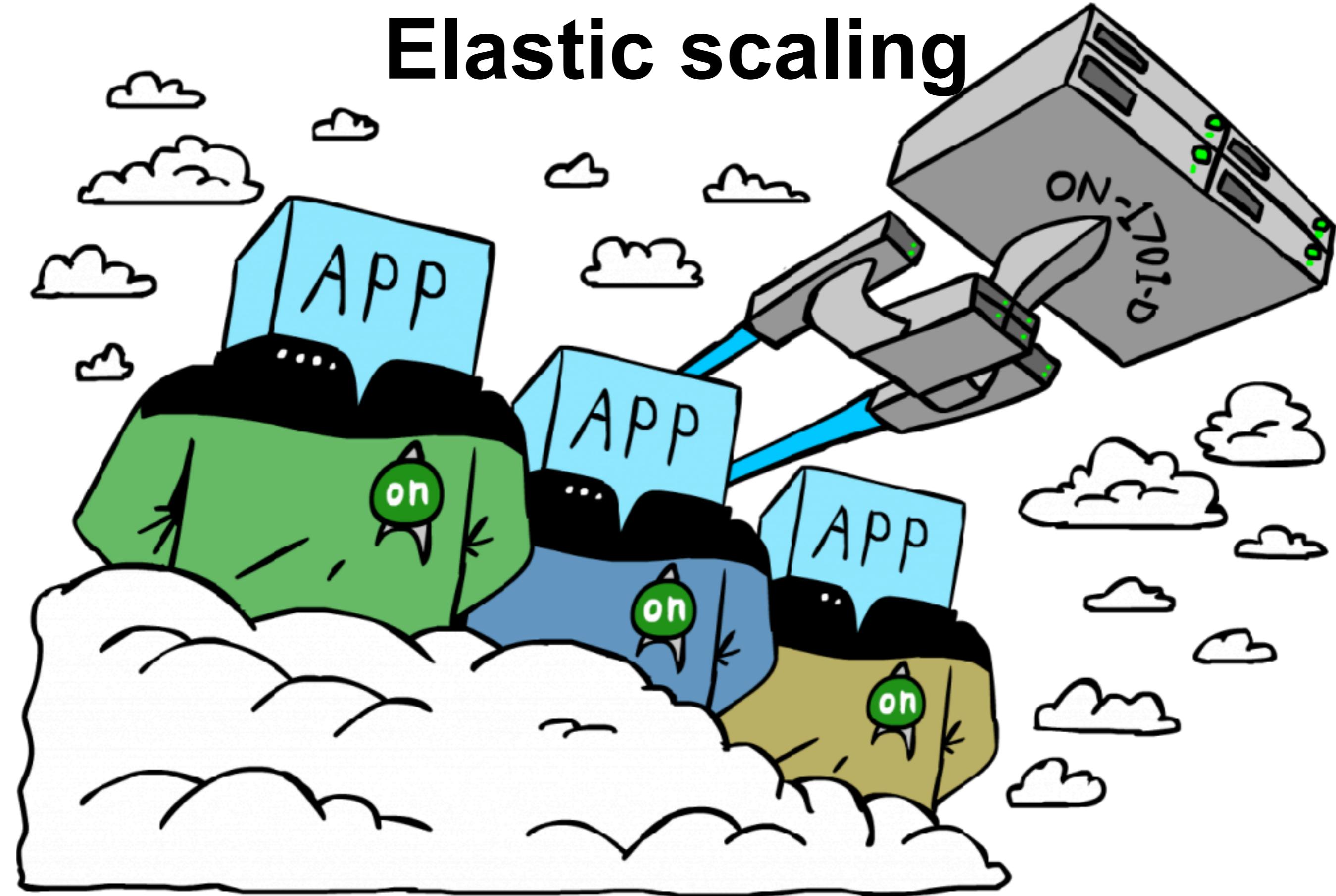
3. Scaling



Elastic scaling

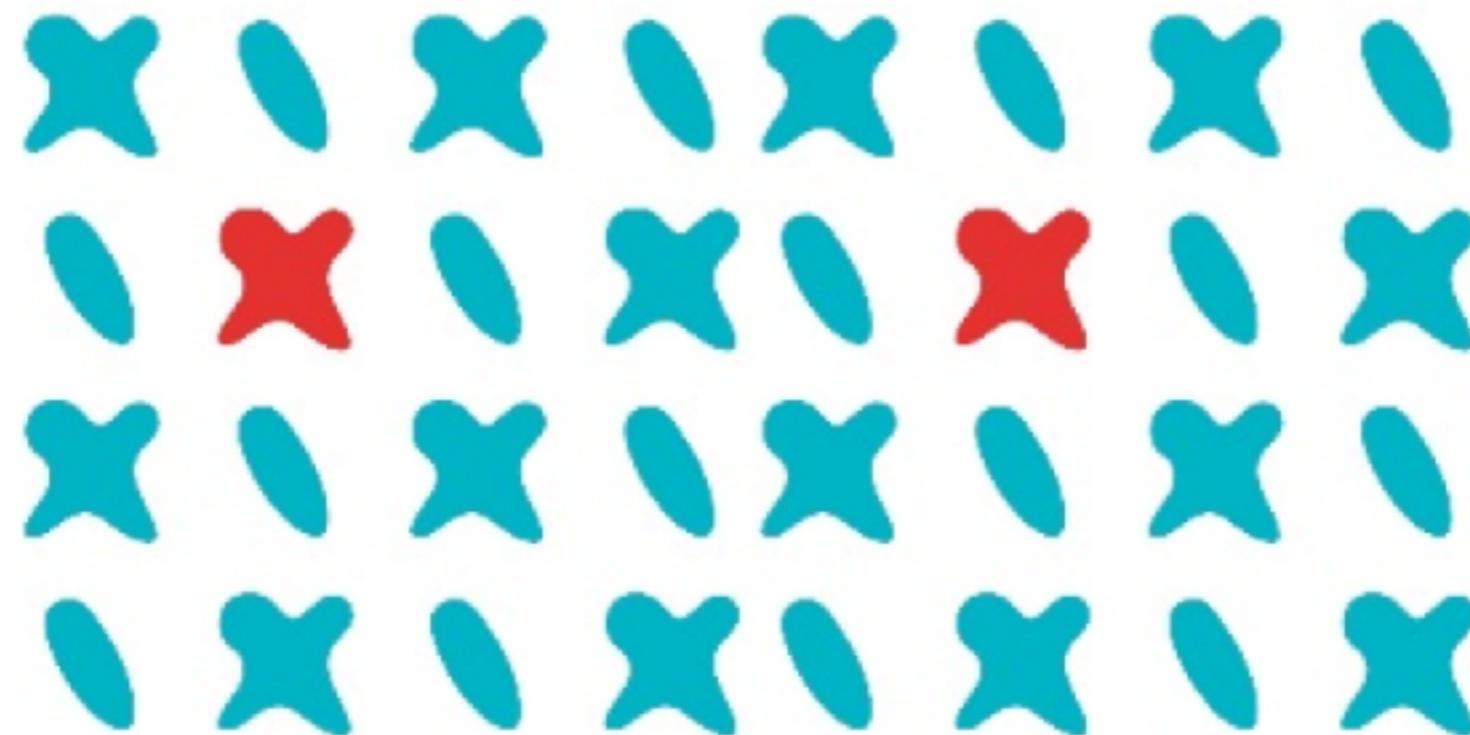


Elastic scaling



4. Ease of deployment

Deploys are faster, independent and problems can be isolated more easily

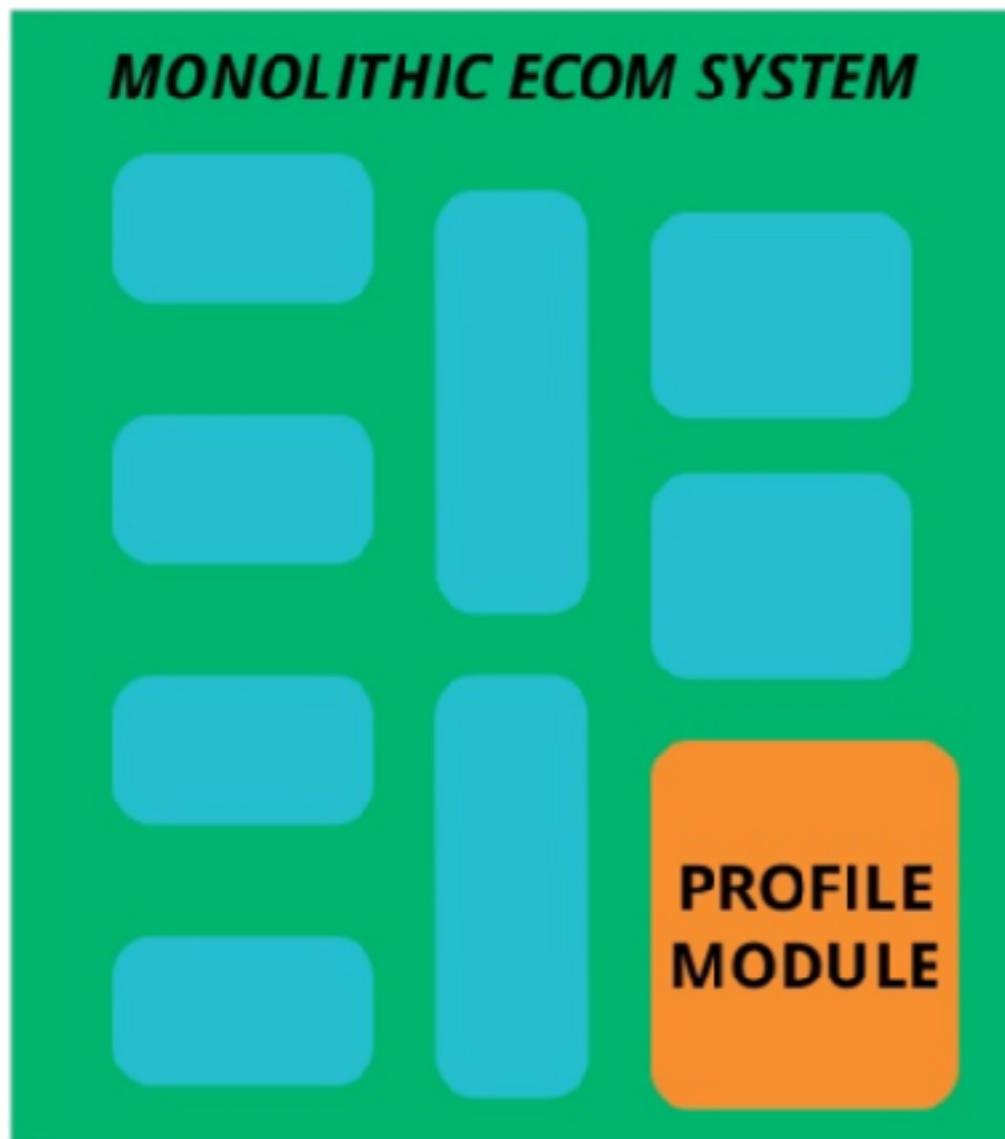


5. Organization alignment

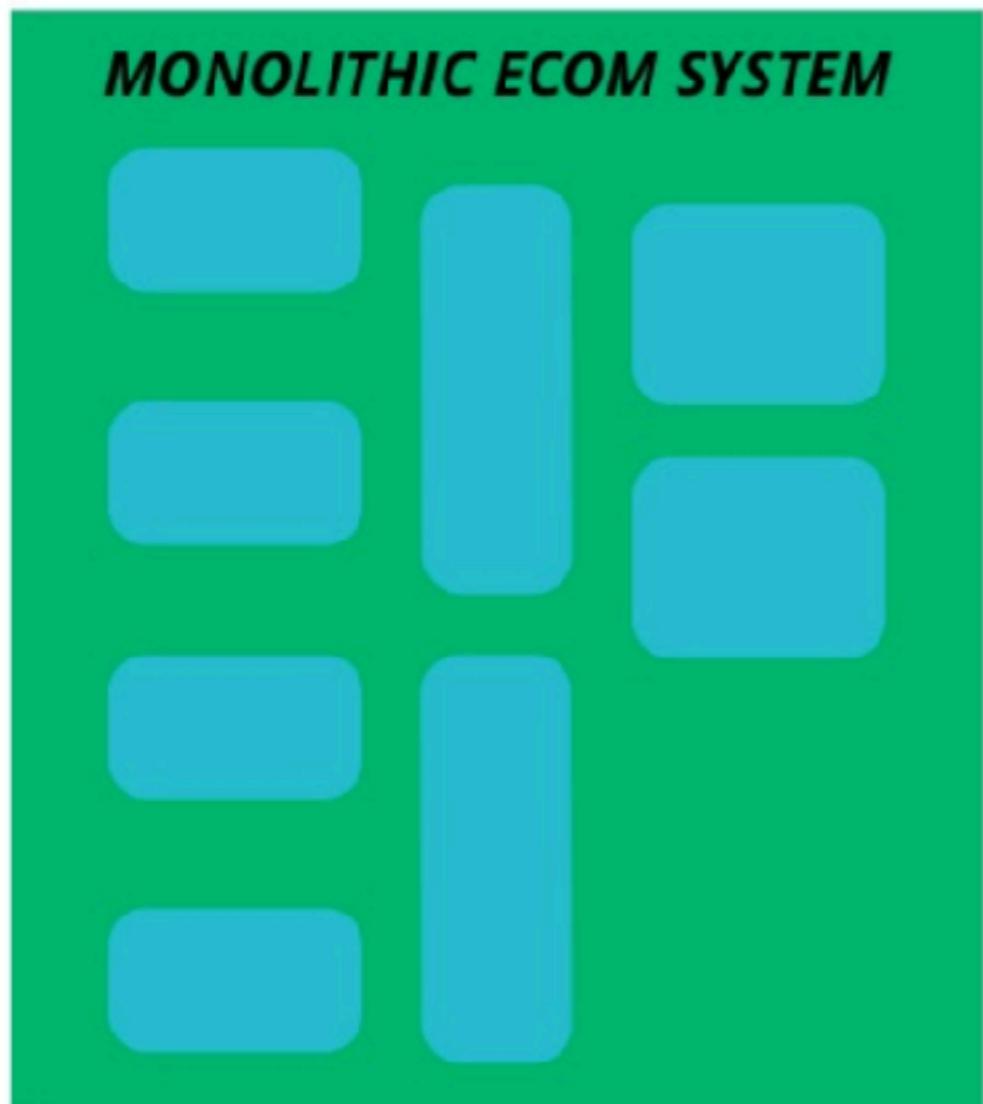
Small teams and smaller codebases



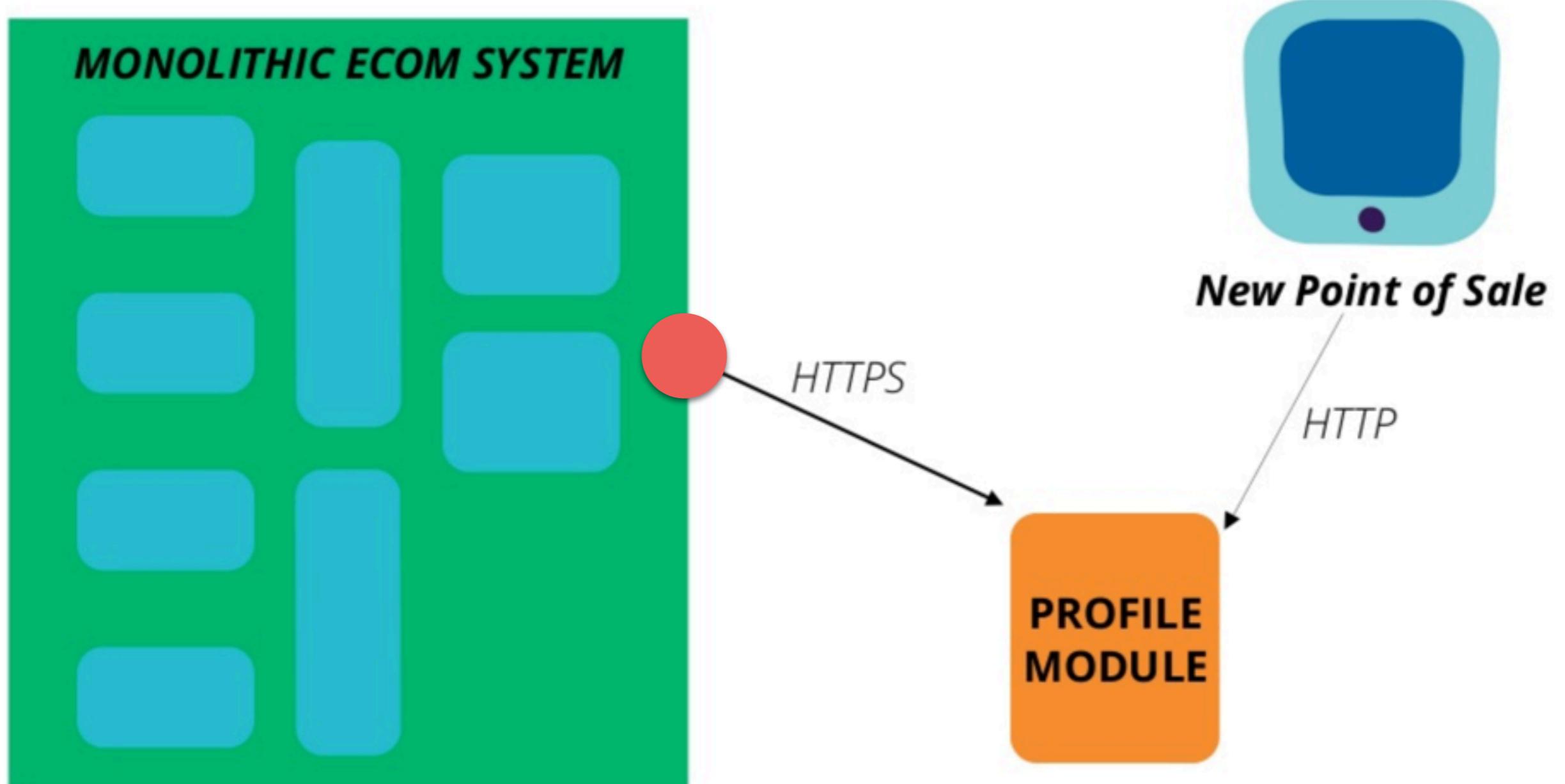
6. Composability and replaceability



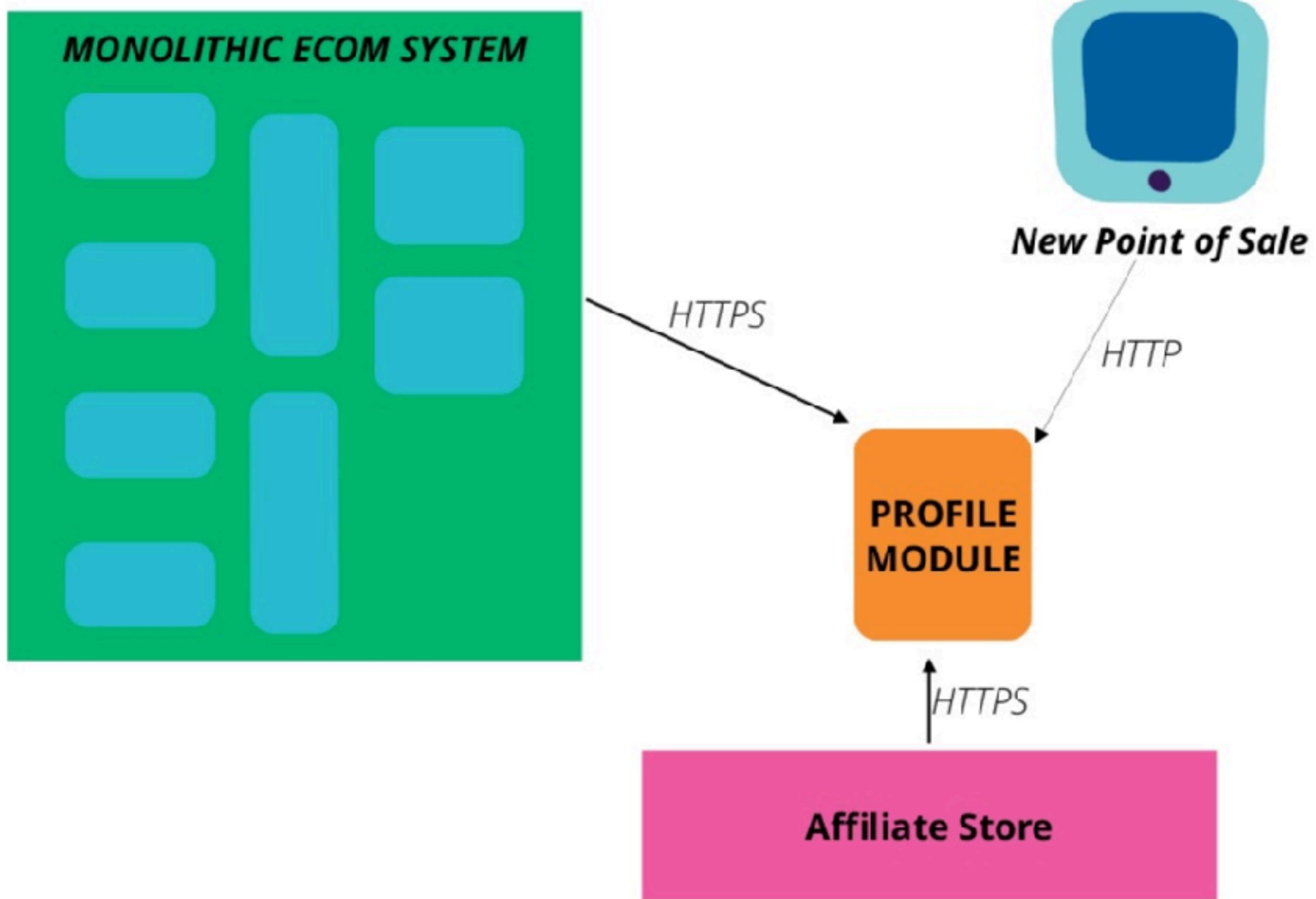
6. Composability and replaceability



6. Composability and replaceability



6. Composability and replaceability



Characteristics



1. Responsible for a single capability



Types of capabilities

Business capability
Technical capability





Booking System



Booking System



Web

Mobile

Payment

Listing

Search

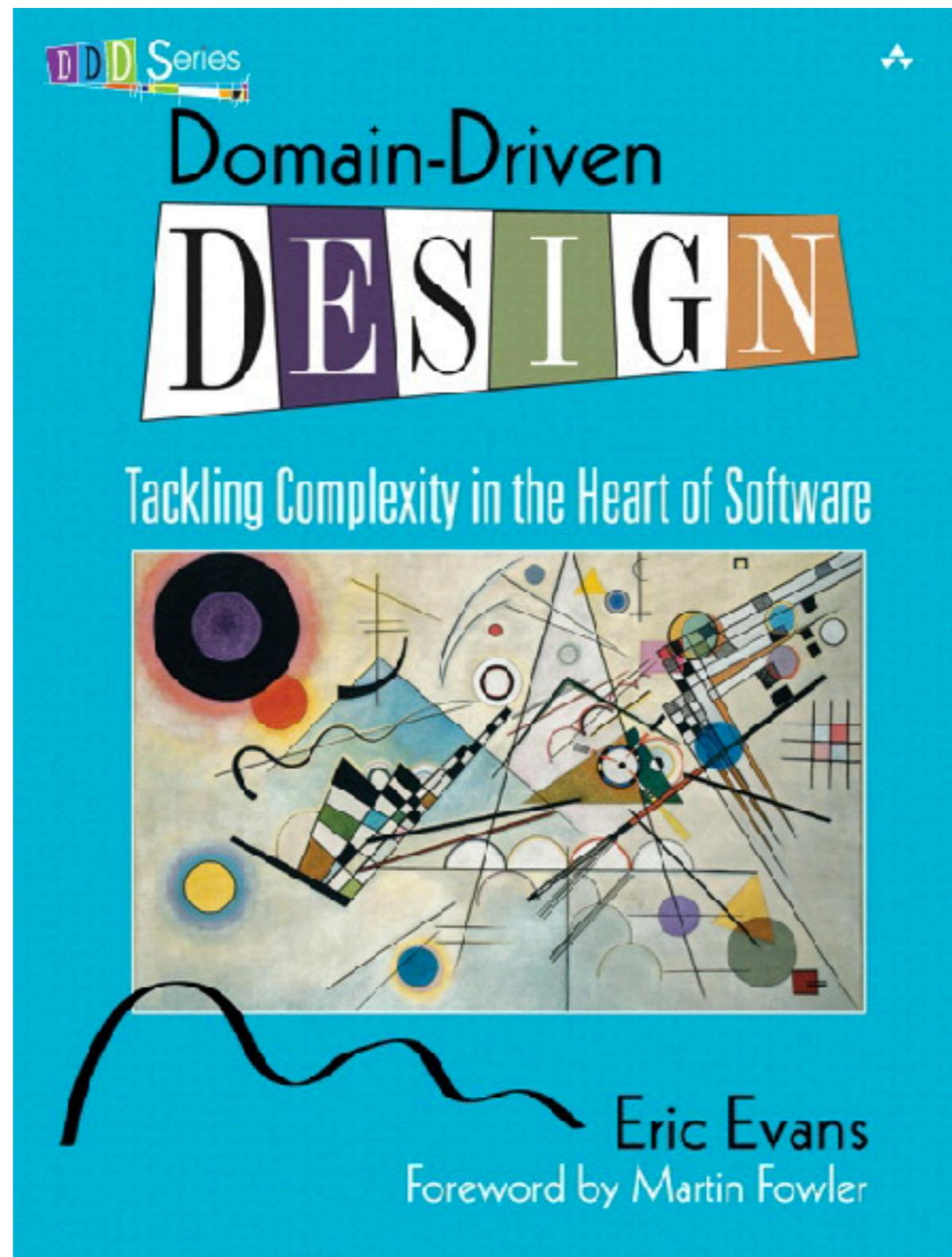
Booking

Marketing

Backoffice

Reporing





2. Individually deployable



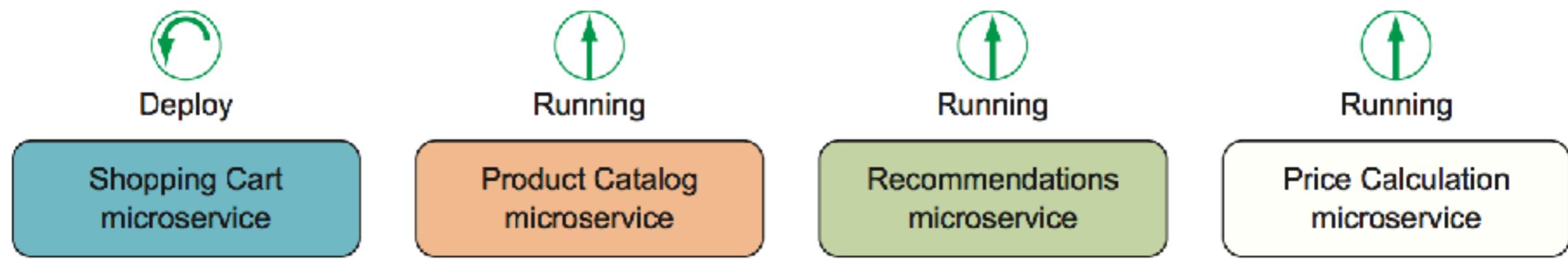


Figure 1.2 Other microservices continue to run while the Shopping Cart microservice is being deployed.



3. Consists of one or more processes



Problematic process boundary.
Microservices should run in separate processes to avoid coupling.

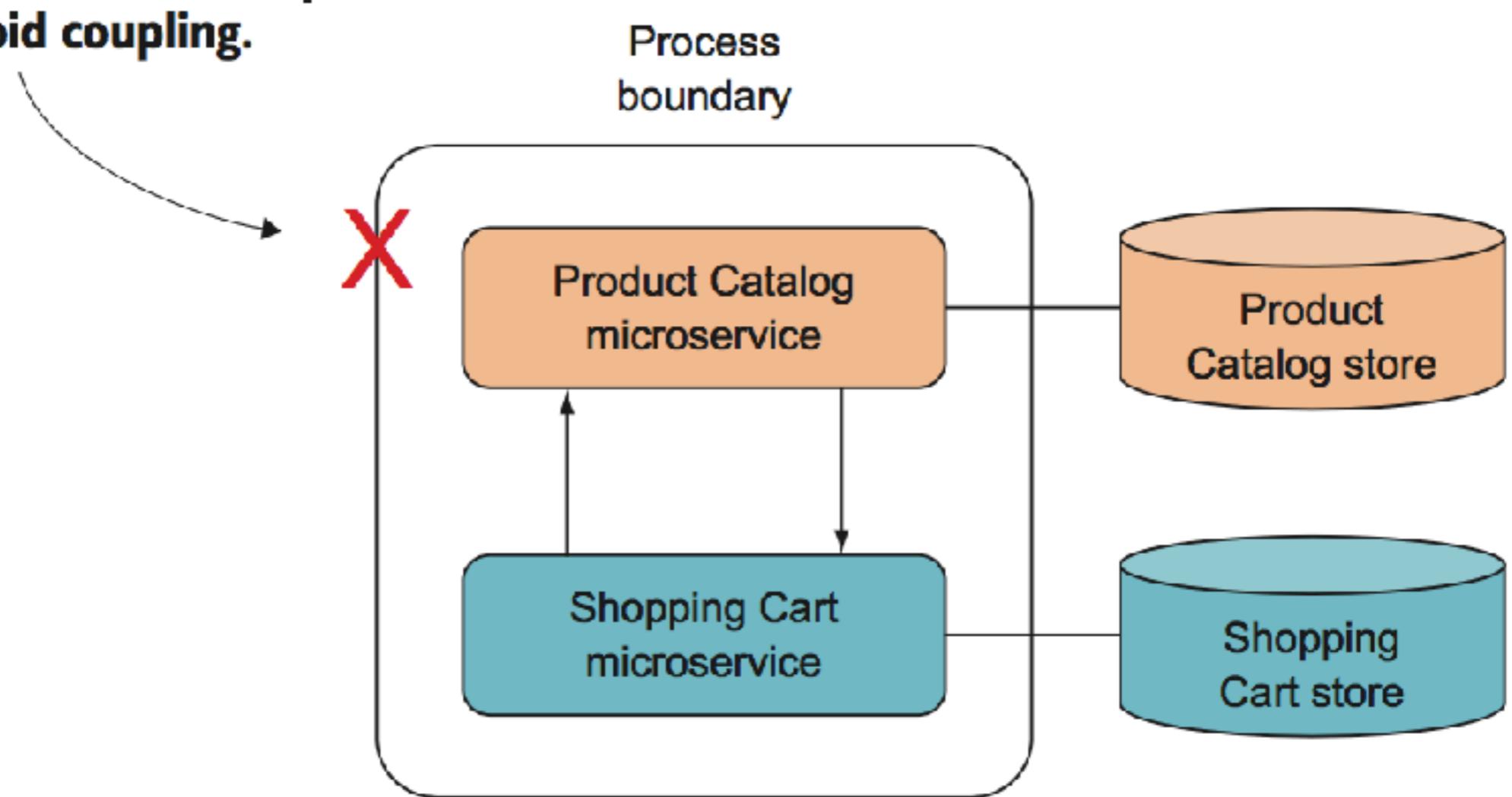


Figure 1.3 Running more than one microservice within a process leads to high coupling.



**Problematic process boundary.
Microservices should run in separate
processes to avoid coupling.**

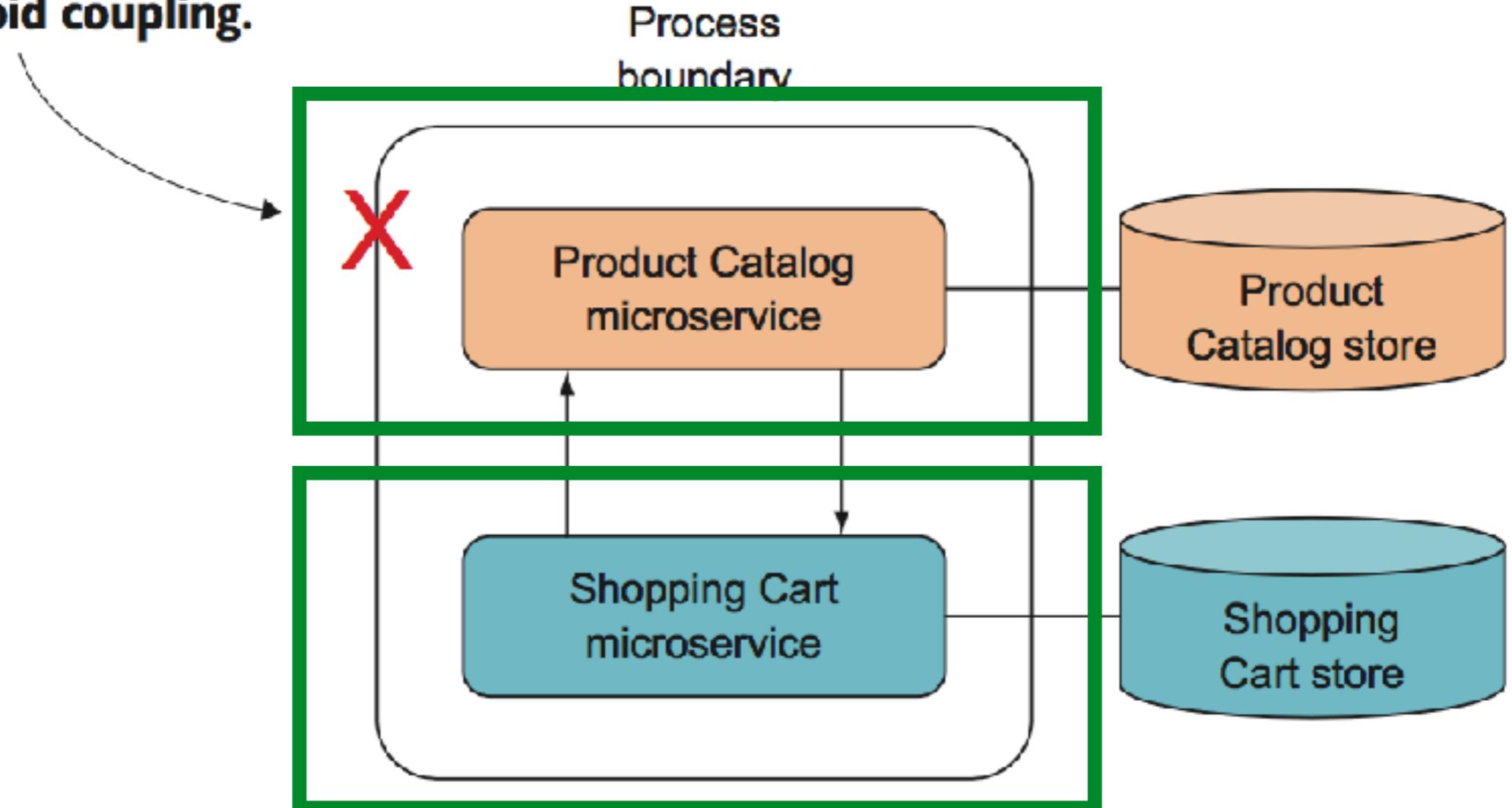


Figure 1.3 Running more than one microservice within a process leads to high coupling.



4. Own data store



All communication with the Product Catalog microservice must go through the public API.

Direct access to the Product Catalog store is not allowed. The Product Catalog microservice owns the Product Catalog store.

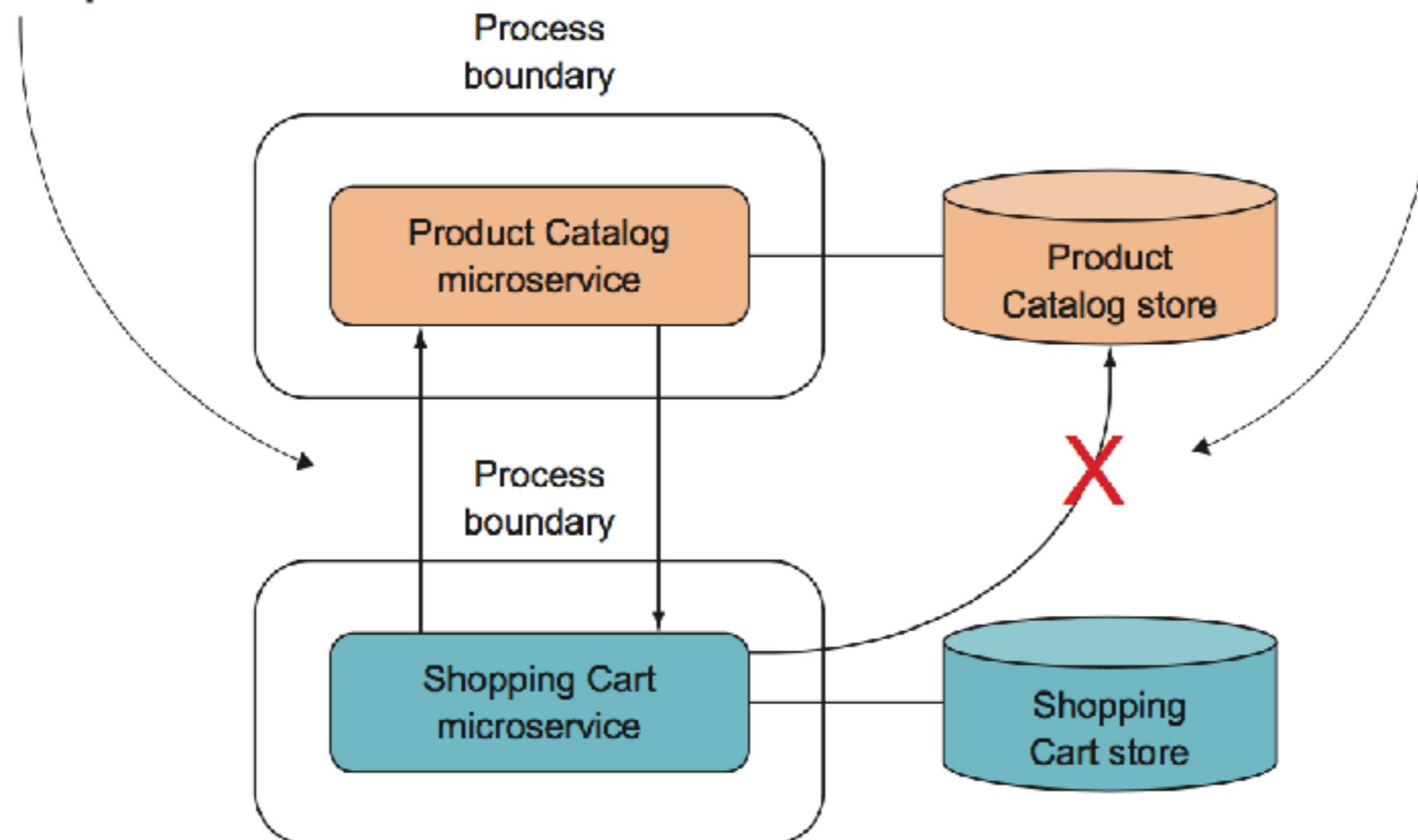


Figure 1.4 One microservice can't access another's data store.



All communication with the Product Catalog microservice must go through the public API.

Direct access to the Product Catalog store is not allowed. The Product Catalog microservice owns the Product Catalog store.

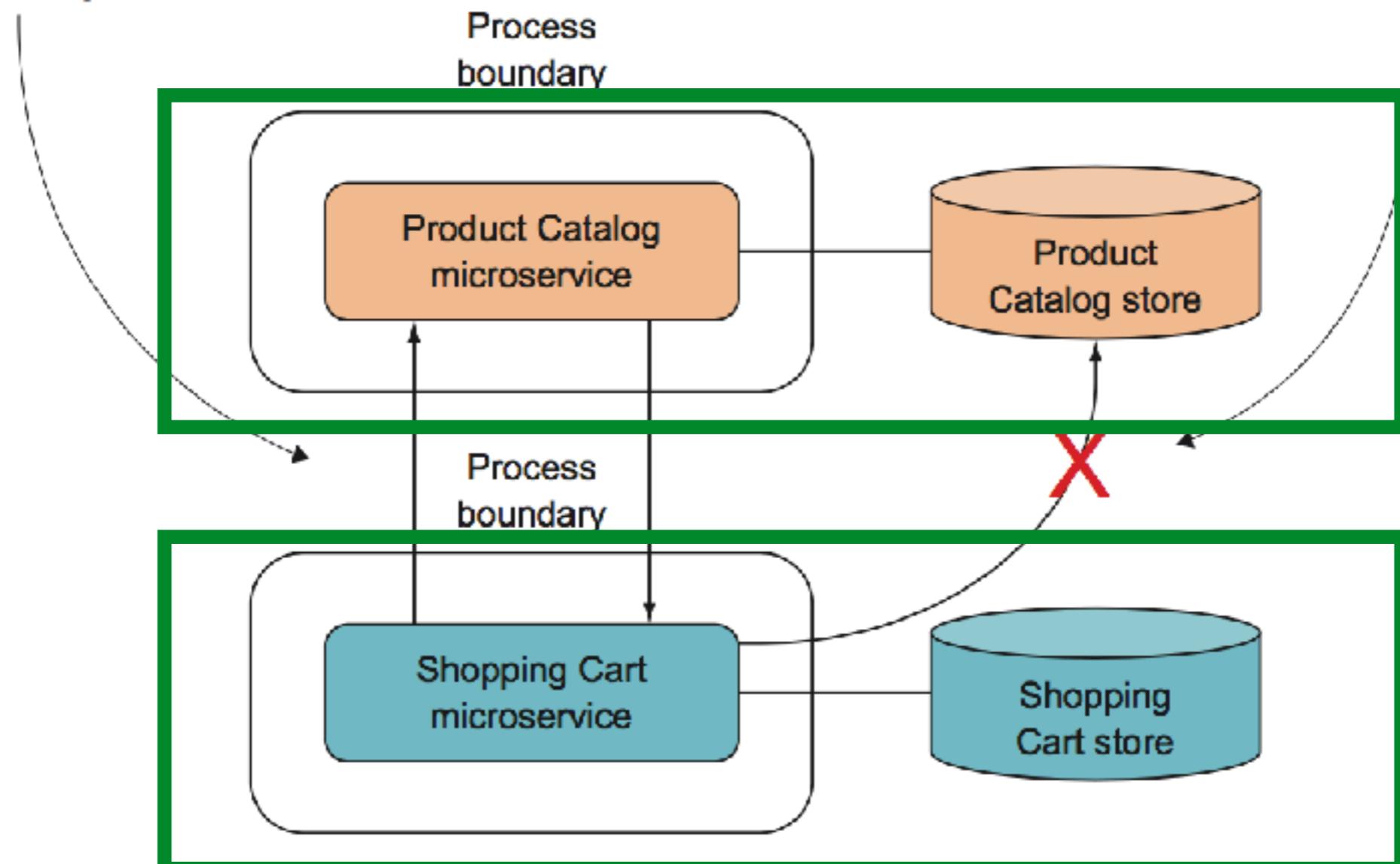
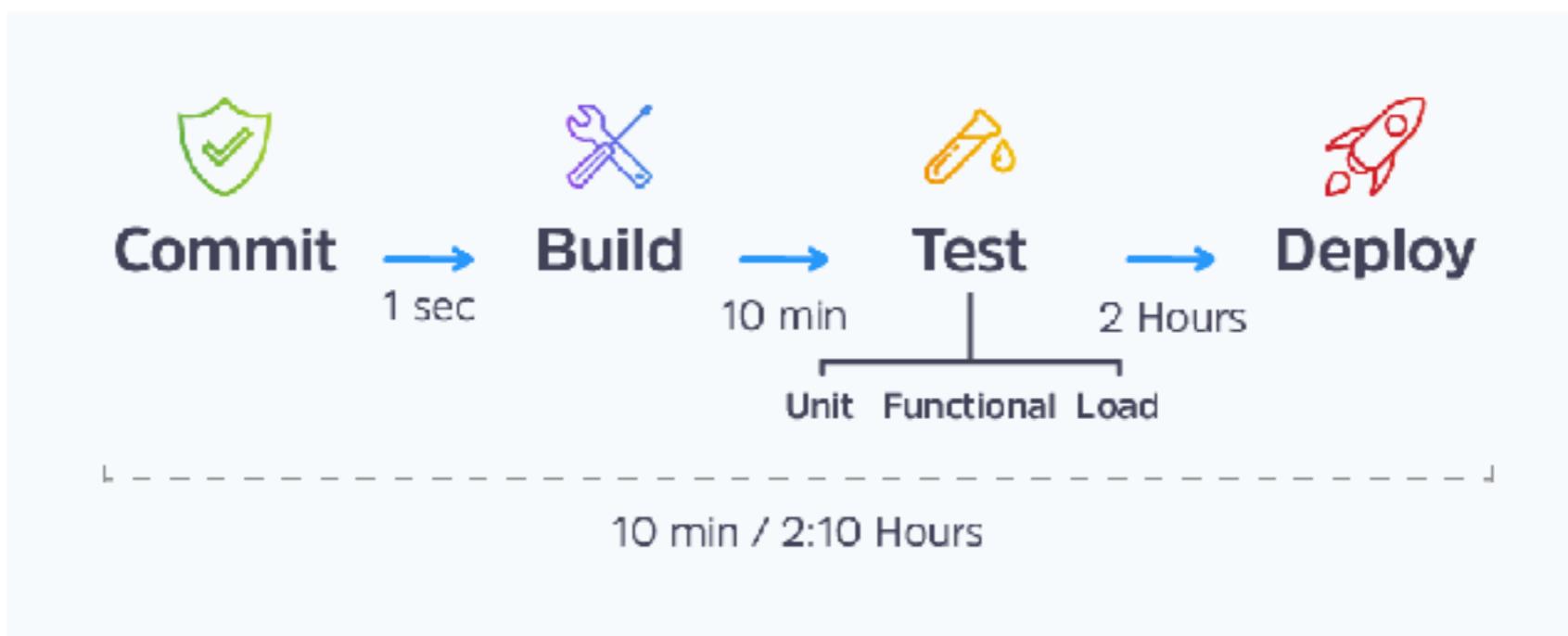


Figure 1.4 One microservice can't access another's data store.



5. Small team can maintain



6. Replaceable



Problems with Microservices ?



Challenges ~~Problems~~ with Microservices ?



1. How to define the boundaries of each microservices ?



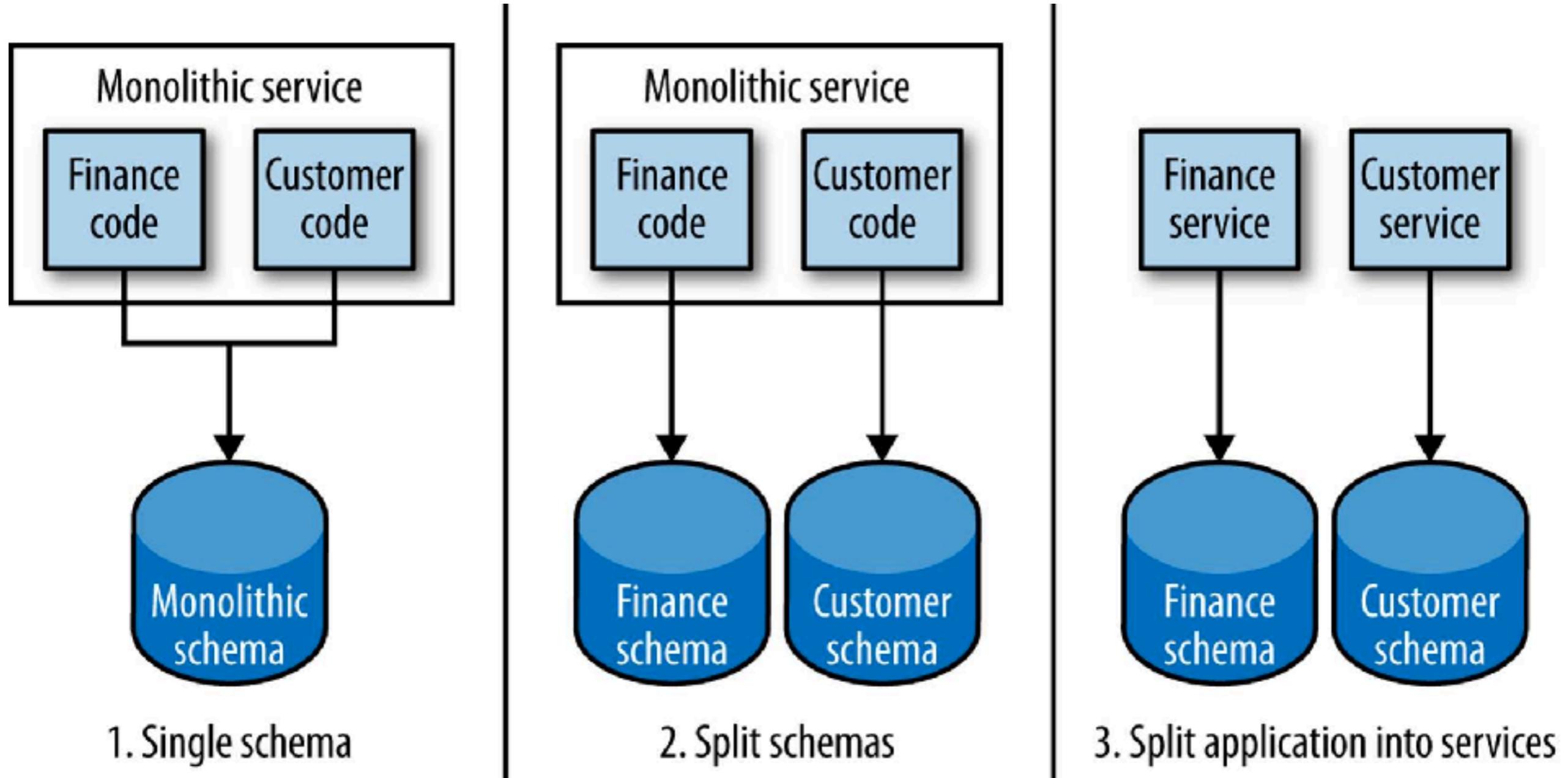
Premature splitting is the root of
all evil.



Every time you make the decision
to split out a new microservices,
there's a **risk** of ending up with a
bloated app.



Example



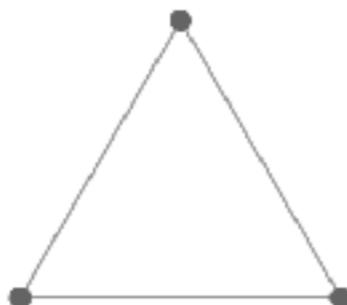
How Small should be Microservices ?



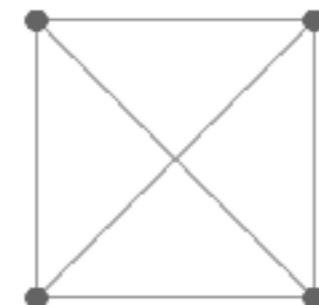
Two pizzas team



Effective communication



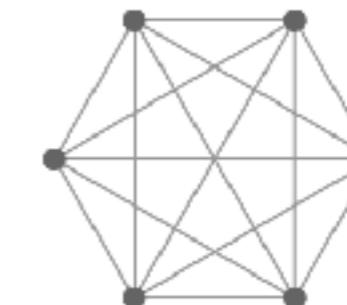
3 people, 3 lines



4 people, 6 lines



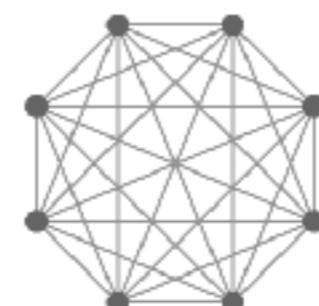
5 people, 10 lines



6 people, 15 lines



7 people, 21 lines



8 people, 28 lines



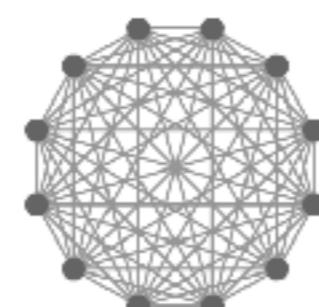
9 people, 36 lines



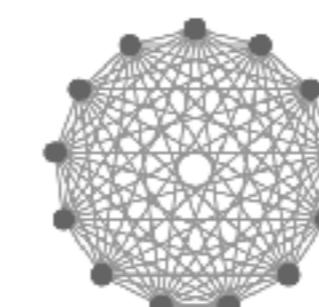
10 people, 45 lines



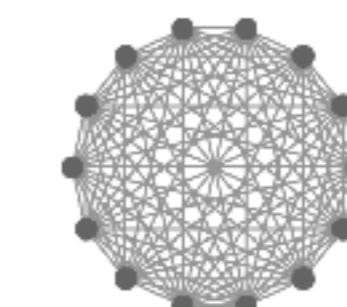
11 people, 55 lines



12 people, 66 lines



13 people, 78 lines



14 people, 91 lines



Single Responsibility Principle



Isolation from changes perspective

1

2

3



Isolation from business critical services

1

2

3



Team setup perspective

1

2

3



Technology perspective

1

2

3



Data consistency and transaction boundary perspective

1

2

3



Non-functional requirement perspective

1

2

3



Try by yourself



2. How to create queries that retrieve data from several Microservices ?



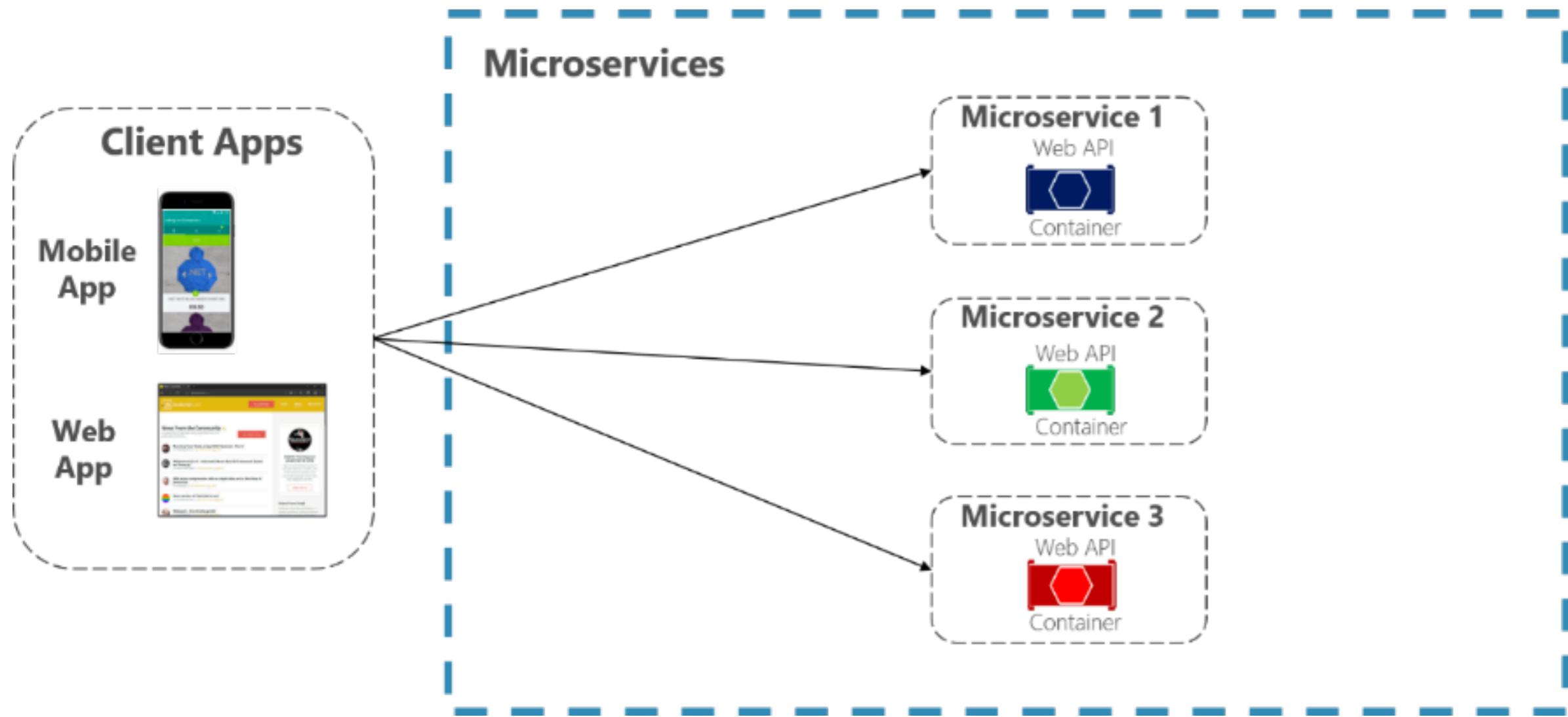
Popular solutions

API Gateway
CQRS with query/read tables
Cold data in centralize database

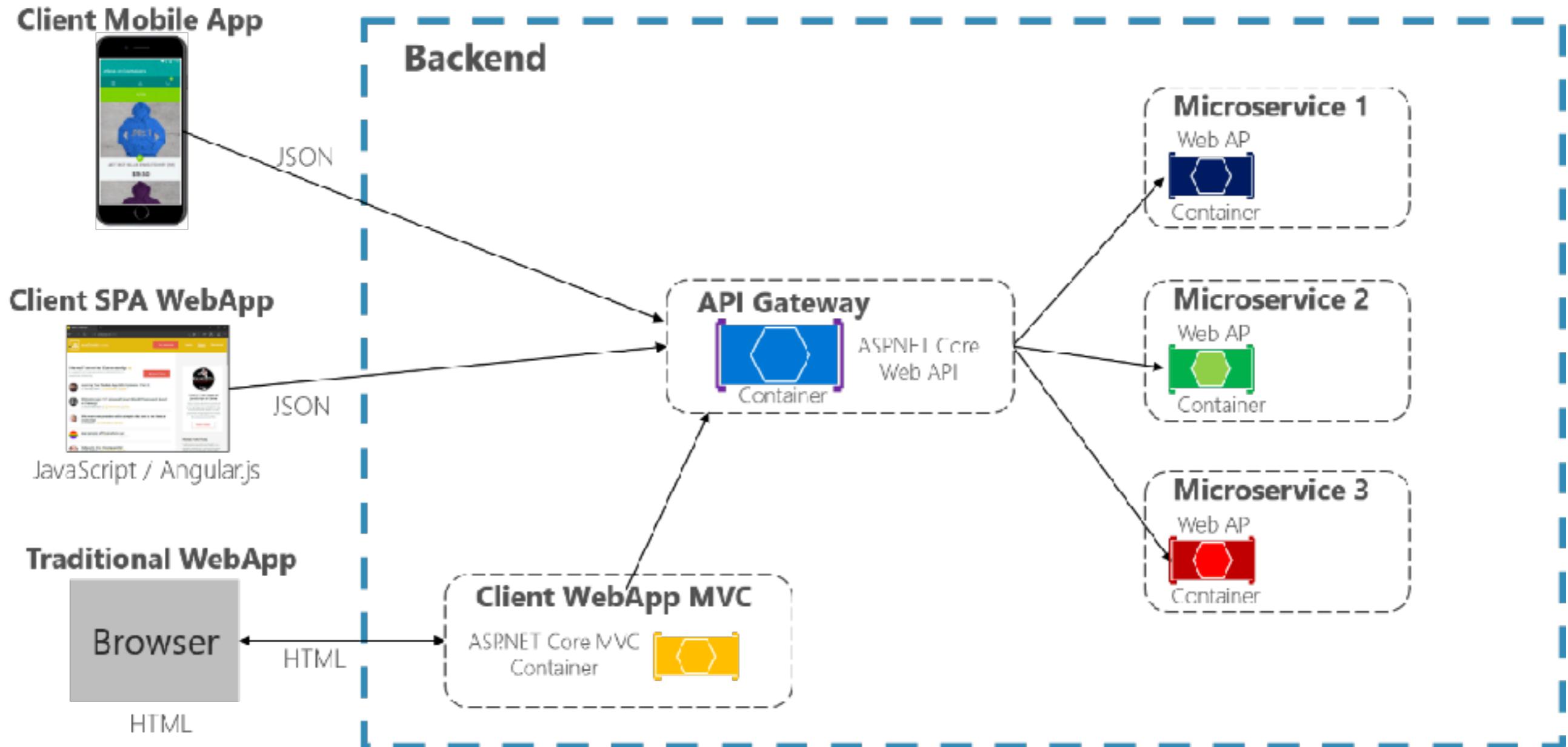


Direct Client-To-Microservice communication

Architecture



Using the API Gateway Service



Queries ⇒ database (type)

POST
PUT
DELETE

GET /customers/id GET /orders?text=xyz GET ...

Aggregate

Command side

MongoDB

Query side

ElasticSearch

Query side

Neo4j

Query side

Events

Event Store

@crichtson



3. How to achieve consistency across multiple Microservices ?



Ordering microservice

Ordering API



ID	Quantity	ProductID

OrderItems Table
in Ordering-DB
(Remote SQL)

Catalog microservice

Catalog.API



ID	Stock	Name

Products Table
in Catalog-DB
(Remote SQL)

Don't

Databases are private per microservice

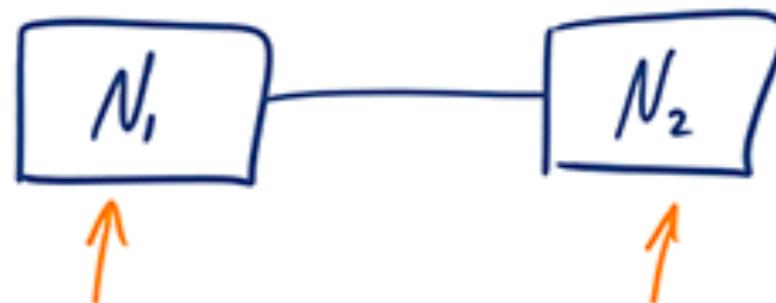


CAP Theorem

Consistency



Availability



Partition Tolerance



<http://robertgreiner.com/2014/08/cap-theorem-revisited/>



4. How to design communication across microservices boundaries ?



Protocols

HTTP and REST Messaging



Communication

Request-Response model
Observer model

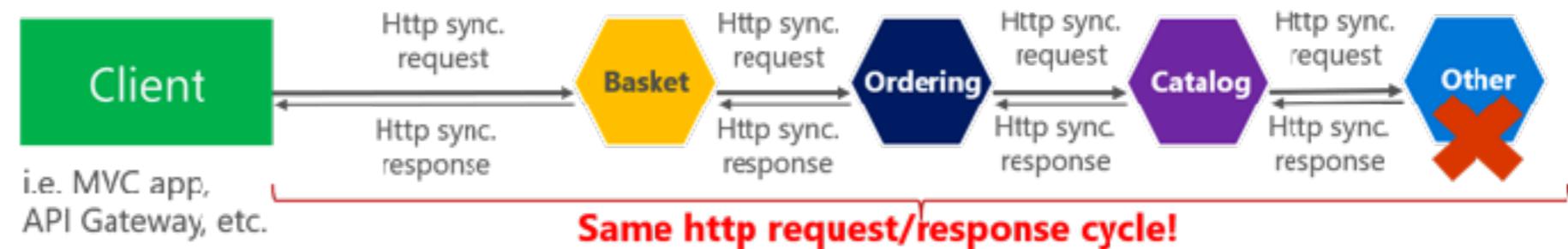


Communication

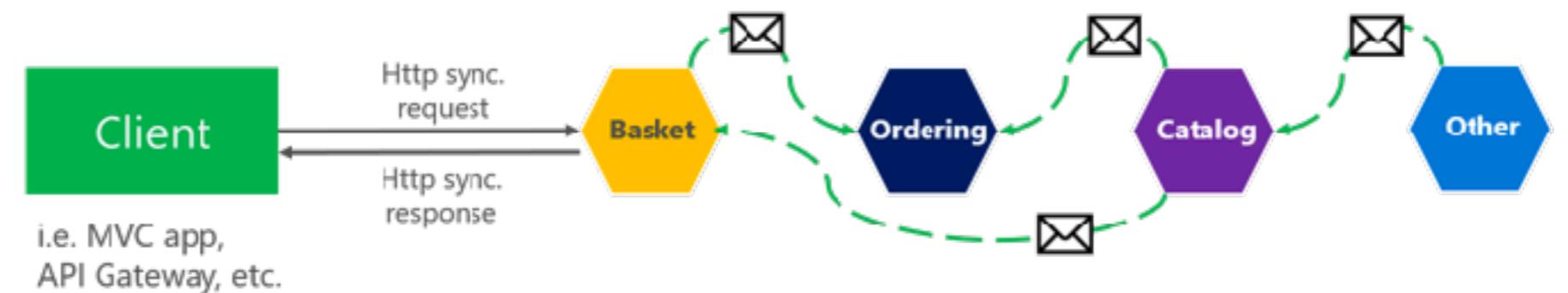
Synchronous vs. async communication across microservices

Anti-pattern

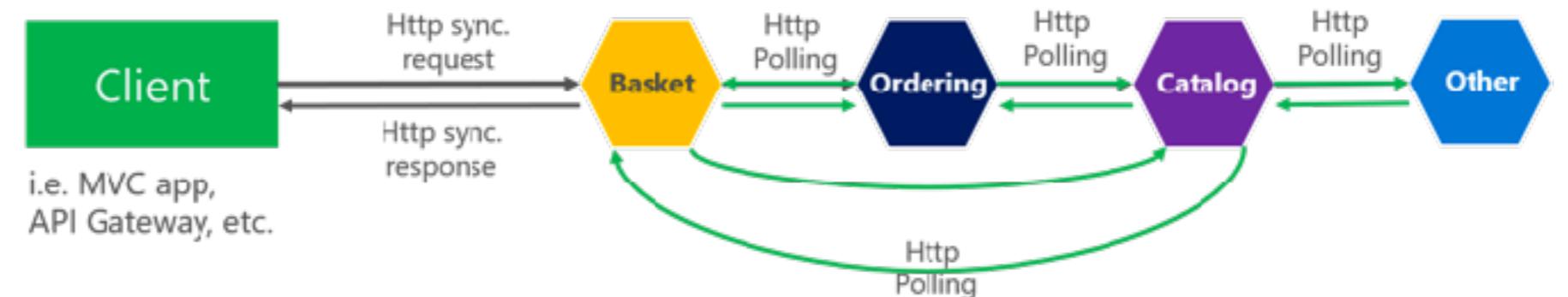
Synchronous
all req./resp. cycle



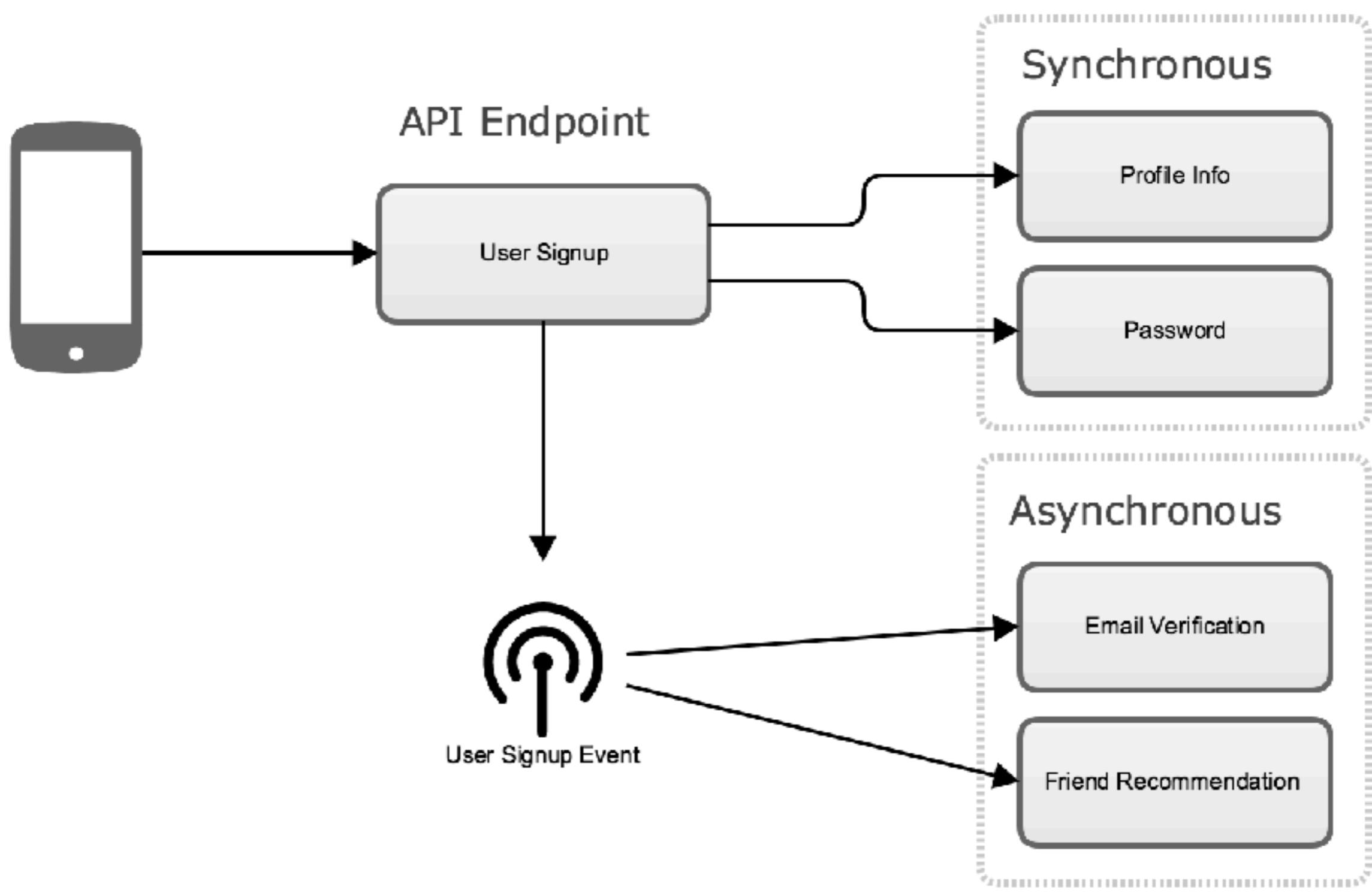
Asynchronous
Comm. across
internal microservices
(EventBus: i.e. **AMQP**)



"Asynchronous"
Comm. across
internal microservices
(Polling: **Http**)

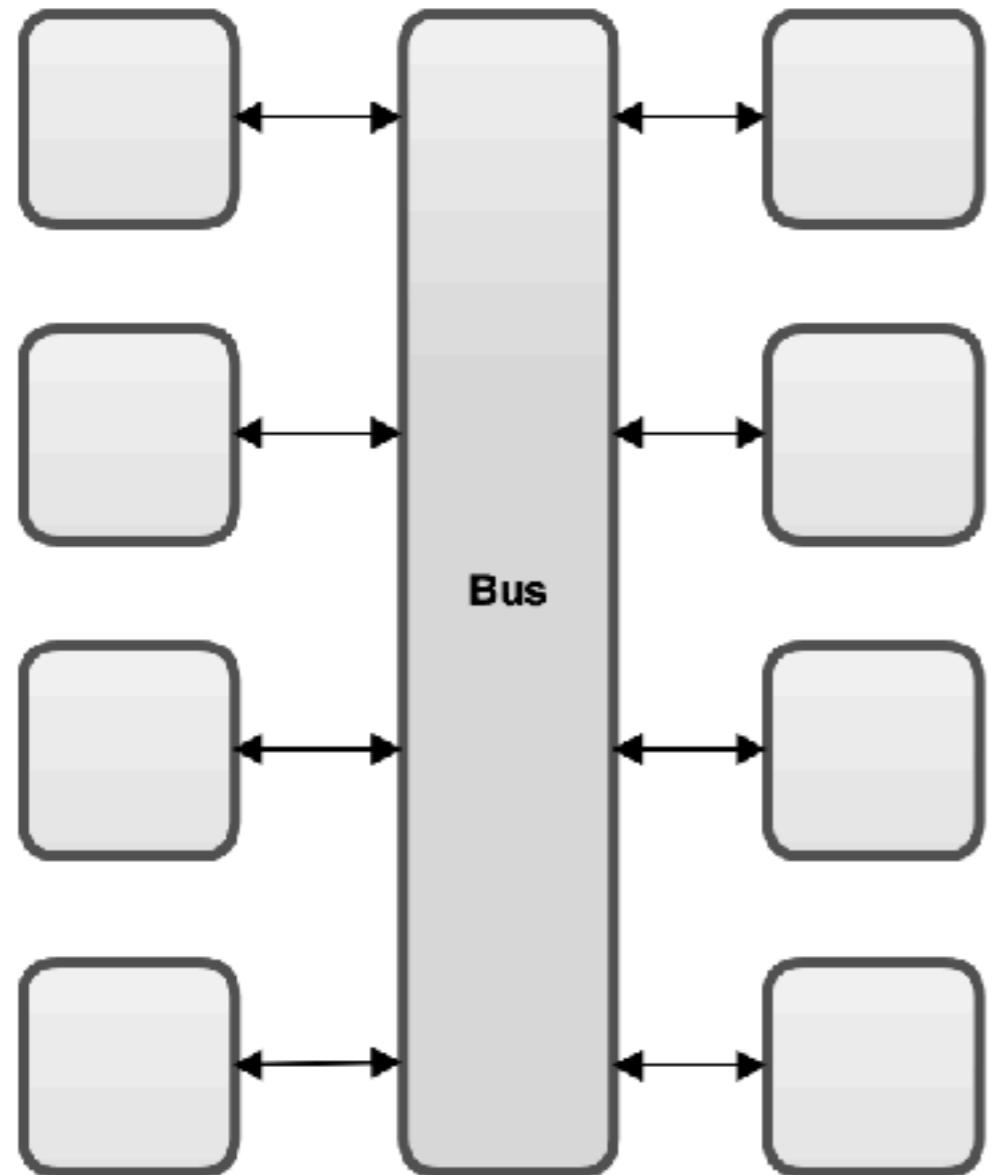


Communication

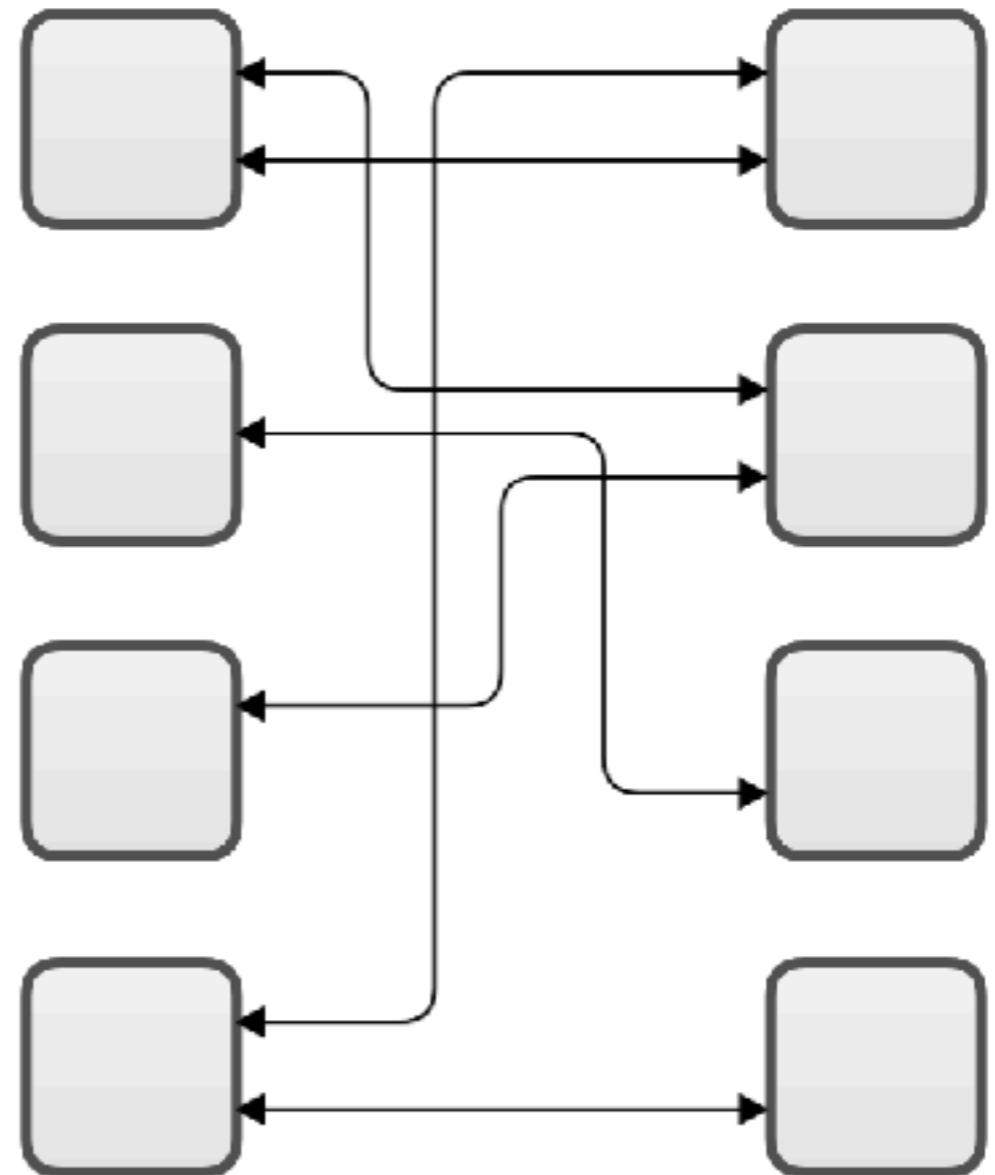


Anti-pattern :: centralize bus service

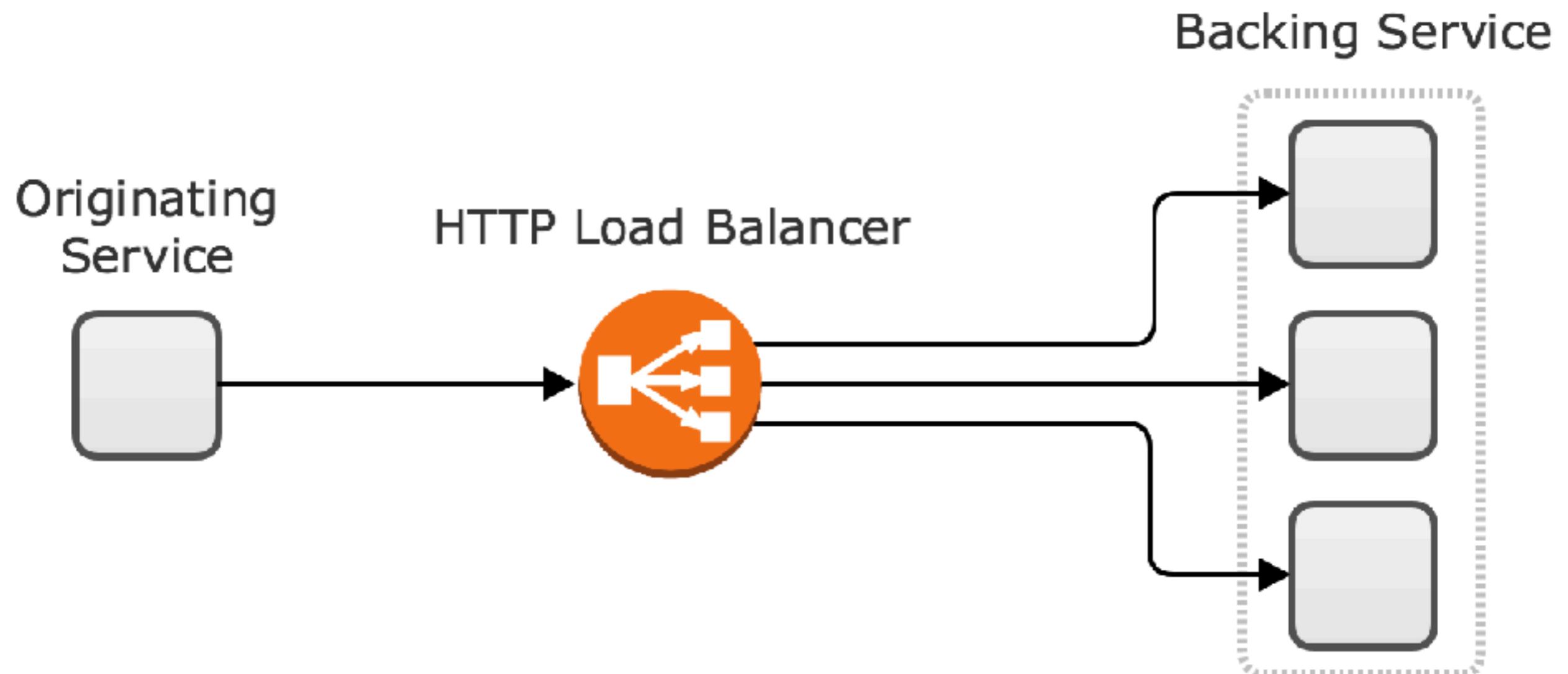
Central Bus



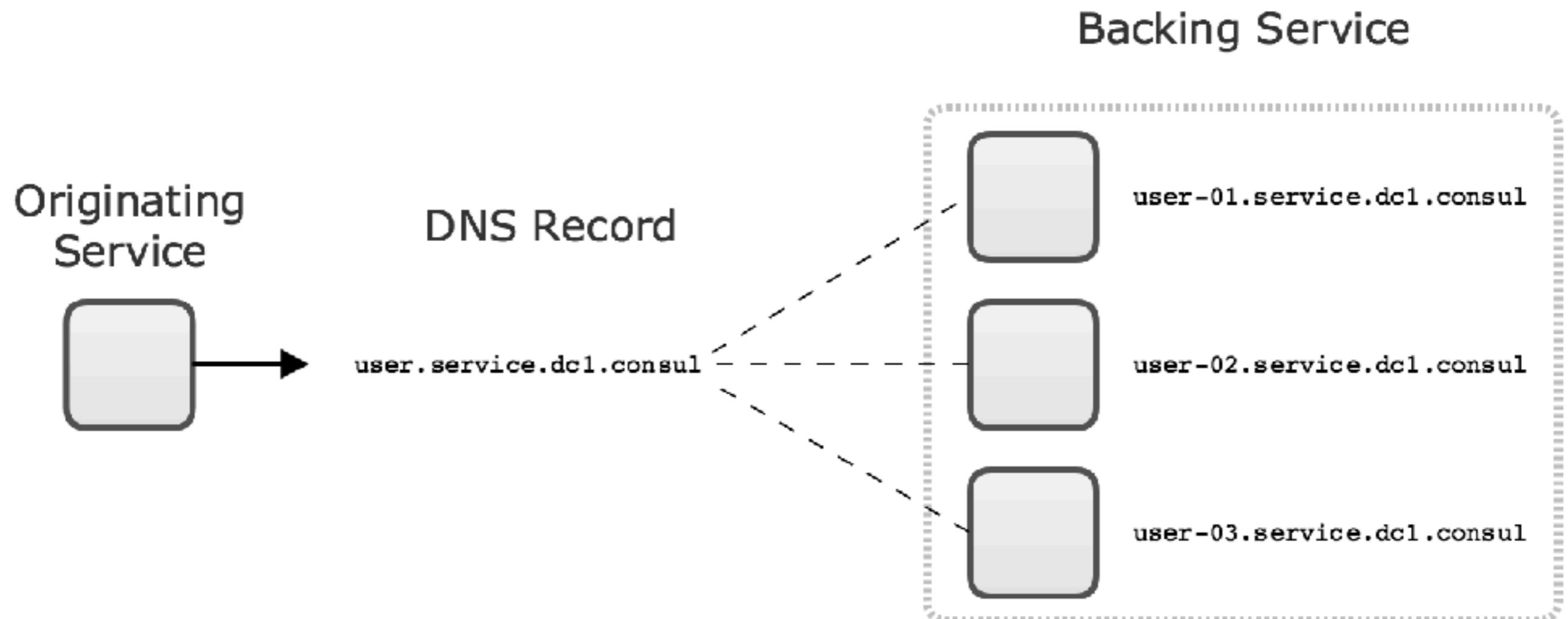
Decentralized



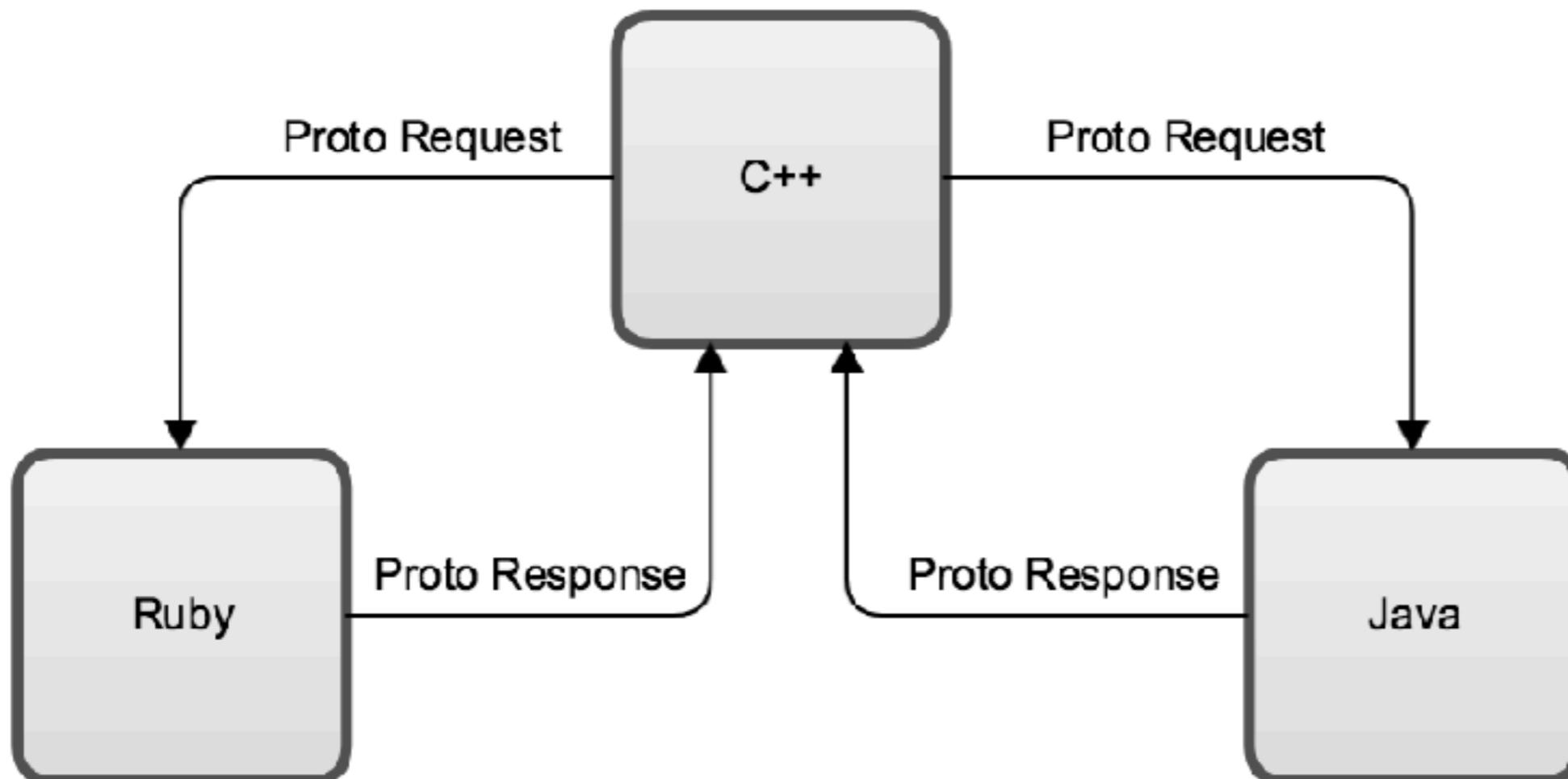
Request-response model



Request-response model



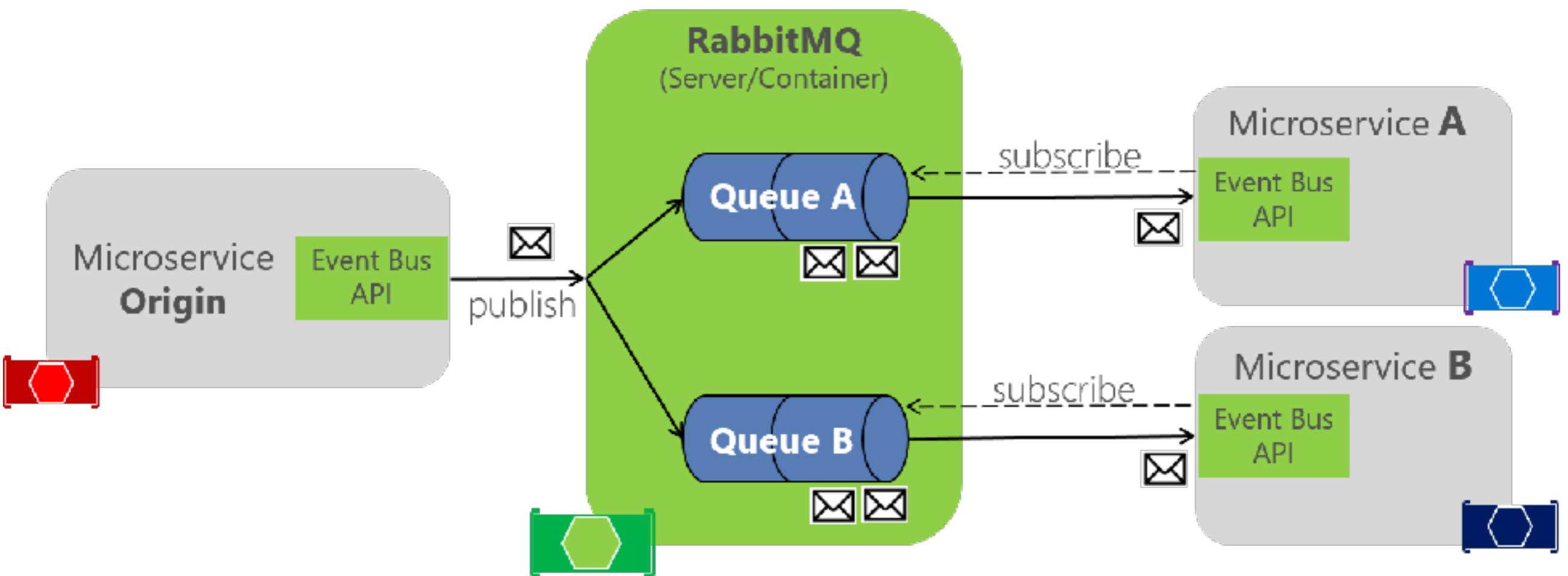
Request-response model



Observer model

**Message
Sender**

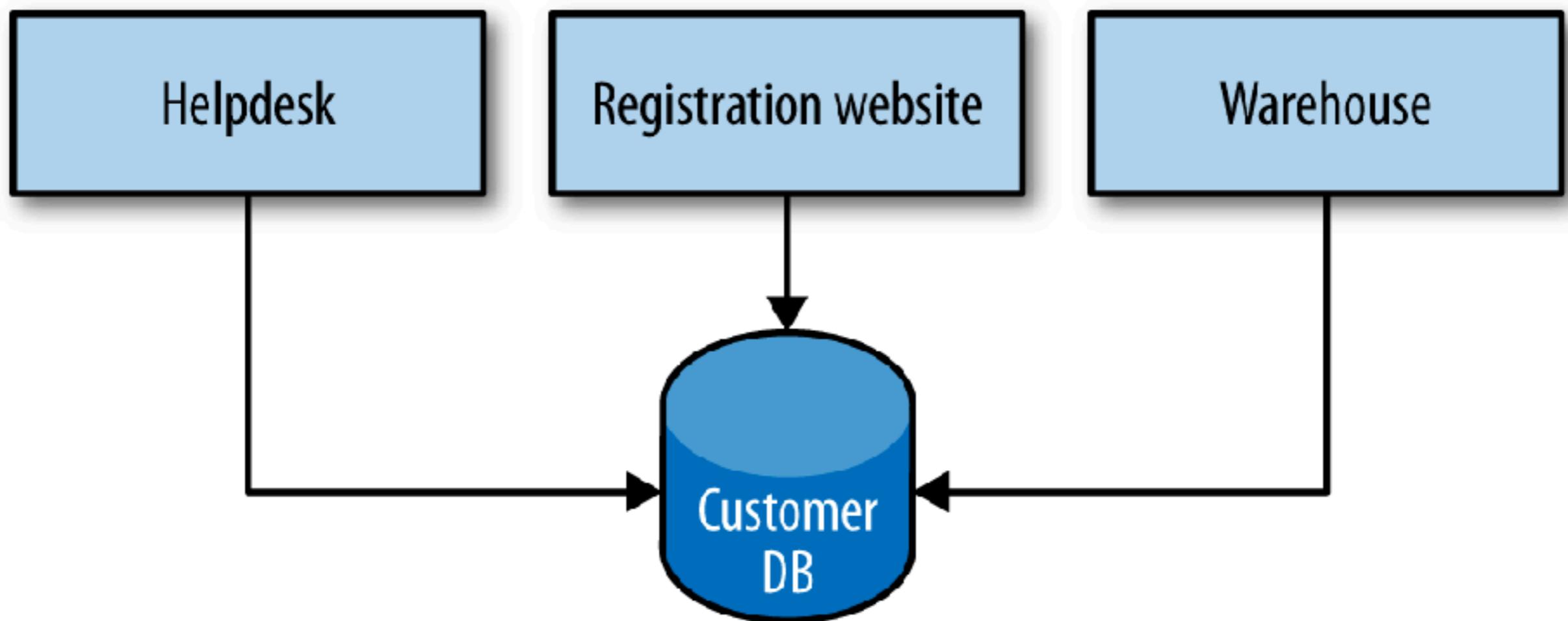
**Message
Receivers**



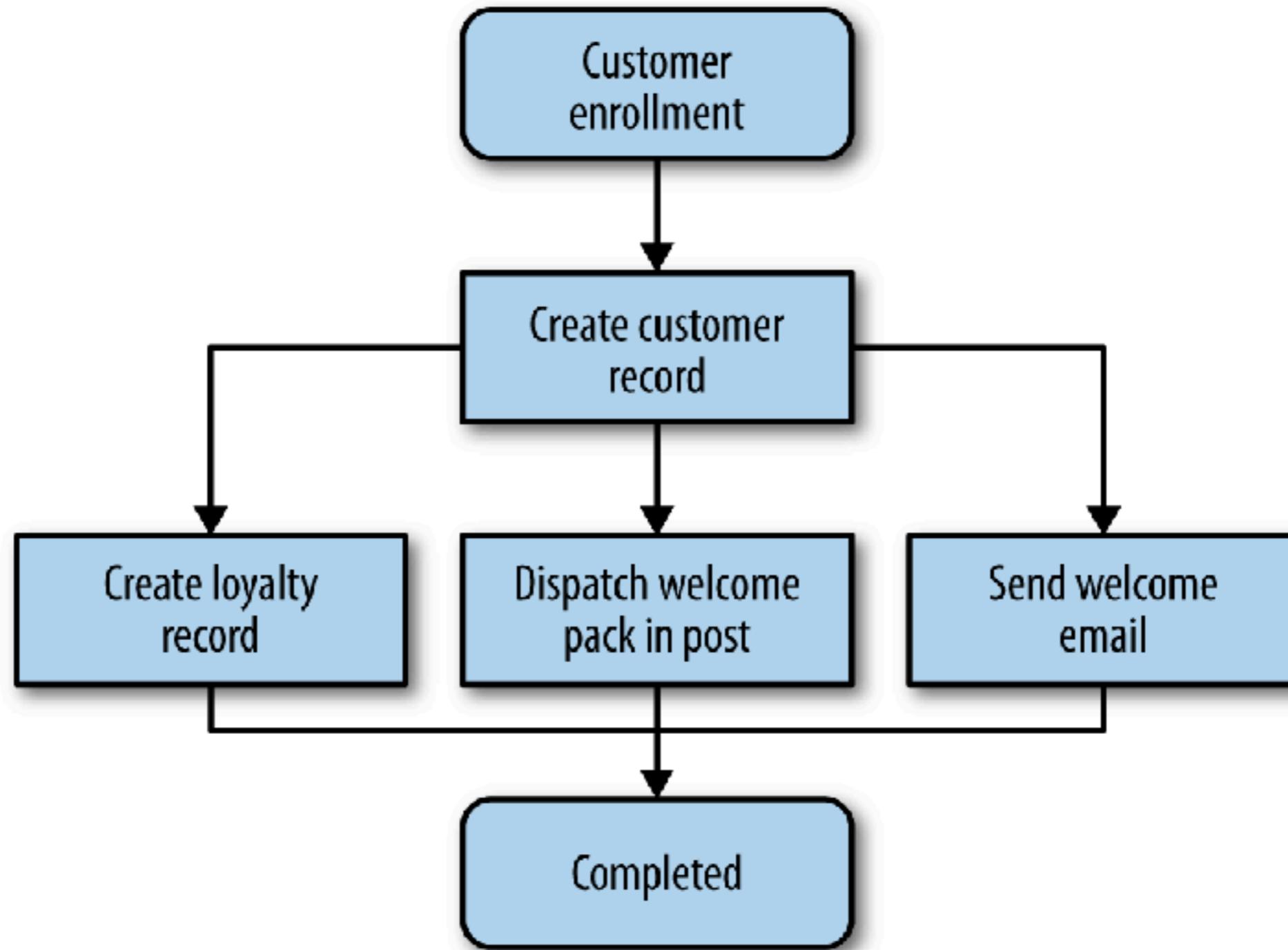
Services Integration



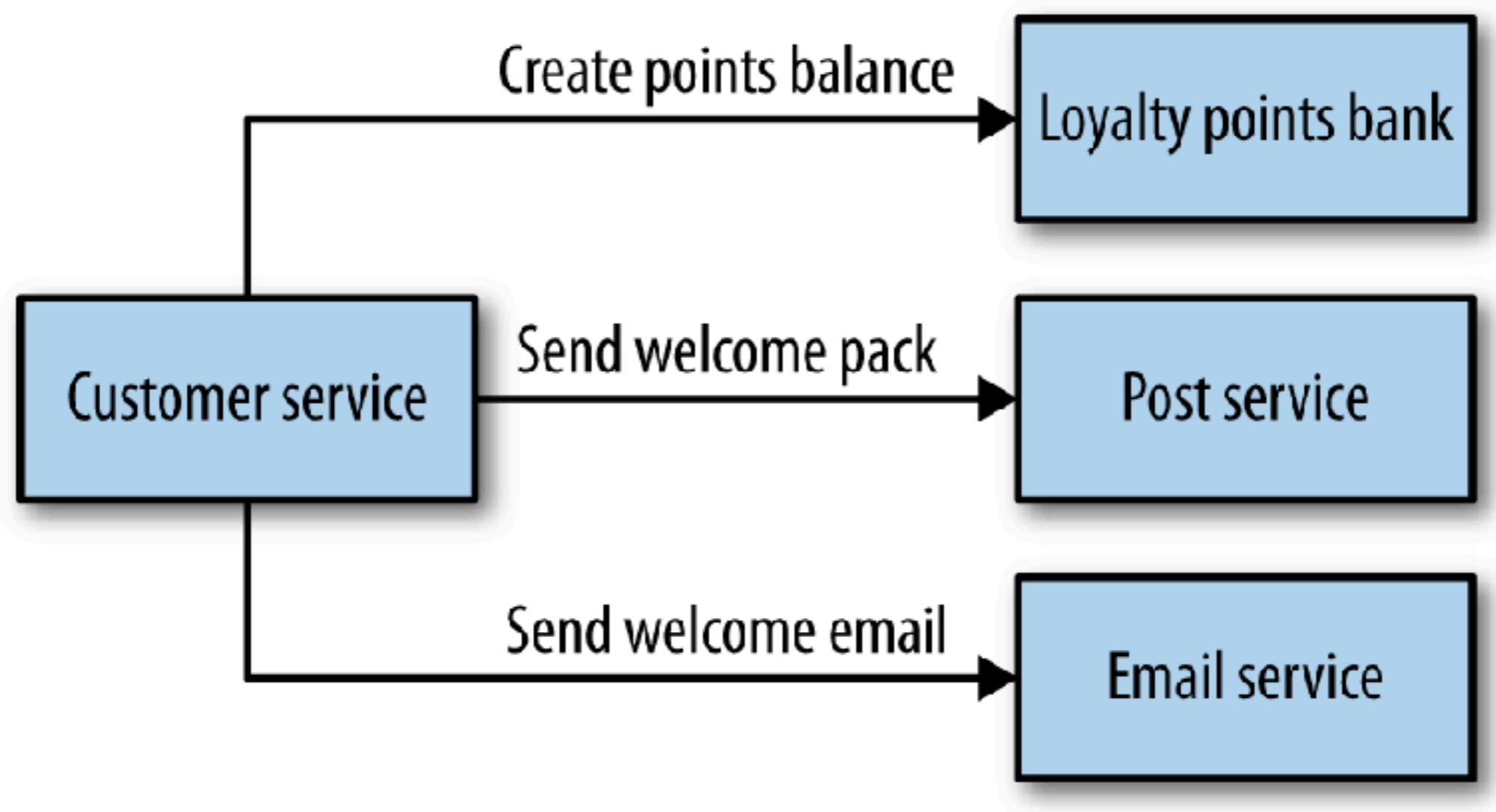
Shared database



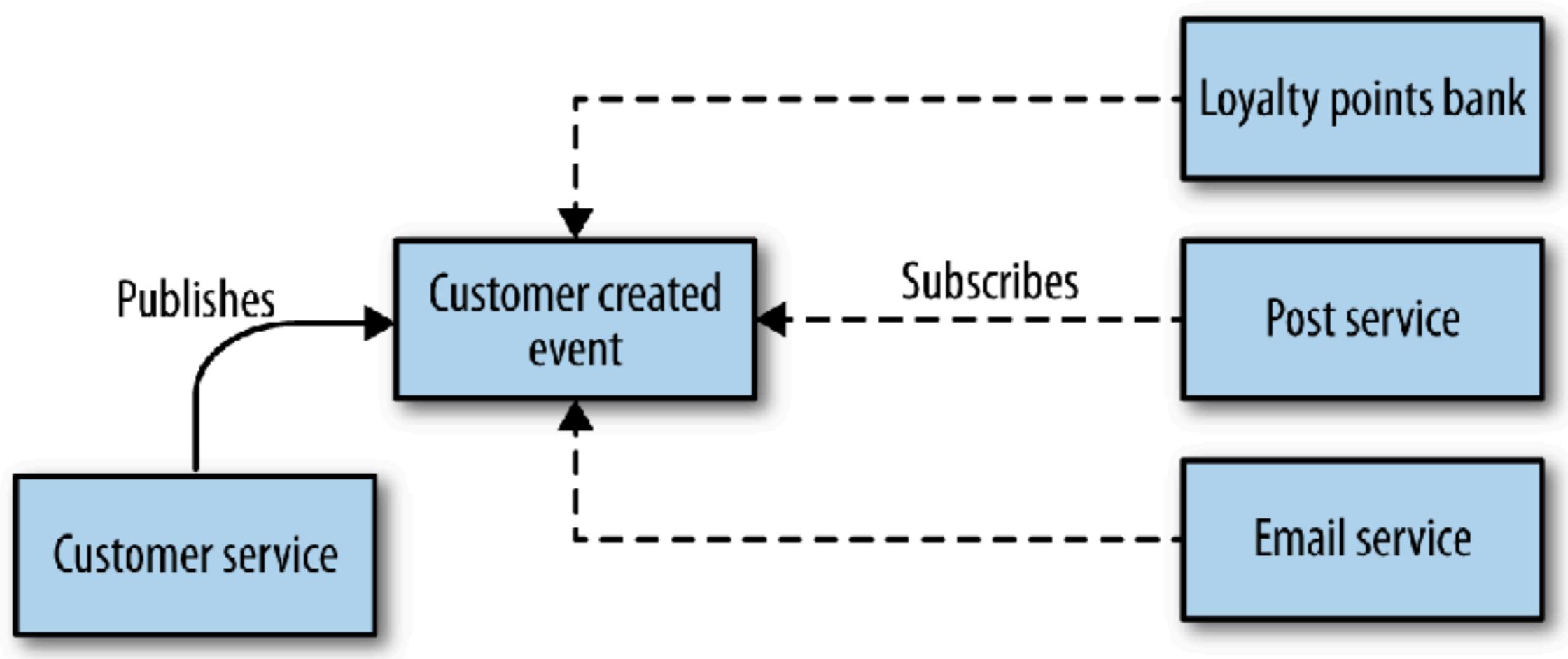
Orchestration vs Choreography



Orchestration



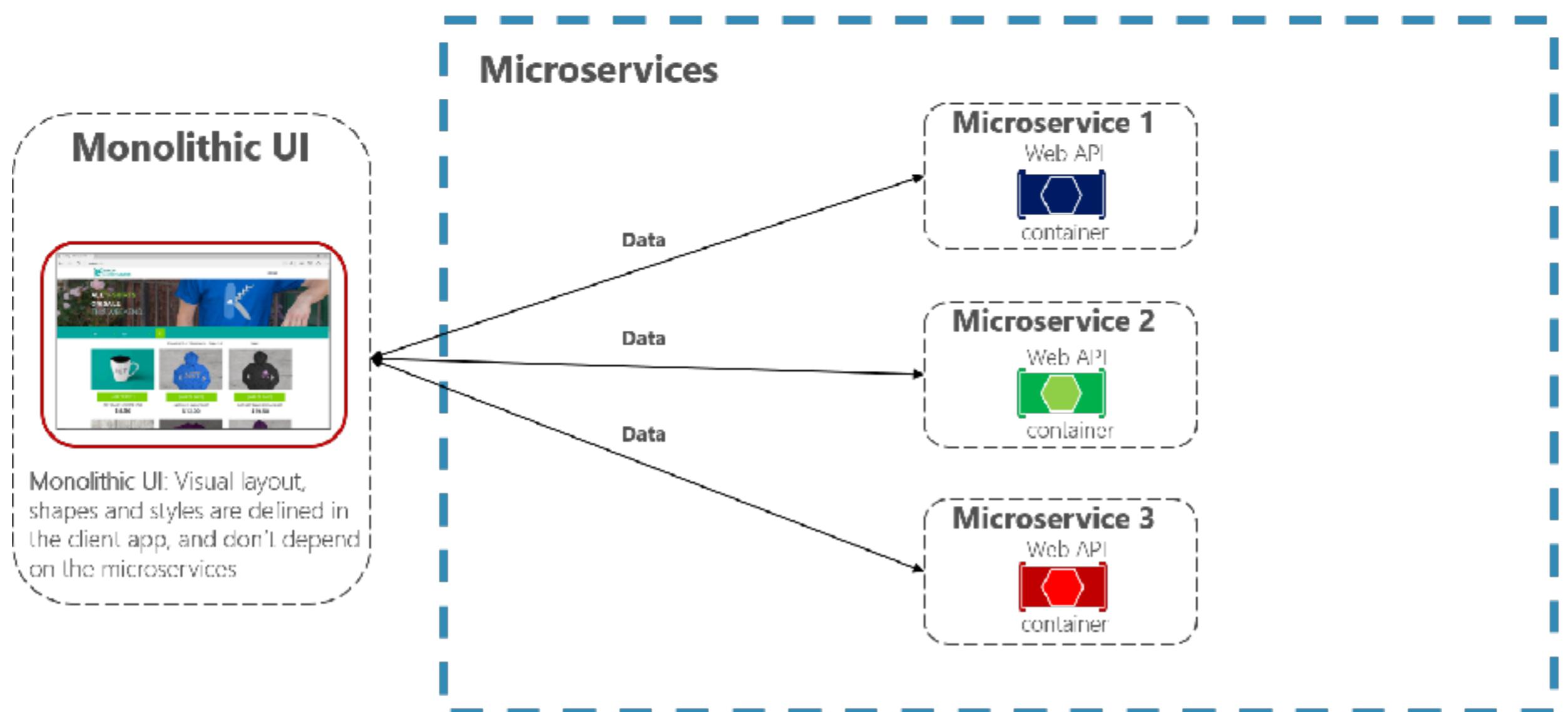
Choreography



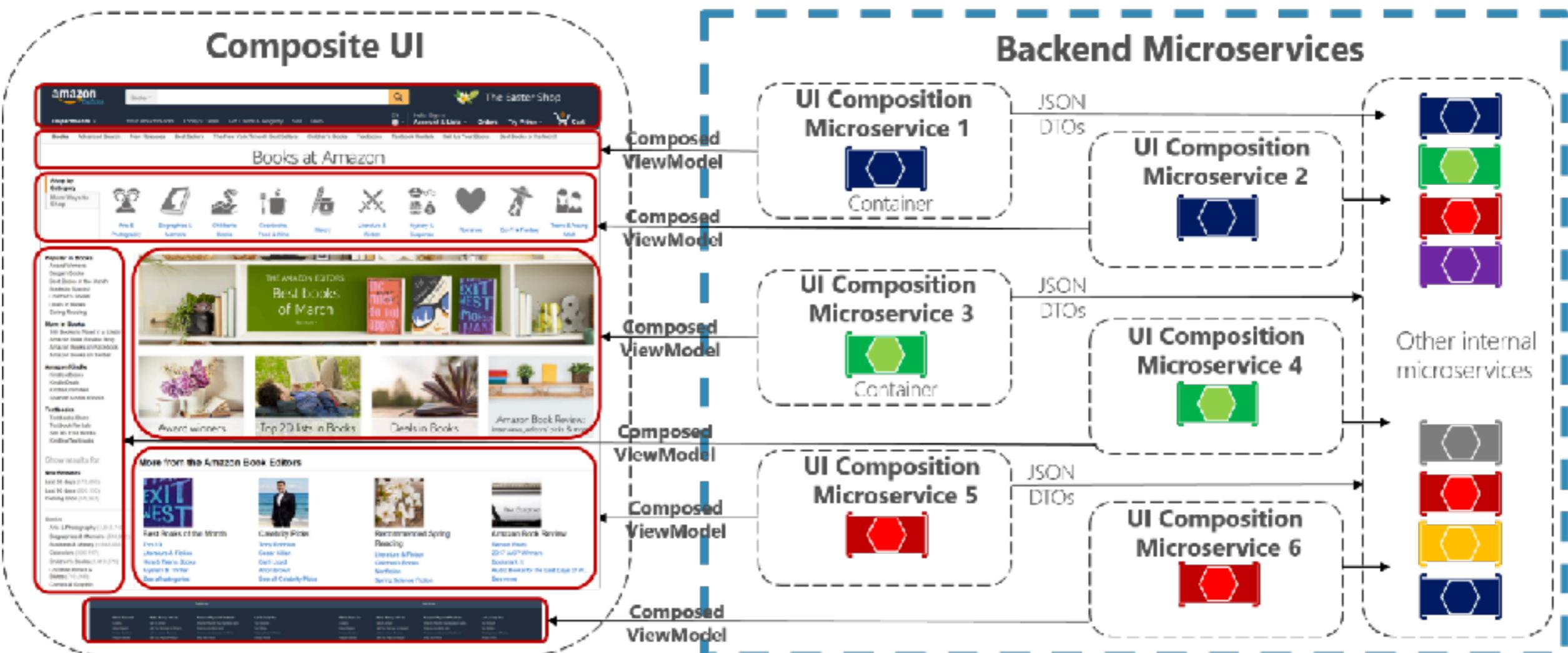
Integrate with User Interface



Monolithic UI consuming microservices



Composite UI generated by microservices



Micro Frontend

<https://micro-frontends.org/>

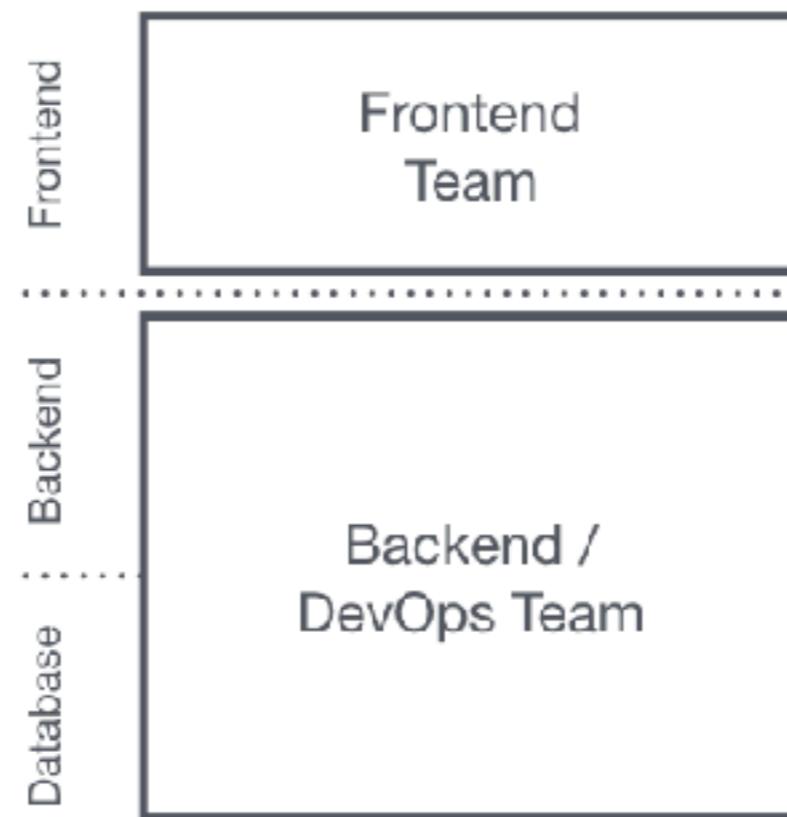


Monolith Frontend

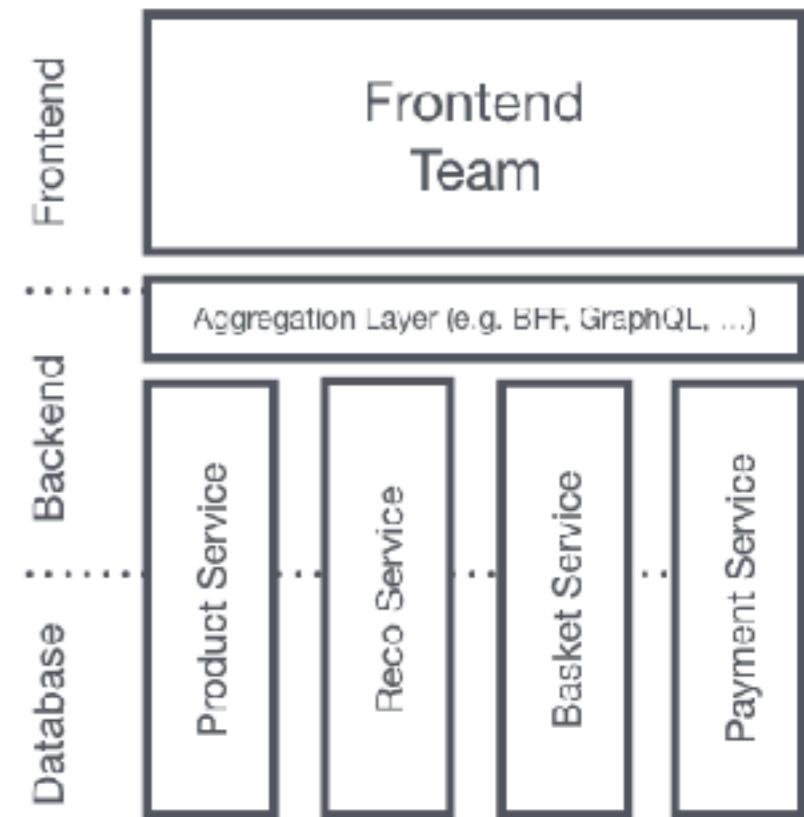
The Monolith



Front & Back

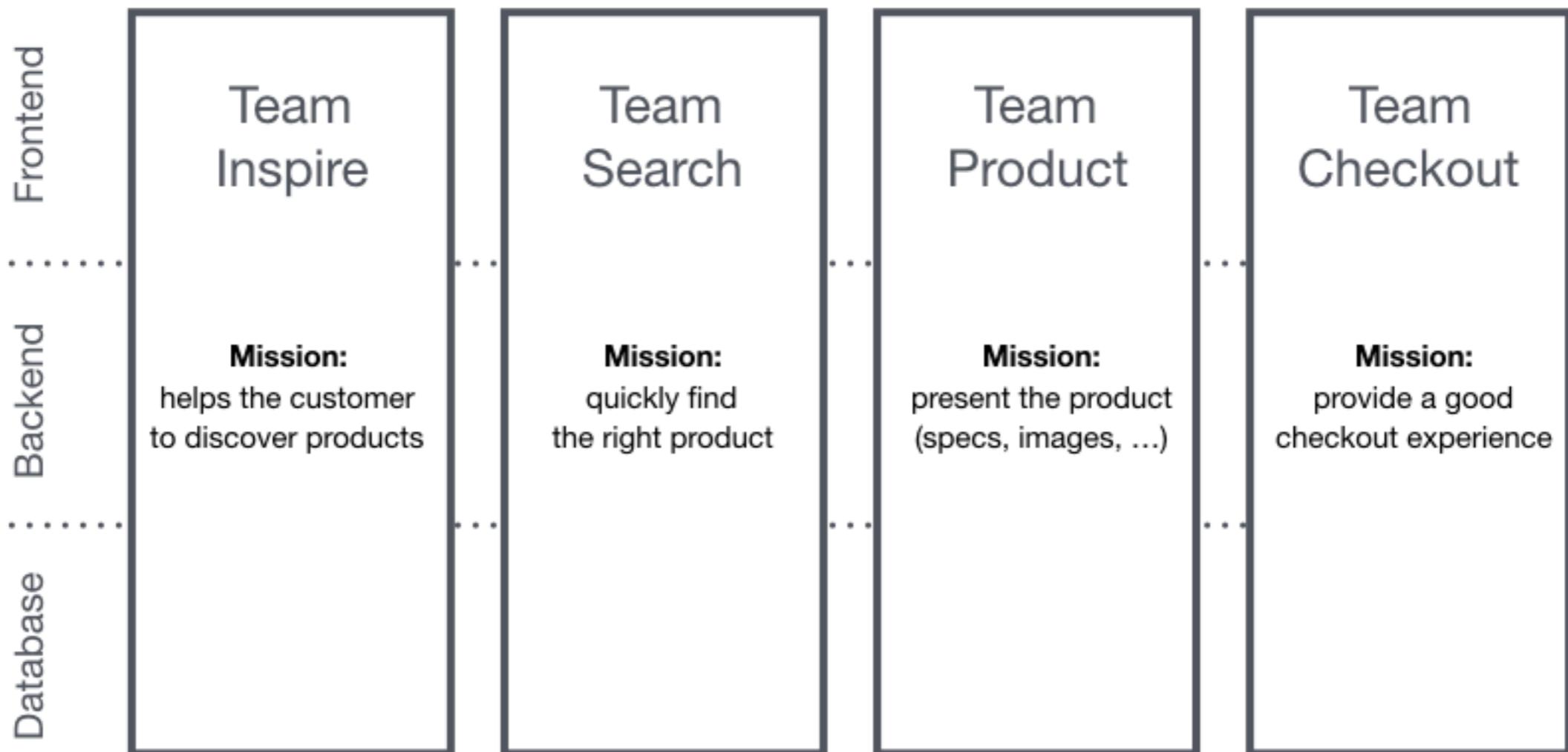


Microservices



Micro Frontend

End-to-End Teams with Micro Frontends



Microservices pitfalls



Microservices pitfalls

More/Low splitting

More network interaction

Data storing and sharing

Compatibility issues

Testing

Operation & Monitoring

Security

Performance

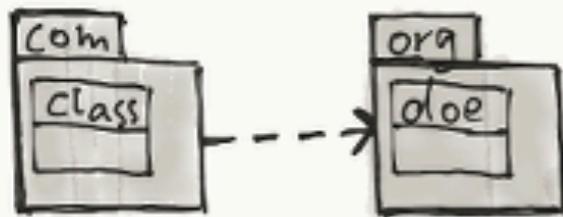




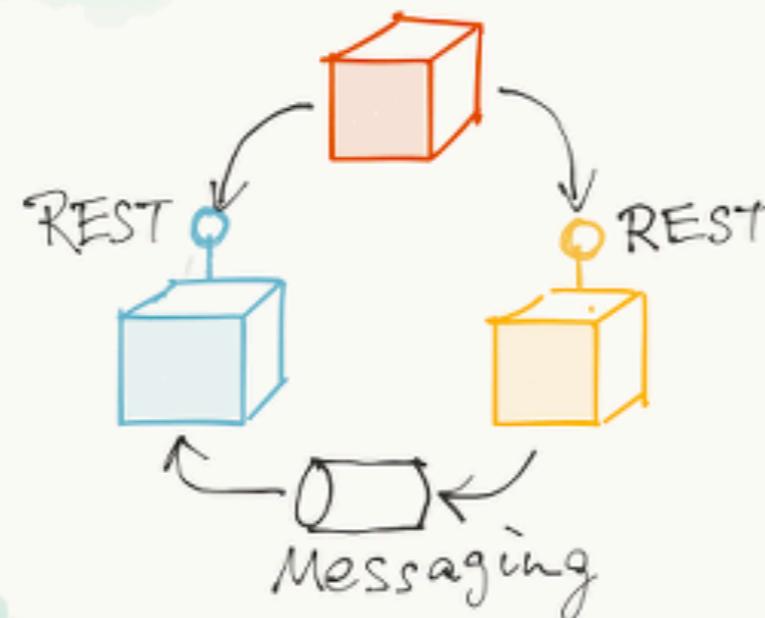
Microservices in one page



Architecture



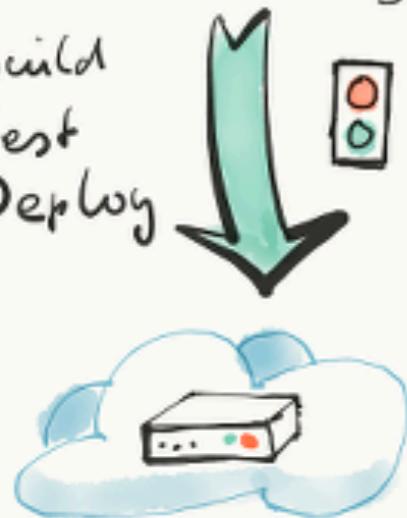
Microservices



Deployment

Continuous Delivery

`{ var i=1; }`
Build
Test
Deploy



Infrastructure

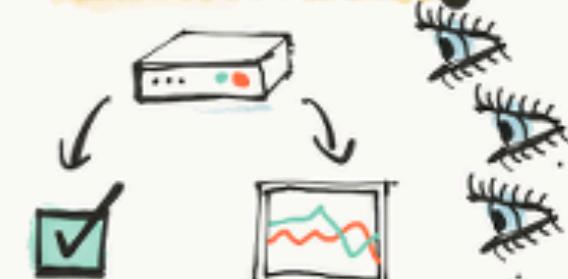


People & Teams



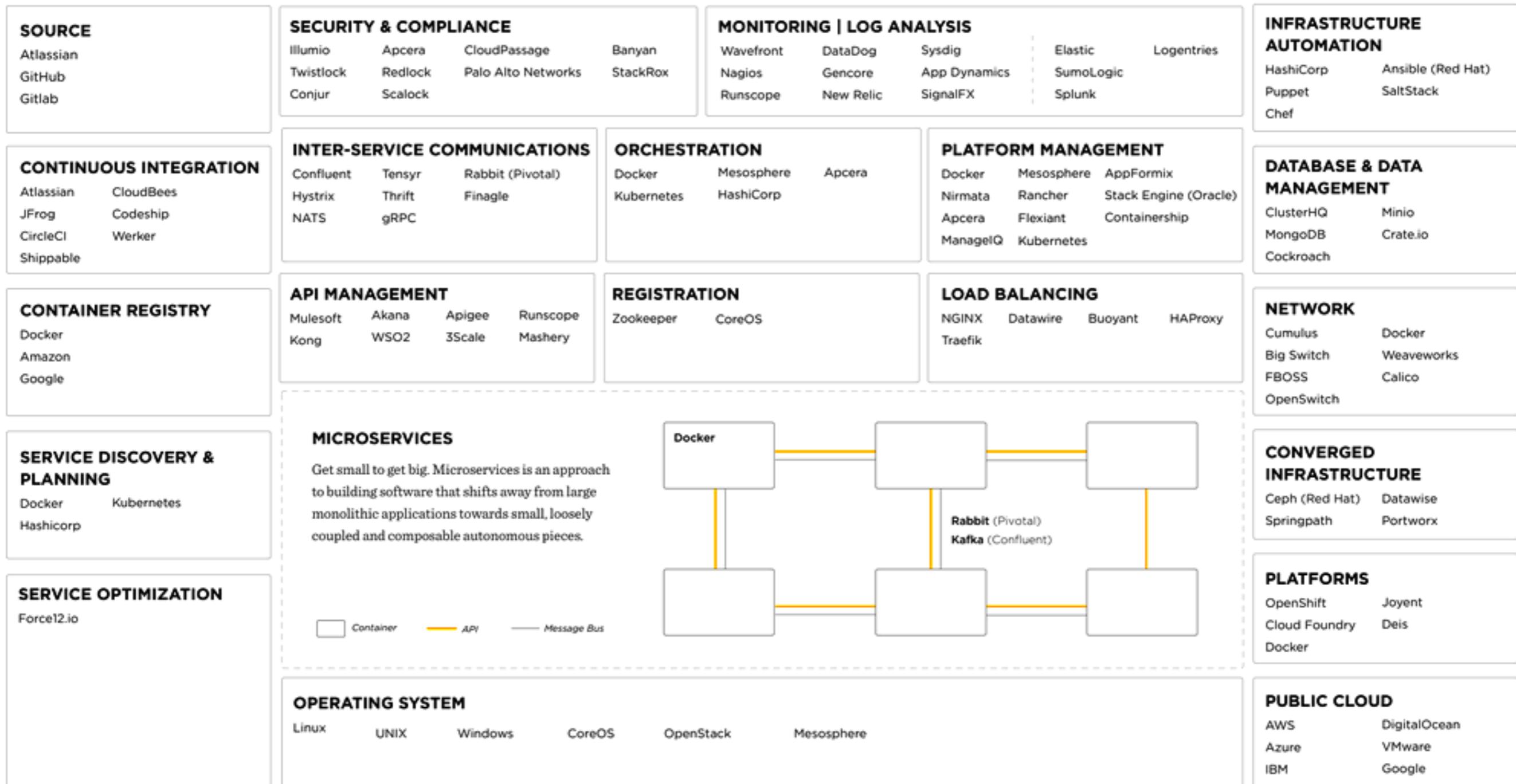
Communication
Collaboration

Monitoring



Features & Technology





Includes both companies and open source projects

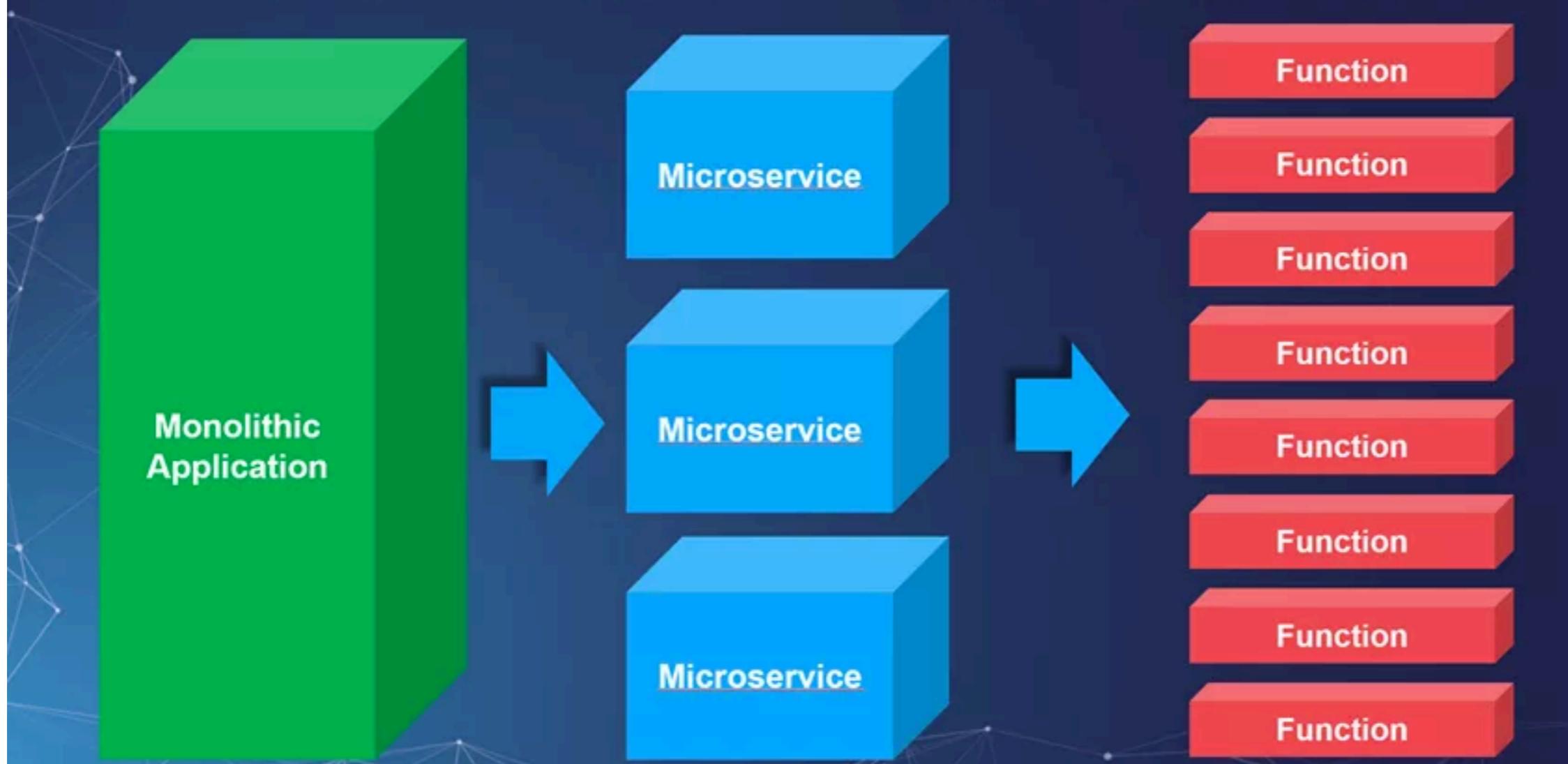
Version 1.2 | 01.18.16



more ...



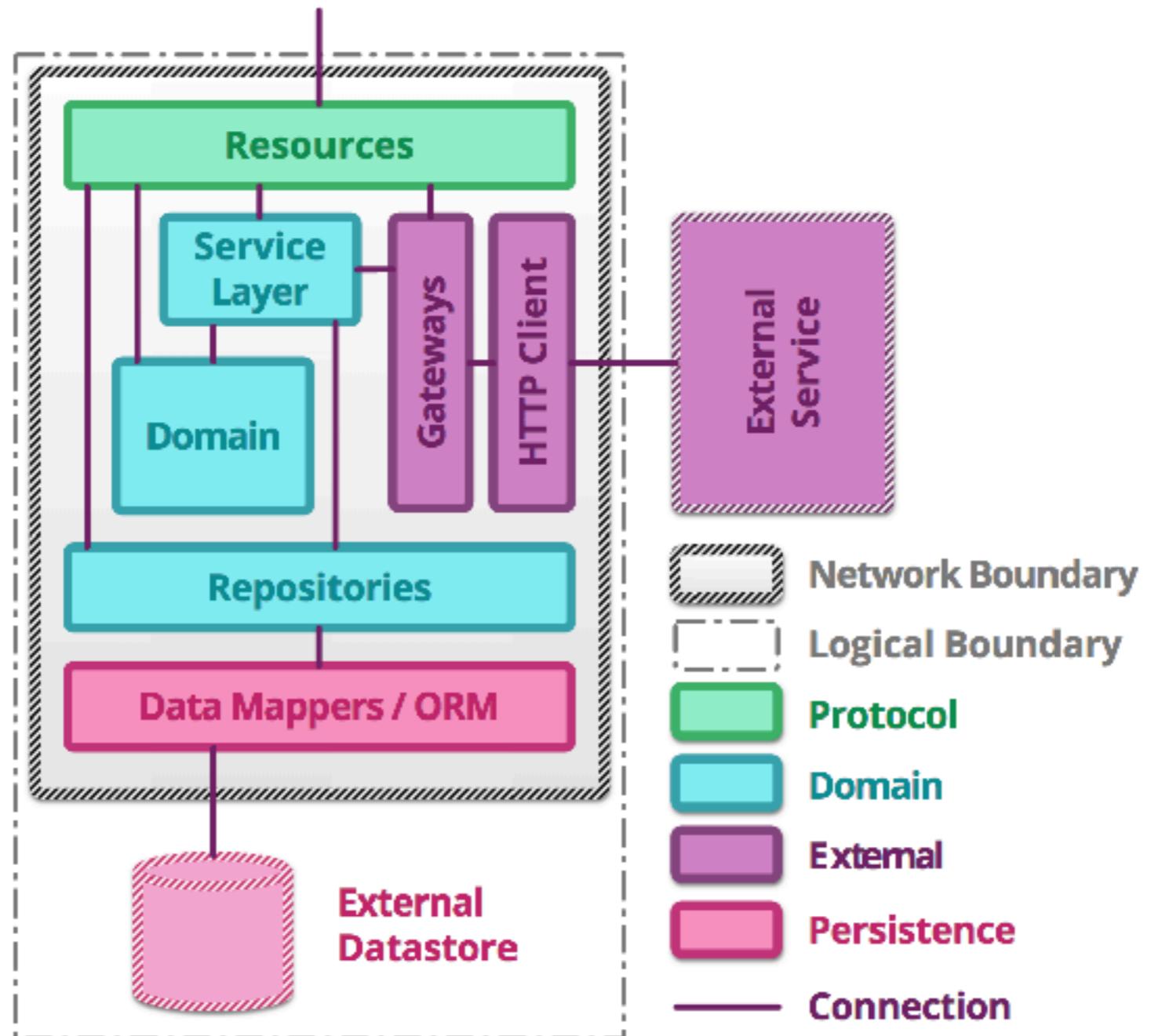
Monolithic vs Microservice vs FaaS



How to Design ?



Service structure



<https://martinfowler.com/articles/microservice-testing>



Let's workshop with Design



E-commerce system



1. Search product by name

Adidas NMD

🔍

🔍 ร้านค้า ทางการ ⏪ Taobao คอลเลคชัน ⏪ ไลฟ์สไตล์ & เดินเงิน ⏪ ไอศครีม สเตเพิม

350 คันพันสินค้าสำหรับ "Adidas NMD"

เรียงตาม: ความเป็นที่นิยม ▾

จำนวนคณิต: grid

Adidas Yeezy Boost 350 V2 Beluga 2.0 (AH2203)	Adidas NMD R1 Pimeknit Core Black / Core Black...	Adidas NMD R1 PK Japan Triple Black (BZ0220)	POCA SHOE NMD Sneakers Fashion รองเท้า ถักส่อง ผ้าใบ ...	Adidas NMD R1 Color Core Black/Icey Blue (BY9951)
฿28,900.00 ฿20,900.00 -28%	฿9,900.00 ฿15,000.00 -34%	฿12,900.00 ฿15,000.00 -14%	฿399.00 ฿1,900.00 -79% 🔥	฿7,990.00 ฿12,000.00 -33%
รายละเอียด	รายละเอียด	รายละเอียด	รายละเอียด	รายละเอียด



2. Choose a product

Adidas NMD

🔍

🔍 ร้านค้า ทางการ

淘 Taobao คอลเลกชัน

⠇ ไฟฟ์สไตร์ & เติมเงิน

⠇ ใส่โค้ด ลดเพิ่ม

350 ค้นพบสินค้าสำหรับ "Adidas NMD"

เรียงตาม: ความเป็นที่นิยม

จำนวนคณิต:

Adidas Yeezy Boost 350 V2 Beluga 2.0 (AH2203)	Adidas NMD R1 Pimeknit Core Black / Core Black...	Adidas NMD R1 PK Japan Triple Black (BZ0220)	POCA SHOE NMD Sneakers Fashion รองเท้า ลำลอง ผ้าใบ ...	Adidas NMD R1 Color Core Black/Icey Blue (BY9951)
฿28,900.00 ฿30,000.00 -28%	฿9,900.00 ฿15,000.00 -34%	฿12,900.00 ฿15,000.00 -14%	฿399.00 ฿1,000.00 -79% 🔥	฿7,990.00 ฿12,000.00 -33%
รายละเอียด	รายละเอียด	รายละเอียด	รายละเอียด	รายละเอียด



3. Show product detail

POCA SHOE NMD Sneakers Fashion รองเท้า ลำลอง ผ้าใบ ผู้หญิง-ผู้ชาย แฟชั่น
ราคาถูกswyxy Sport Unisex รุ่น PSN-Black/White

★★★★☆ (70) แสดงความคิดเห็น

ชื่อ Poca Shoes | เพิ่มเติม สุภาพบุรุษ จาก Poca Shoes



2 Weeks Warranty by Seller [เพิ่มเติม](#)

- สวมใส่สบาย [เพิ่มเติม](#)

เลือก ขนาด

ขนาด [เปลี่ยน](#)

399 บาท

ราคาปกติ 1,900 บาท,
ประหยัดทันที 79%
ราคาโปรโมชั่นสามารถใช้ได้กับ 25/2/2018

ใส่ตะกร้า

← วิธีการสั่งซื้อ



4. Add product to basket

POCA SHOE NMD Sneakers Fashion รองเท้า ลำลอง ผ้าใบ ผู้หญิง-ผู้ชาย แฟชั่น
ราคาถูกswyga Sport Unisex รุ่น PSN-Black/White

★★★★ (70) แสดงความคิดเห็น

ชื่อ Poca Shoes | เพิ่มเติม สุภาพบุรุษ จาก Poca Shoes



2 Weeks Warranty by Seller [เพิ่มเติม](#)

- สวมใส่สบาย [เพิ่มเติม](#)

เลือก ขนาด

ขนาด [เลือก](#)

ขนาด [เลือก](#)

399 บาท

ราคาปกติ 1,900 บาท,
ประหยัดทันที 79%
ราคาโปรโมชั่นสามารถใช้ได้กับ 25/2/2018

ใส่ตะกร้า



5. Show data in basket

✓ สินค้า 1 ชิ้น ได้ถูกเพิ่มเข้าไปยังตะกร้าสินค้าของคุณ



POCA SHOE NMD Sneakers
Fashion รองเท้า ล่าสุด ผ้าใบ ผู้หญิง-ผู้ชาย แฟชั่น ราคาถูกswy Sport
Unisex รุ่น PSN-Black/White

ไซส์: EU:40

Poca Shoes

399 บาท

1,900 บาท 79% ลด

ตะกร้าสินค้าของคุณ (1 สินค้า)

มูลค่าสินค้า: **399 บาท**

ยอดสุทธิ รวมภาษีมูลค่าเพิ่ม (จำนวน): **399 บาท**

[เลือกชื่อสินค้าต่อ](#)

[ชำระค่าสินค้า](#)

People Who Bought This Item Also Bought



กางเกงสแลคขายาว Hopper Progress พัฒนา ทรงขากระบอก

900 บาท

67% ลด

299 บาท



6. Checkout

✓ สินค้า 1 ชิ้น ได้ถูกเพิ่มเข้าไปยังตะกร้าสินค้าของคุณ



POCA SHOE NMD Sneakers
Fashion รองเท้า ล่าสุด ผ้าใบ ผู้หญิง-ผู้ชาย แฟชั่น ราคาถูกswy Sport
Unisex รุ่น PSN-Black/White

ไซส์: EU:40

Poca Shoes

399 บาท

1,900 บาท 79% ลด

ตะกร้าสินค้าของคุณ (1 สินค้า)

มูลค่าสินค้า: **399 บาท**

ยอดสุทธิ รวมภาษีมูลค่าเพิ่ม (จำนวน): **399 บาท**

เลือกชื่อสินค้าต่อ

ชำระค่าสินค้า

People Who Bought This Item Also Bought



กางเกงสแลคขาขวาง Hopper Progress ผ้ายืด ทรงเข้ารูป

900 บาท

67% ลด

299 บาท



Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

7. Shipping

LAZADA
CO-TH

1. คำสั่งซื้อ

2. ชำระเงิน

ที่อยู่ที่จะจัดส่ง

Login for speedy checkout

ชื่อและนามสกุล	ที่อยู่	รหัสไปรษณีย์	เมือง	จังหวัด	โทรศัพท์มือถือ
อีเมล กุลมาศ อีเมล์ของห่าน	ที่อยู่	กรุงเทพมหานคร/ Bangkok	กรุงเทพฯ	กรุงเทพมหานคร/ Bangkok	+66 ๐๘๑ ๒๓๔๕๖๗๘

ทางเราจะทำการตรวจสอบเนื้องและจังหวัดของคุณ

ที่อยู่สำหรับใบเสร็จ/ในการรับของ - กรุณาเดือนของการขอรับมุ่งเพื่อทำการขอในกำกันภาษี

ข้อมูลการส่งเงินค่า

ช่องแบบชำระเงิน: พรี

Get it วันอังคาร, 27 ก.พ. - วันจันทร์, 5 มี.ค. 2018

ค่าจัดส่ง

สูปการสั่งซื้อ (1 items)

สินค้า	จำนวน	ราคารวม
POCA SHOE NMD Sneakers Fashion รองเท้า ลั่นสอง ผ้าใบ ผู้หญิง-ผู้ชาย แฟชั่น ราคาถูกสุดๆ Sport Unisex รุ่น PSN-Black/White ขนาด: EU:40	1	399 บาท
รวมค่าสินค้า		399 บาท
ยอดสุทธิ รวมความภาระค่าเพิ่ม (ถ้ามี)		399 บาท

 คุ้มครองสูงสุด 100%





8. Payment

LAZADA
.CO.TH

✓ 1. ค่าซื้อขั้นต่ำ

2. ชำระเงิน

เลือกคัวเลือกสำหรับการชำระเงิน

บัตรเดบิตหรือ เทิร์บเงินปลายทาง	ชำระเงินผ่าน เดบิตหรือ	PayPal/Amex	มอนชาร์	LINE Pay	หักบัญชีธนาคาร/ ห้องทางATM

หมายเหตุบัตร

ชื่อบนบัตร

วันที่บัตรหมดอายุ mm yy

CCV / CVV

ข้อมูลใบกำกับภาษีไม่สามารถเปลี่ยนแปลงได้หลังการสั่งซื้อสินค้า

ล็อก สั่งซื้อสินค้า

สมัครรับข่าวสารกับลาซาด้าเพื่อรับข่าวลือและข้อเสนอสุดพิเศษ

โดยการร่วมค้ำประกันของคุณ, คุณยอมรับข้อกำหนดของทางลาซาด้า **ในการซื้อสินค้าทางช่องทางที่กำหนดให้ และ ร้ออกกฎหมายที่ใช้ในประเทศไทย**

ส่งที่ **ไทย**

Somkiat Puisungnoen
122/64 , Soi Phahonyothin 2, Phahonyothin Road Prom Condo กรุงเทพมหานคร/ Bangkok - พญาไท/ Phaya Thai - 10400 โทรศัพท์: 0868696209

สรุปการสั่งซื้อ (1 items)

สินค้า	จำนวน	ราคาร
POCA SHOE NMD Sneakers Fashion รองเท้า ลำลอง ถ้าใบ สีฟ้า-เขียว แพลตฟอร์มสีขาว Soot Unisex รุ่น PSN-Black/White ขนาด: EU:40	1	399
สั่งแบบธรรมด้า วันอังคาร, 27 ก.พ. - วันเสาร์, 3 มี.ค. 2018		

กรอกคุณปวงส่วนลดที่นี่

ขึ้นชั้น

มูลค่าสินค้า ค่าซื้อขั้นต่ำ	399 บาท
ยอดสุทธิ รวมภาษีมูลค่าเพิ่ม (มีภาษี)	399 บาท

ทุมแพร่องค้า 100%

JUN-2016
more security · more value · LAZADA VERIFIED



9. Confirm to order

LAZADA
.CO.TH

✓ 1. ค่าซื้อขั้นต่ำ

2. ชำระเงิน

เลือกคัวเลือกสำหรับการชำระเงิน

บัตรเดบิตหรือ เครดิต	เก็บเงินปลายทาง	ชำระเงินผ่าน เคาน์เตอร์	PayPal/Amex	มอนชาร์	LINE Pay <small>robbit LINE Pay</small>	หักบัญชีธนาคาร/ ช่องทางATM
<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

หมายเลขบัตร

ชื่อบนบัตร

วันที่บัตรหมดอายุ

mm
yy
CCV / CVV
?

ข้อมูลใบสำคัญไม่สามารถเปลี่ยนแปลงได้หลังการสั่งซื้อสินค้า

สั่งซื้อสินค้า

สมควรระบุรายละเอียดตามส่วนลดและเงื่อนไขของสูตรพิเศษ

ส่งที่ แก้ไข

Somkiat Puisungnoen

122/64 , Sci Phahonyothin 2, Phahonyothin Road Prom Condo
กรุงเทพมหานคร/ Bangkok - พญาไท/ Phaya Thai - 10400
โทรศัพท์: 0868696209

สรุปการสั่งซื้อ (1 items)

สินค้า	จำนวน	ราคา
POCA SHOE NMD Sneakers Fashion รองเท้า ลั่วสัง สีขาว-สีเขียว แฟชั่น ราคากลางๆ Scott Unisex รุ่น PSN- Black/White ขนาด: EU:40	1 <small>เม็ดกด</small>	399

ส่งแบบธรรมด้า
วันอังคาร, 27 ก.พ. - วันเสาร์, 3 มี.ค. 2018

ขึ้นชั้น

มูลค่าสินค้า
ค่าซื้อขั้นต่ำ

ยอดสุทธิ
รวมภาษีมูลค่าเพิ่ม (แล้วมี)

ทุมแพร่องค้า
100%

Lazada Security Approved
JUN-2016
more security - less risk

โดยการวางแผนซื้อของคุณ, คุณจะรับข้อเสนอของทางล่าช้าสำหรับในการซื้อสินค้าทางช่องทางที่กำหนดให้ และ รับยกเว้นและเพื่อนไป

Microservices © 2017 - 2018 Siam Chamnkit Company Limited. All rights reserved.

148

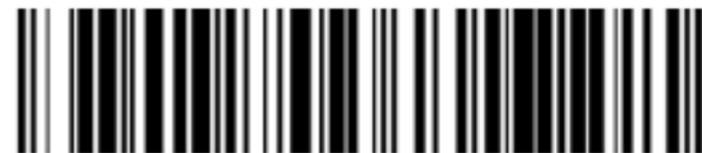
10. Summary



ใบแจ้งการชำระเงิน(PaySlip)

Counter Service Co., Ltd.

เลขที่ใบแจ้ง สั่นด้า/Invoice No:	3779254692
ผู้ชำระ เงิน/Payer:	Somkiat Puisungnoen
วันที่รายการ / Transaction Date:	25/02/2018 23:33
กำหนดชำระเงิน / Expired Date:	27/02/2018 23:33
เพื่อเข้าบัญชี / Payee:	www.lazada.co.th Tel: 020180000
รายละเอียด / Detail:	Lazada



806010855864737

จำนวนเงินที่ชำระ / Amount:

399.00 บาท /THB

* ไม่รวมค่าธรรมเนียมของเด่านี้เดอร์เซอร์วิส
(Excluding service fees at Counter Service)

คลิกปุ่ม "Print" พิมพ์ใบแจ้งการชำระเงิน
หรือ

กด "รหัส 15 หลักใต้บาร์โค้ด" เพื่อเข้าไป
ชำระเงินที่
Press "Print" button or write down
paycode 15 digits for pay in cash at
counter service(7-11)



มาสเตอร์
เคอร์ซิตี้

[Back to merchant](#)

[Print](#)



Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Try to design system



A-DAPT Blueprint

THEME:	EPIC:	FEATURE:	STORY:
DESIGNED BY:	DATE:	NOTE:	



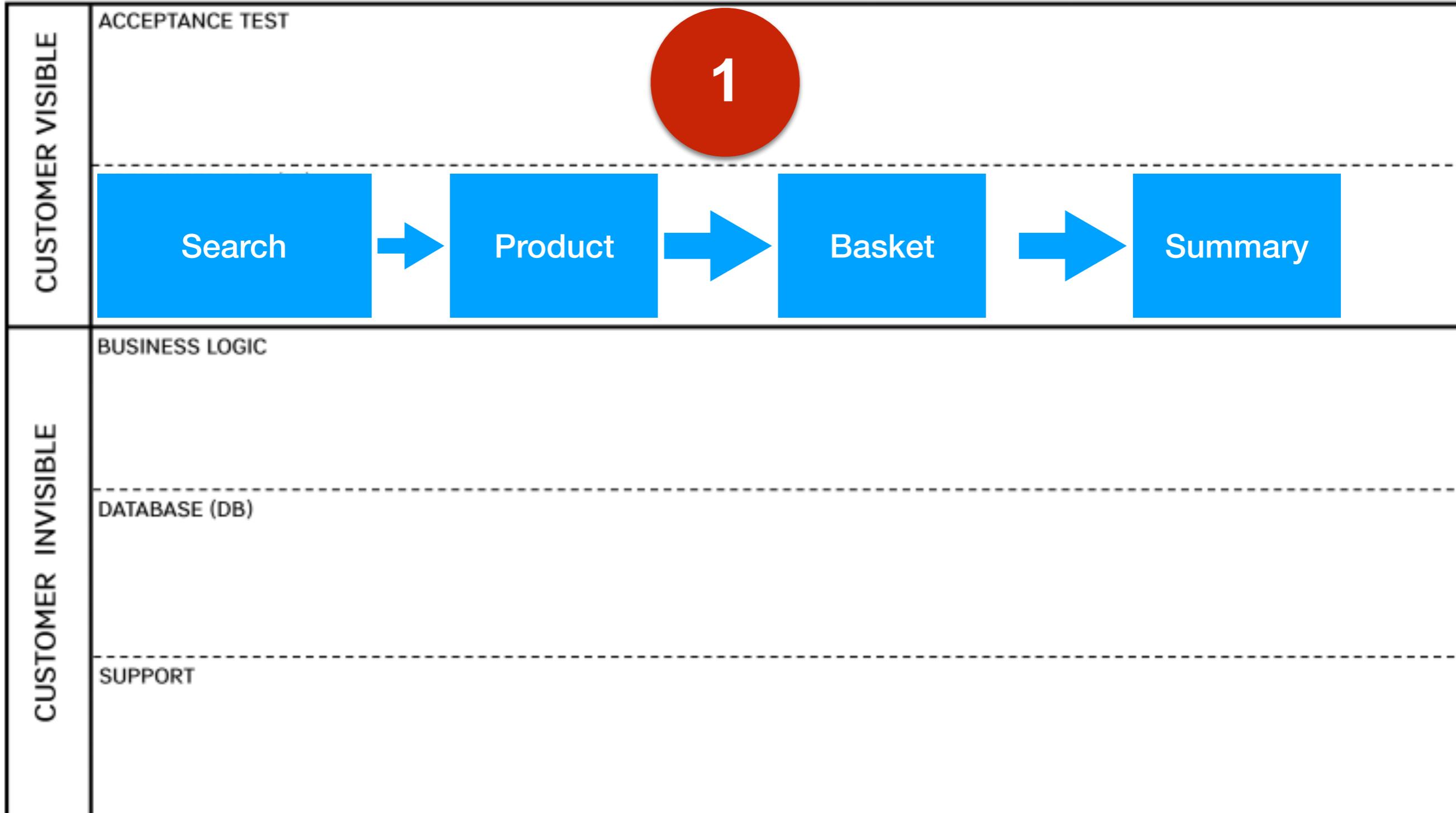
A-DAPT Blueprint

THEME: EPIC: FEATURE: STORY:

DESIGNED BY:

DATE:

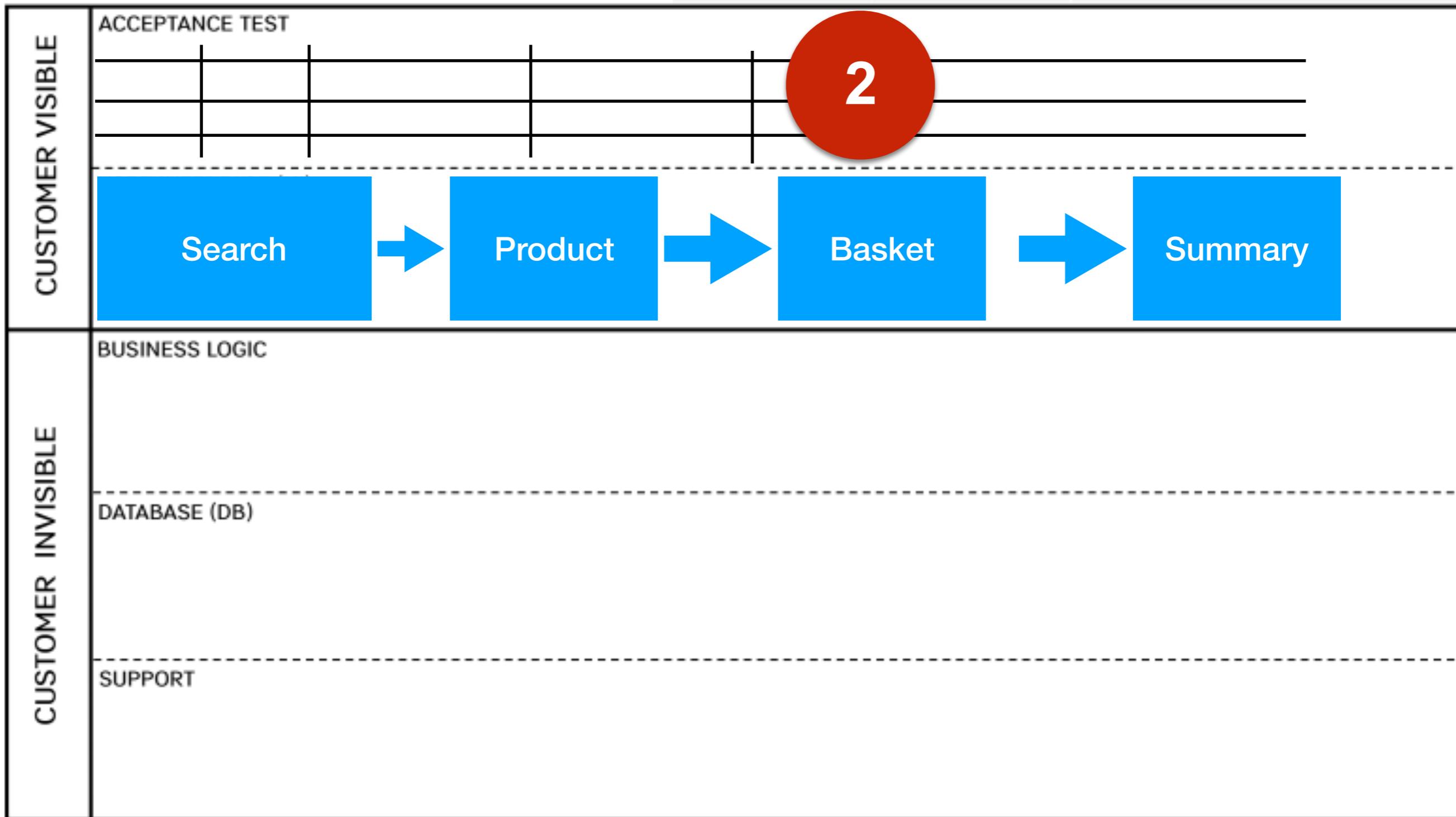
NOTE:



A-DAPT Blueprint

THEME: EPIC: FEATURE: STORY:

DESIGNED BY: DATE: NOTE:



A-DAPT Blueprint

THEME:

EPIC:

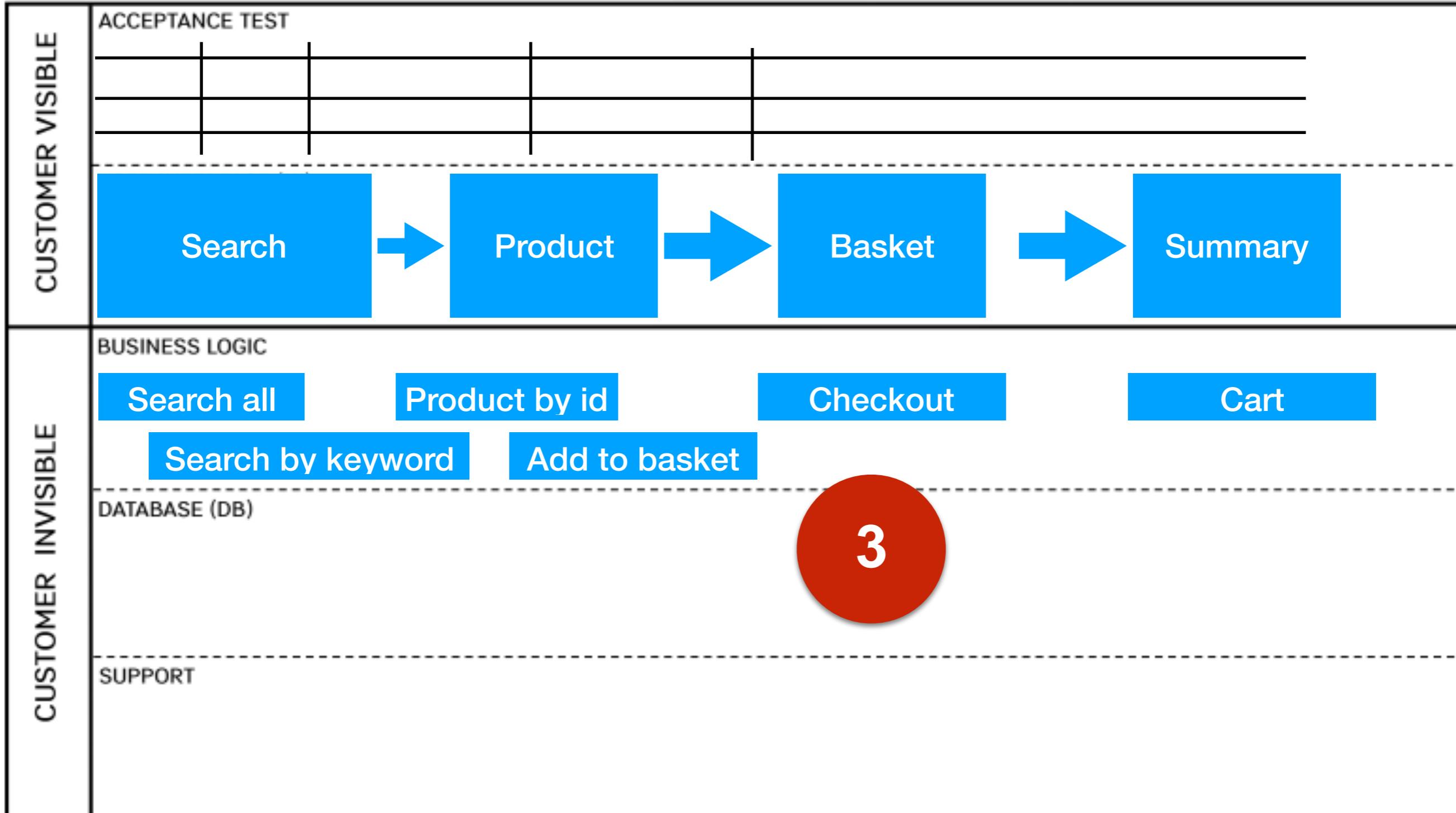
FEATURE:

STORY:

DESIGNED BY:

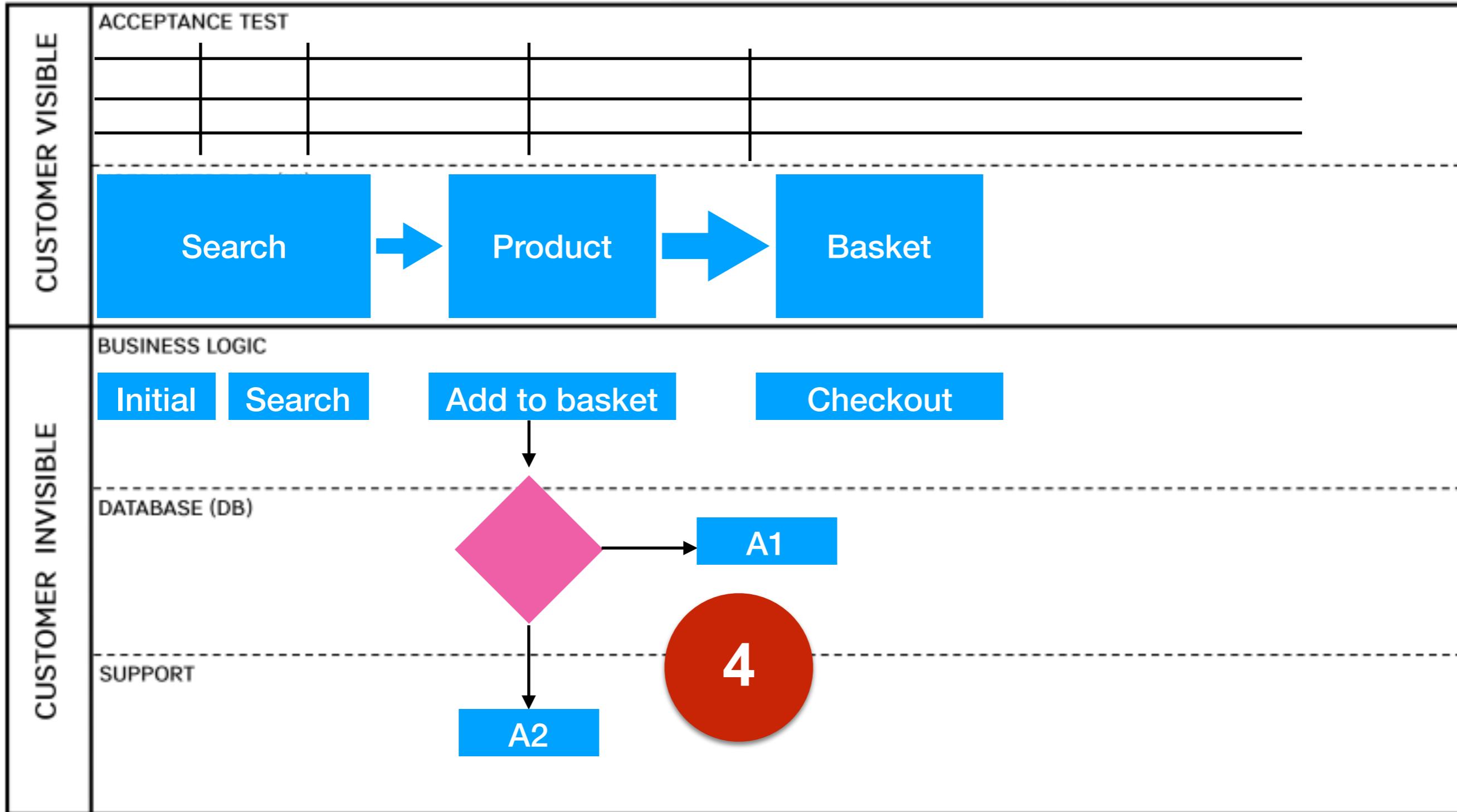
DATE:

NOTE:



A-DAPT Blueprint

THEME:	EPIC:	FEATURE:	STORY:
DESIGNED BY:	DATE:	NOTE:	



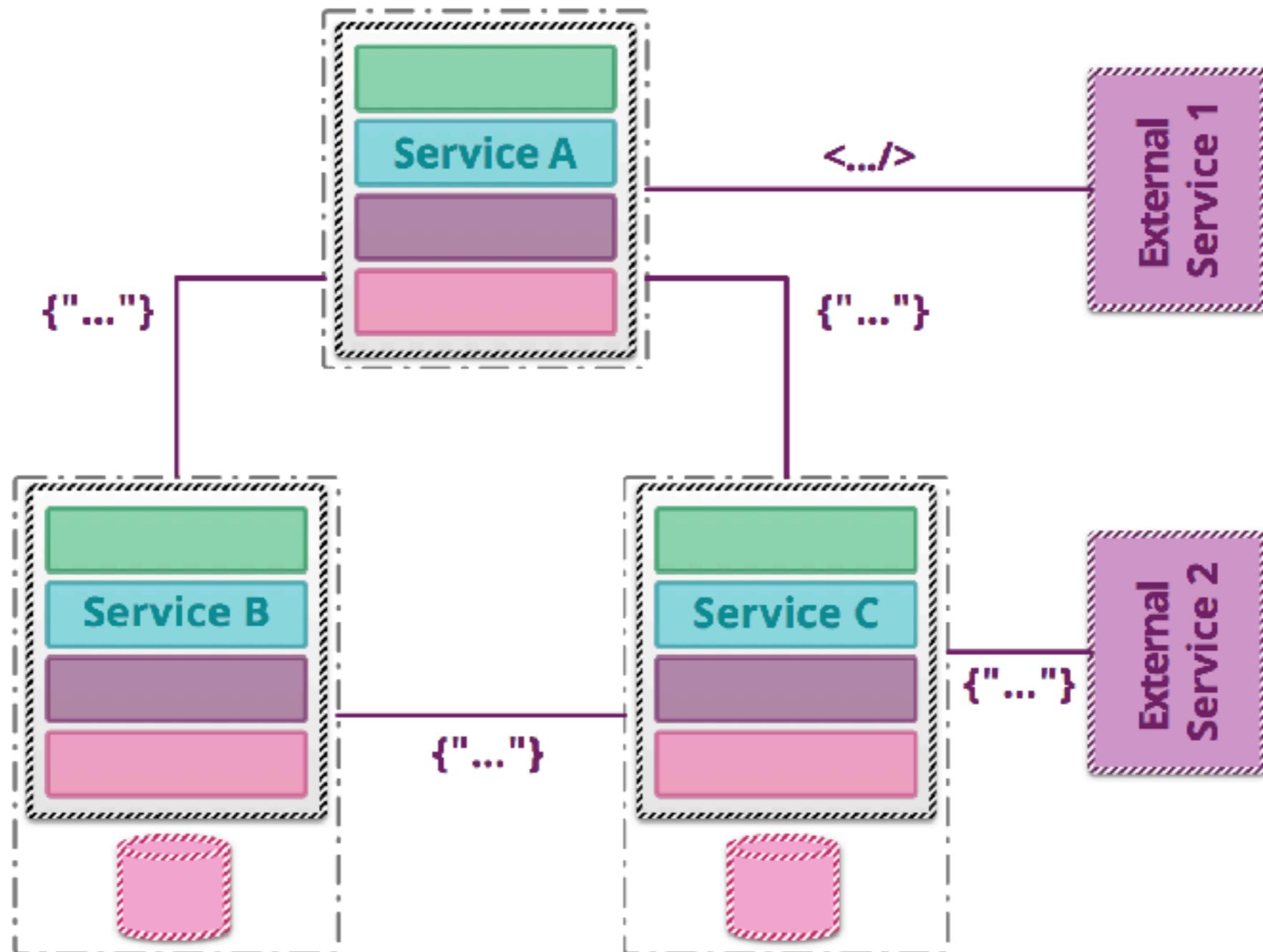
Retrospective



How to Test ?



Multiple services



<https://martinfowler.com/articles/microservice-testing>

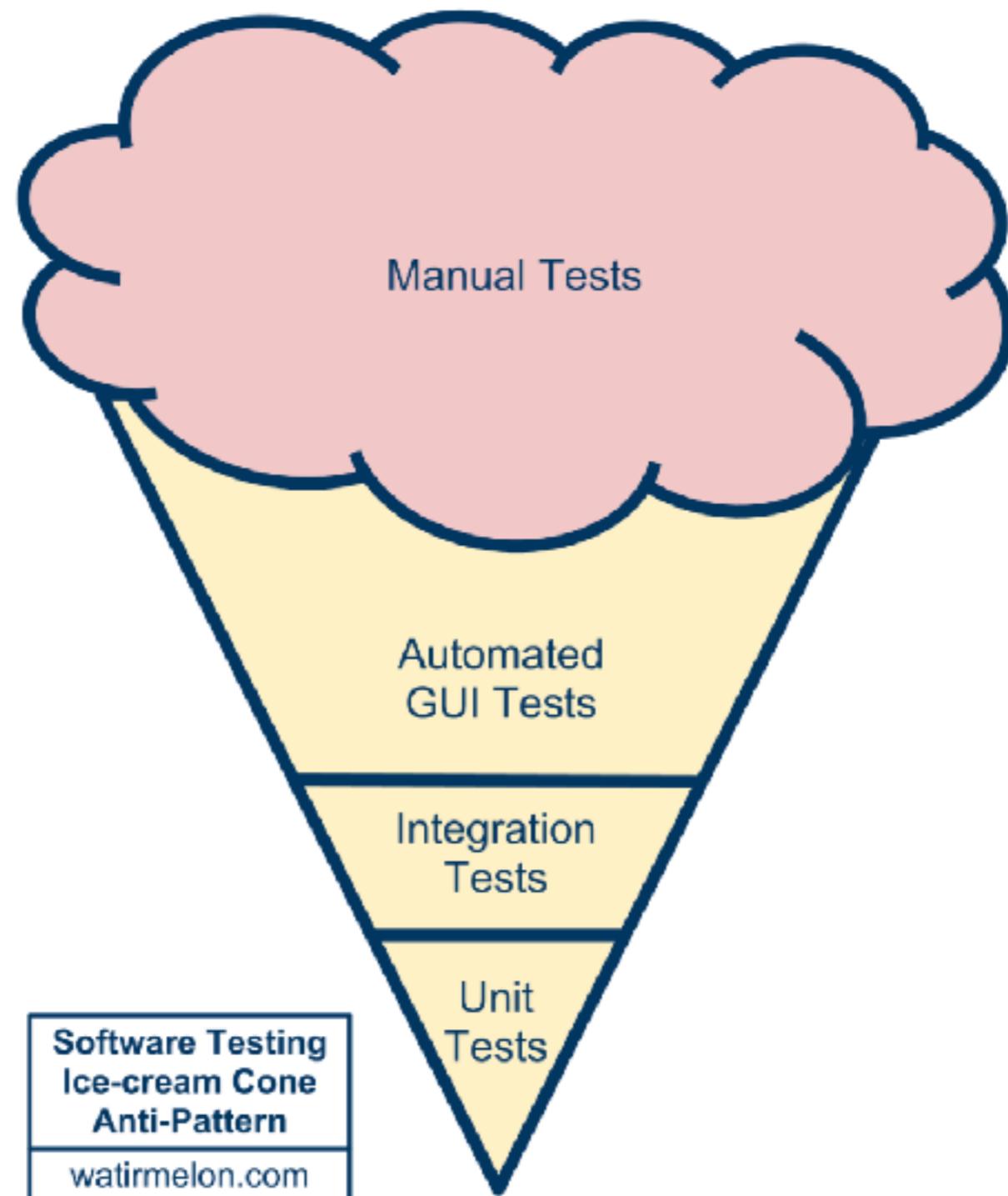


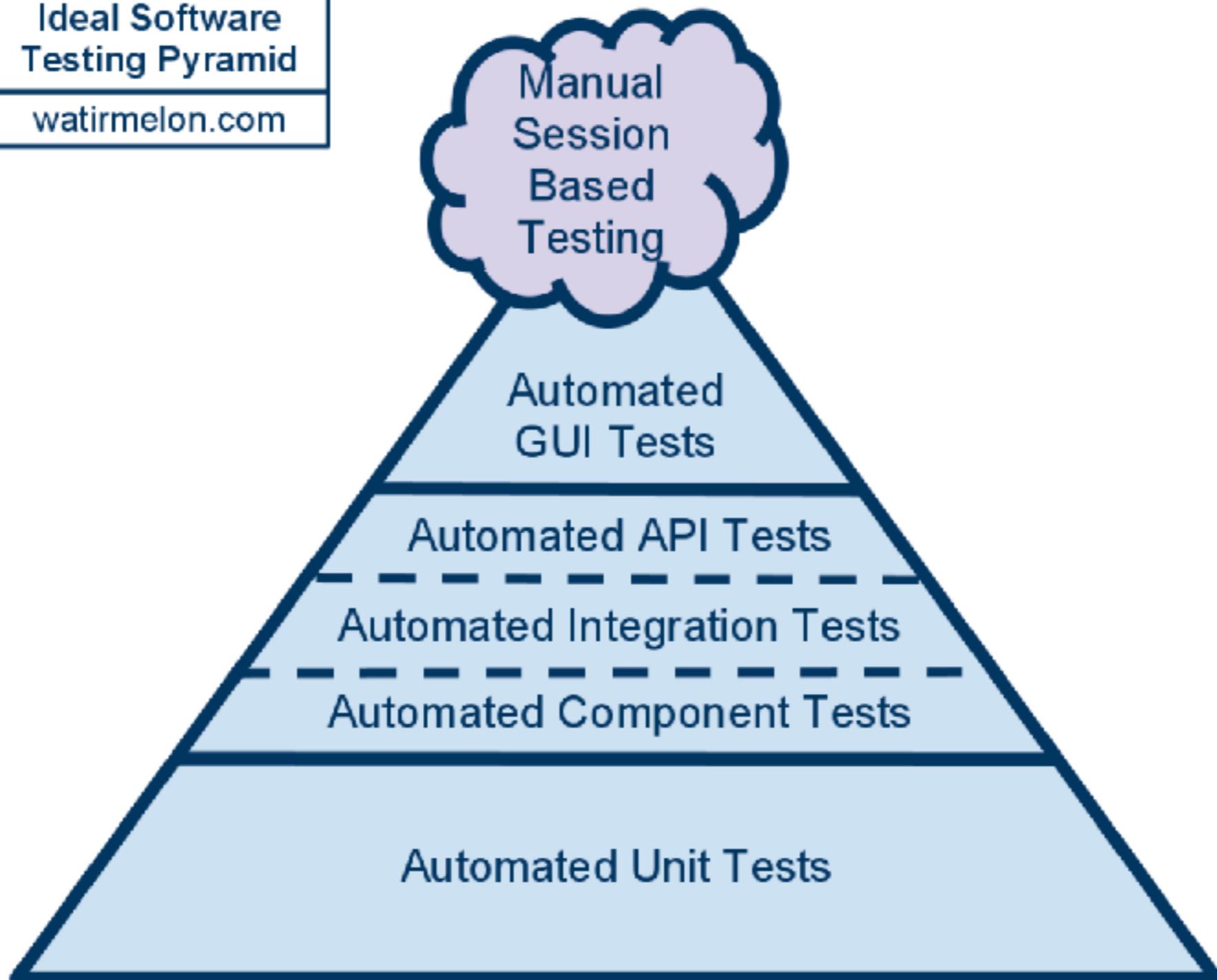
THE #1 PROGRAMMER EXCUSE
FOR LEGITIMATELY SLACKING OFF:
"TESTS ARE RUNNING"

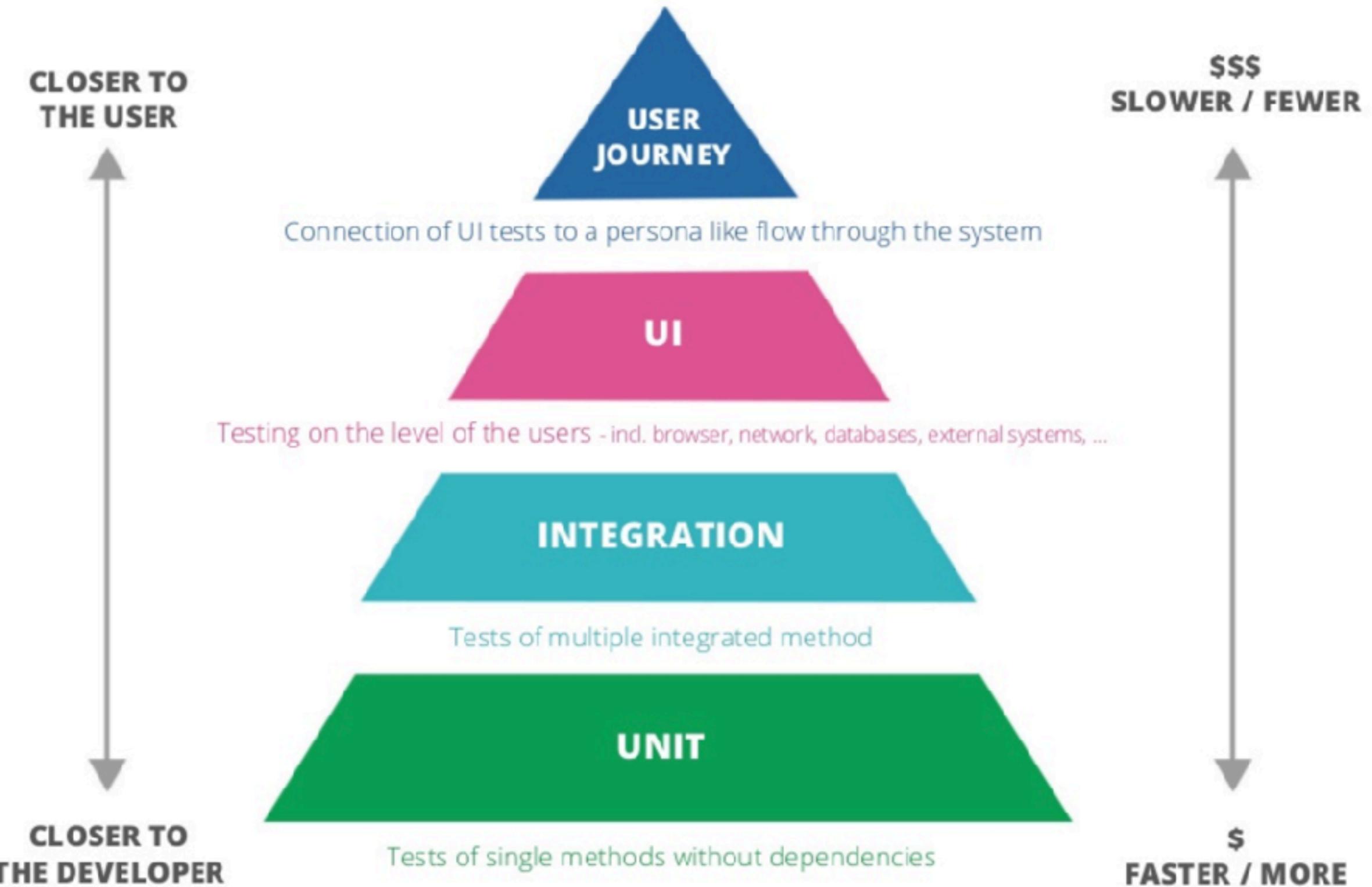


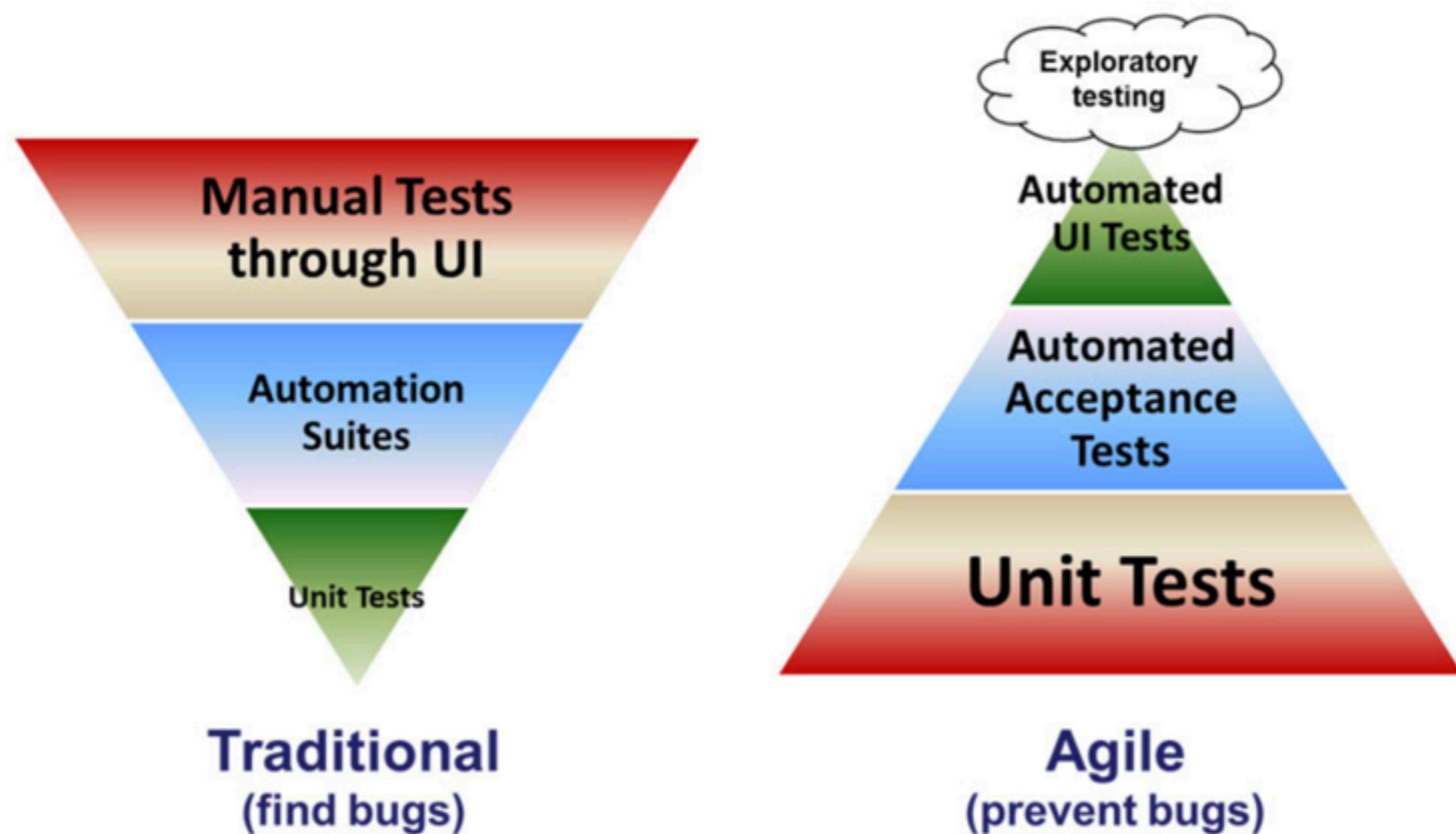
Microservice Testing





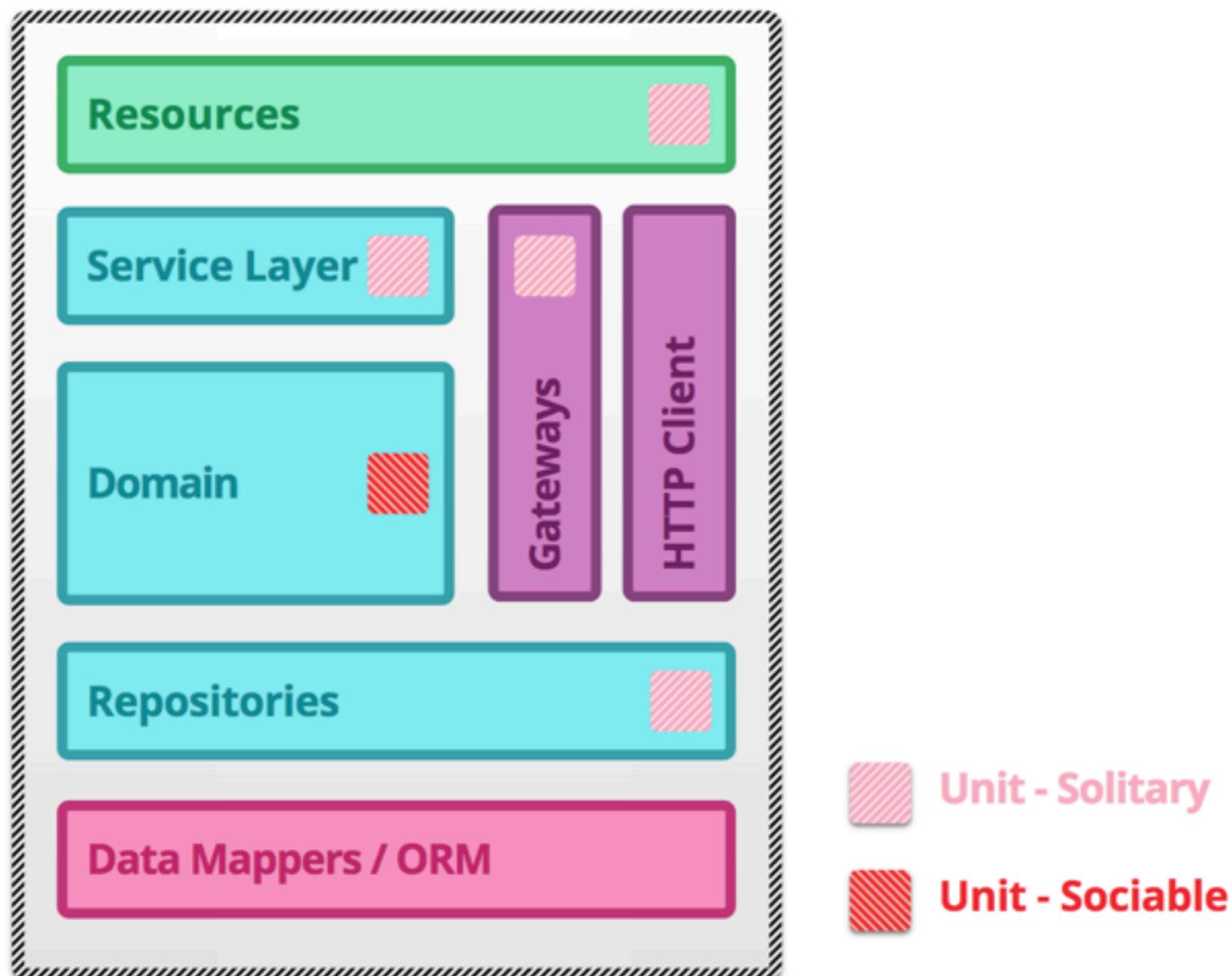




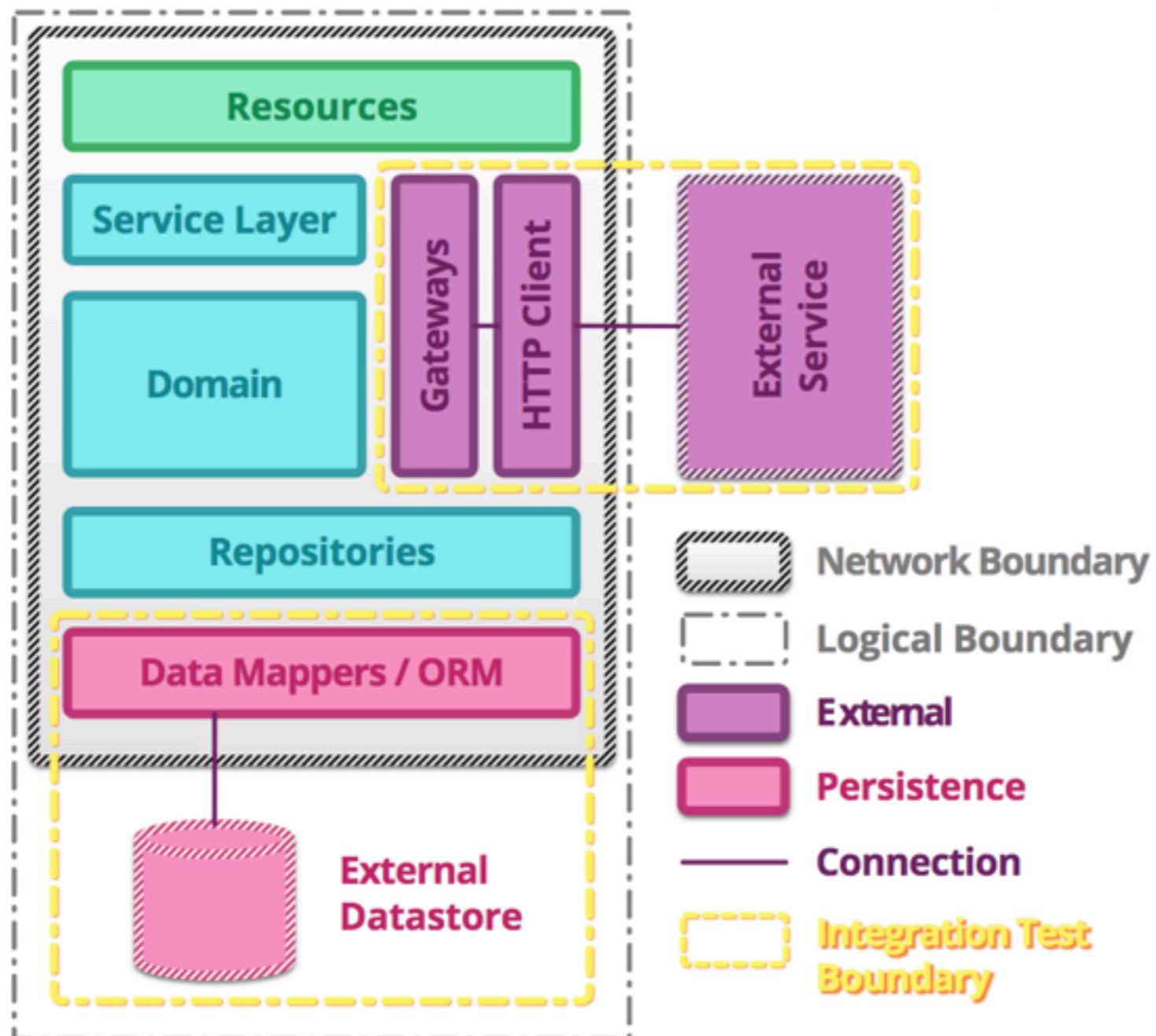




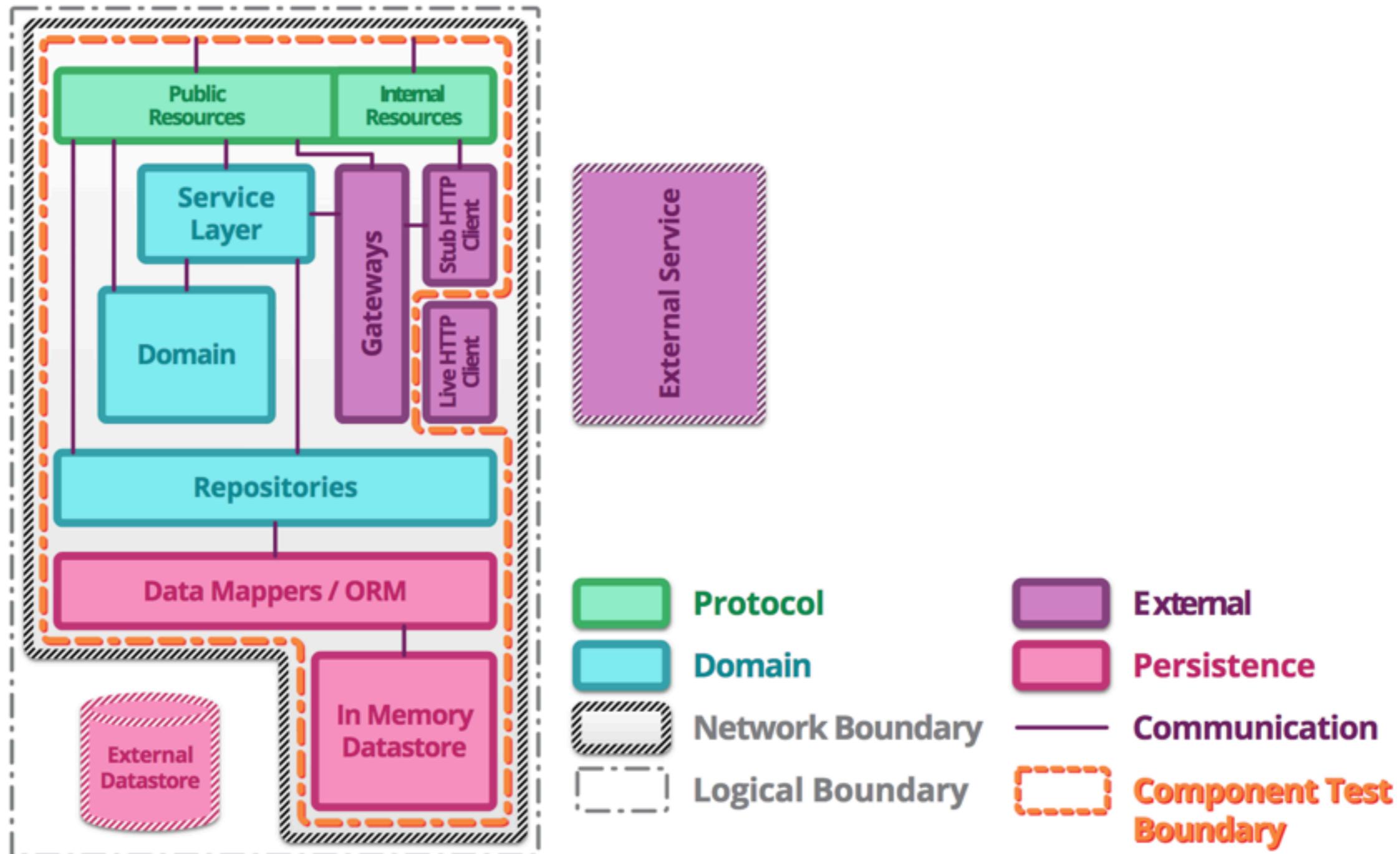
Unit testing



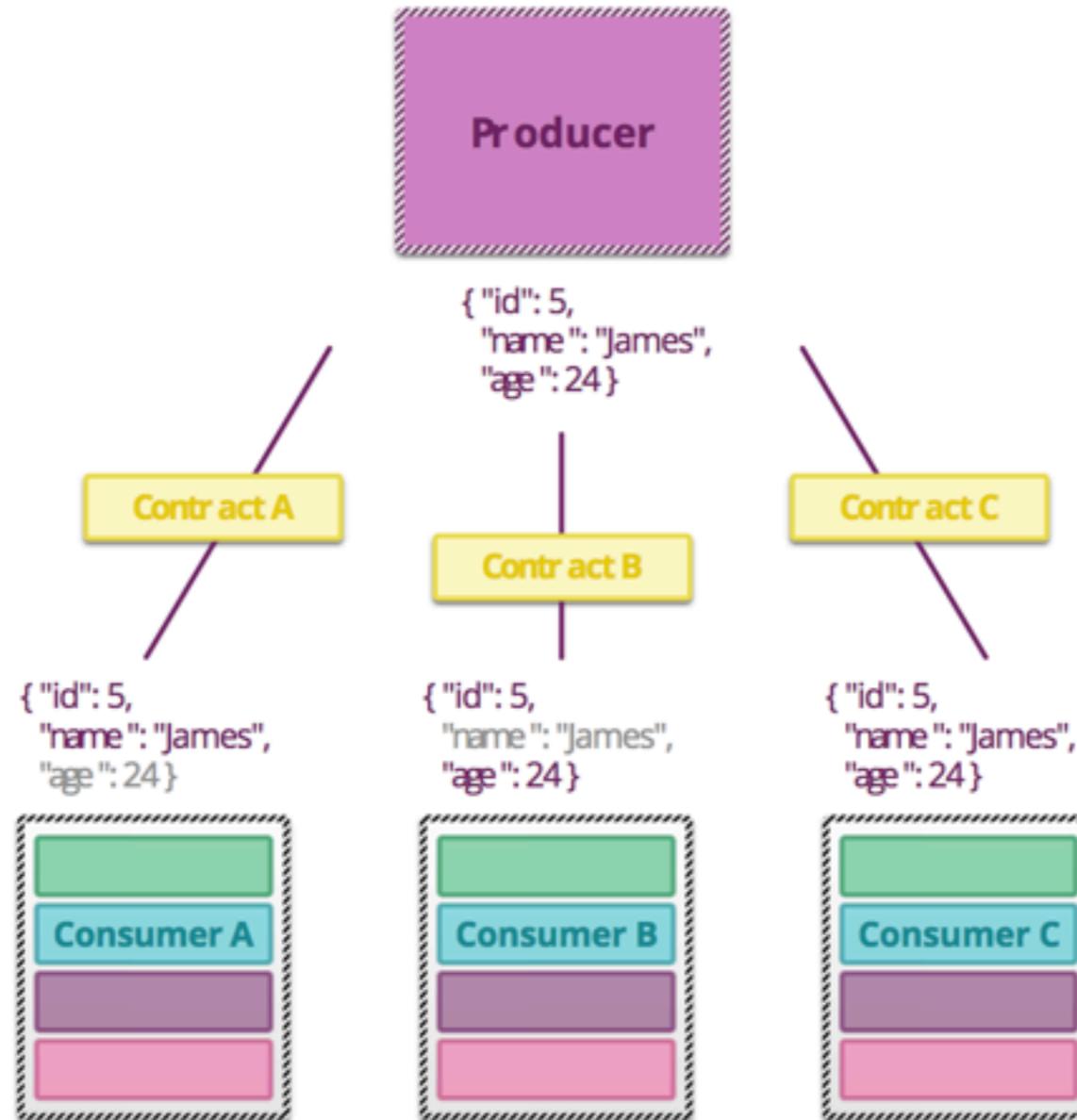
Integration testing



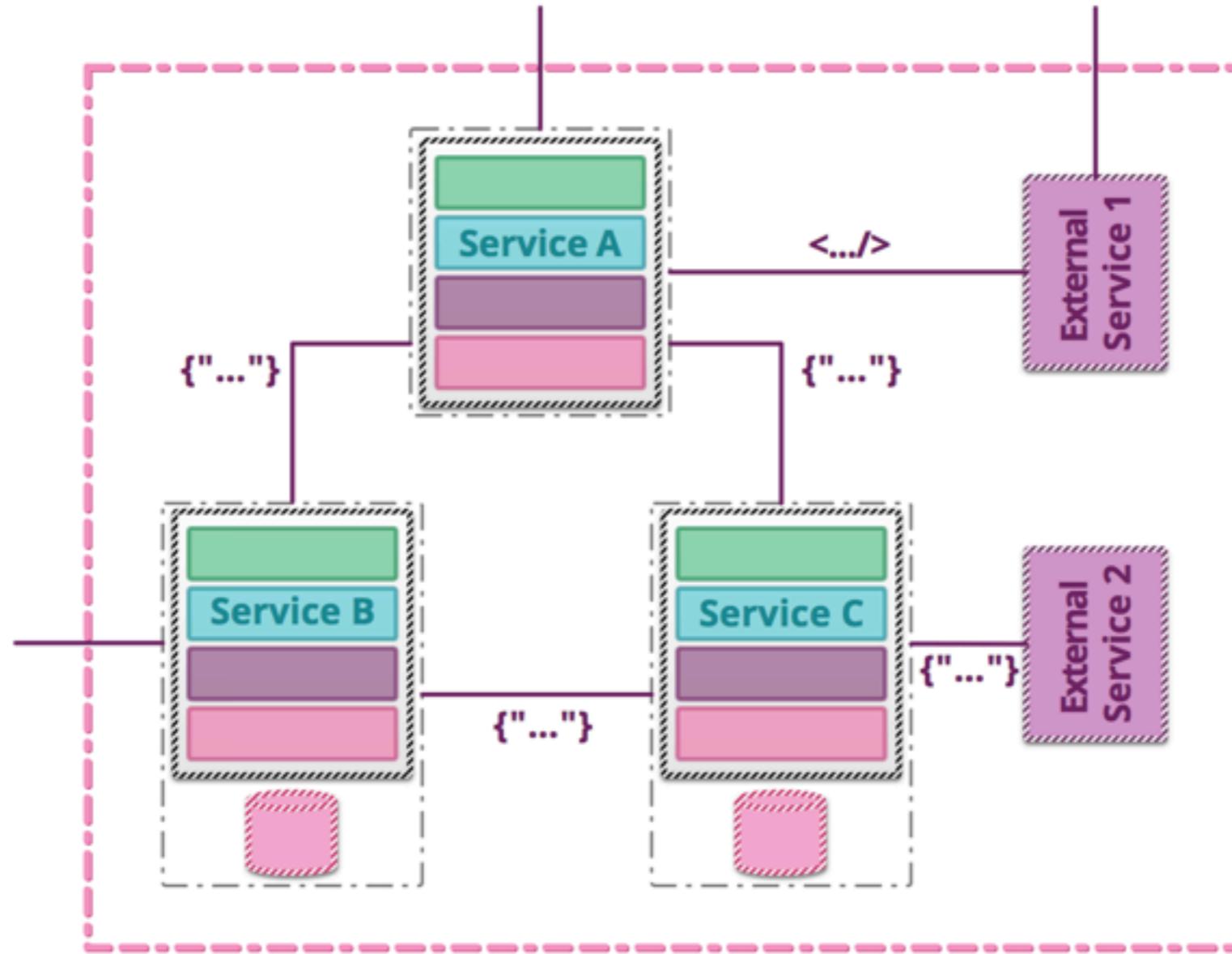
Component testing



Contract testing



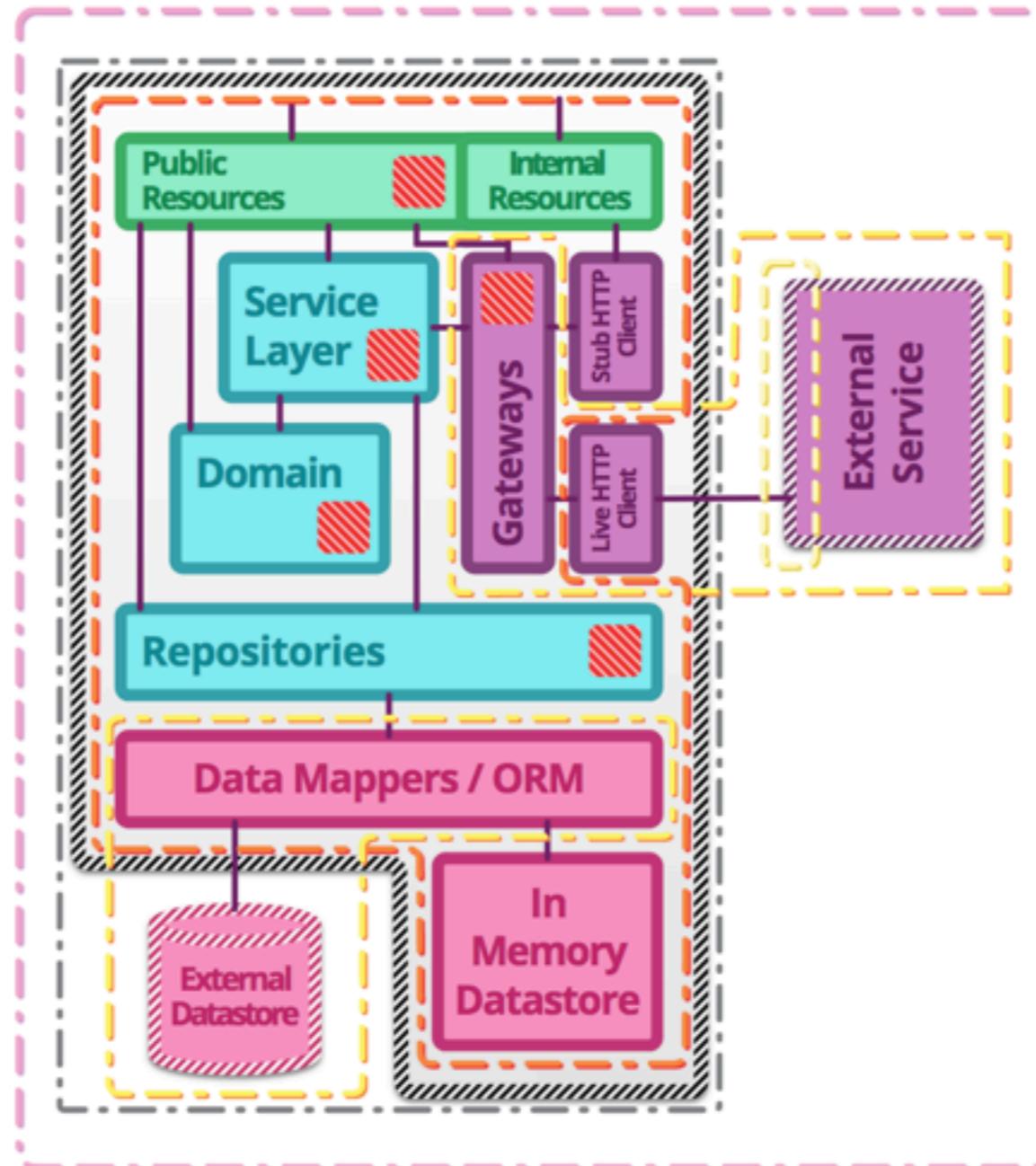
End-to-End testing



Summary

 **Unit tests** : exercise the smallest pieces of testable software in the application to determine whether they behave as expected.

 **Integration tests** : verify the communication paths and interactions between components to detect interface defects.



 **Component tests** : limit the scope of the exercised software to a portion of the system under test, manipulating the system through internal code interfaces and using test doubles to isolate the code under test from other components.

 **Contract tests** : verify interactions at the boundary of an external service asserting that it meets the contract expected by a consuming service.

 **End-to-end tests** : verify that a system meets external requirements and achieves its goals, testing the entire system, from end to end.



What is your testing strategy ?



more ...



Performance testing ?



Security testing ?



Common standards



OpenID

Allows identity and some metadata only



OAuth 2.0

Grant a client access to resources based on a newly created set of credentials



OpenID Connect

Identity on top of OAuth 2

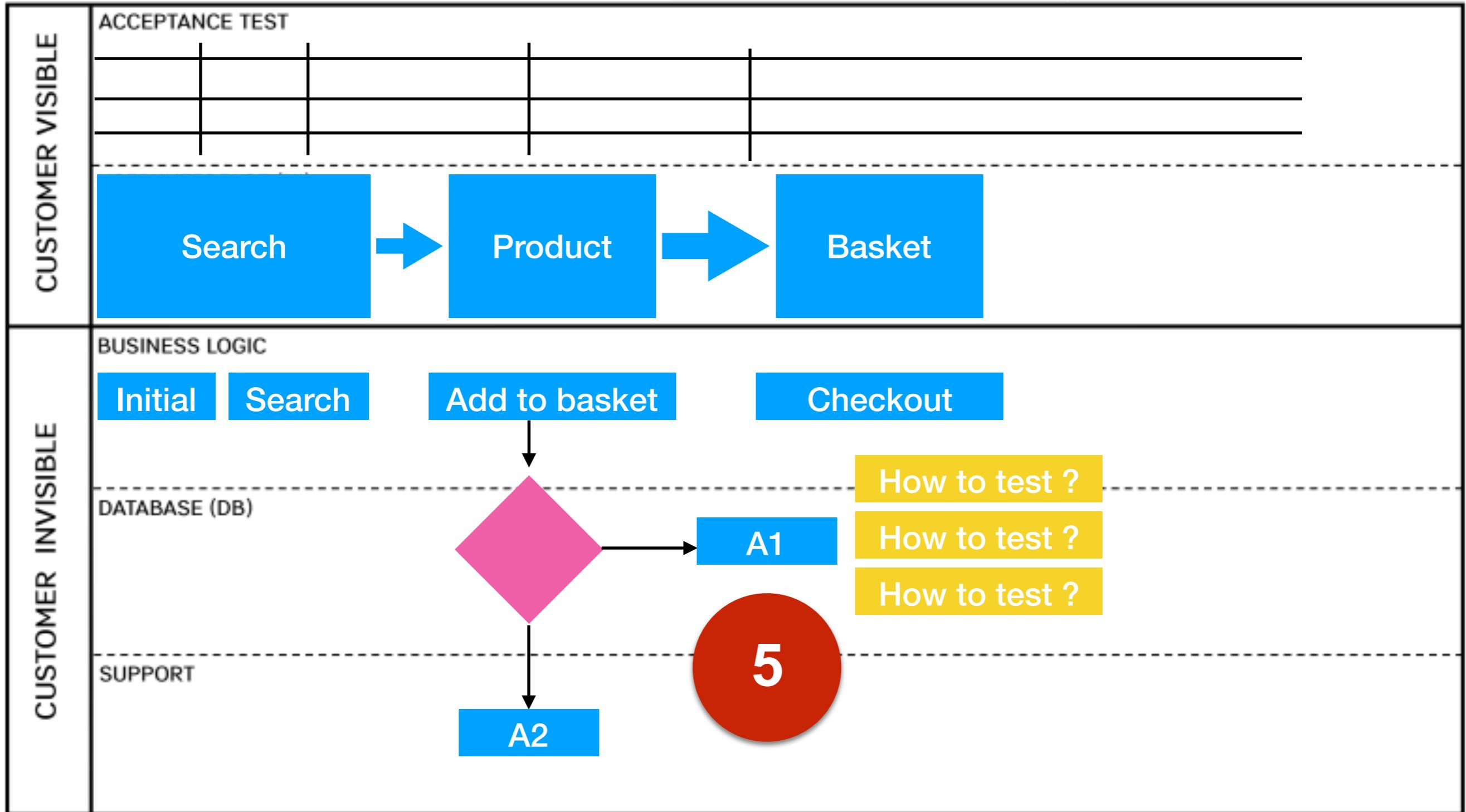


Try to test your services



A-DAPT Blueprint

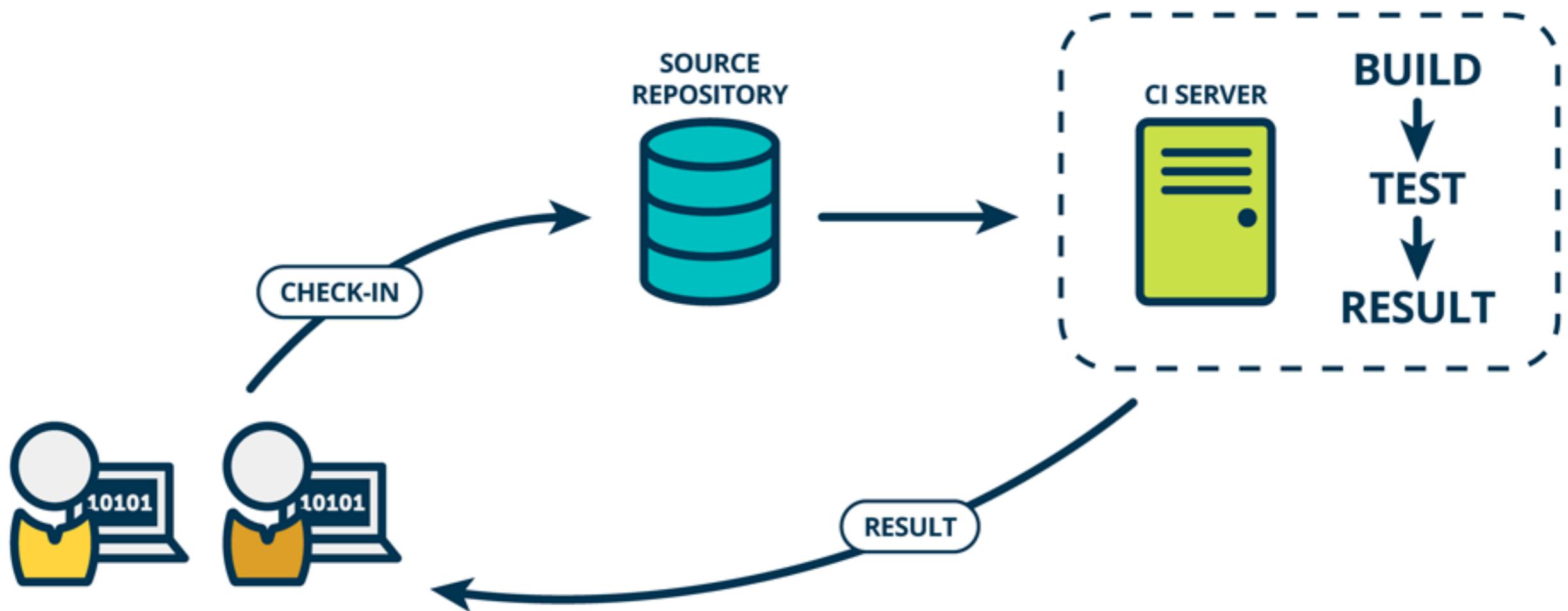
THEME:	EPIC:	FEATURE:	STORY:
DESIGNED BY:	DATE:	NOTE:	



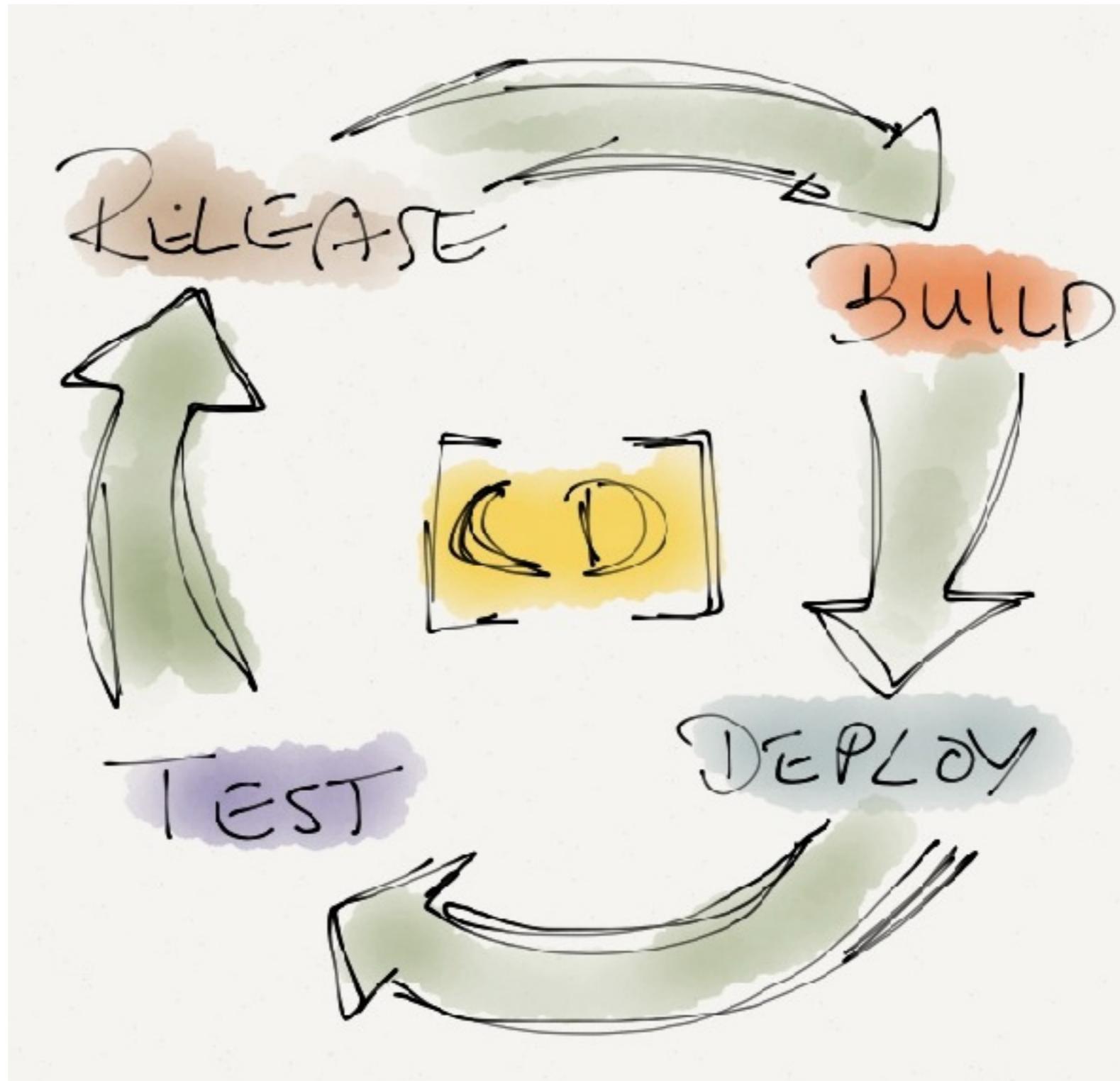
How to Deploy ?



Continuous Integration



CD ?



CD ?

CONTINUOUS DELIVERY



CONTINUOUS DEPLOYMENT



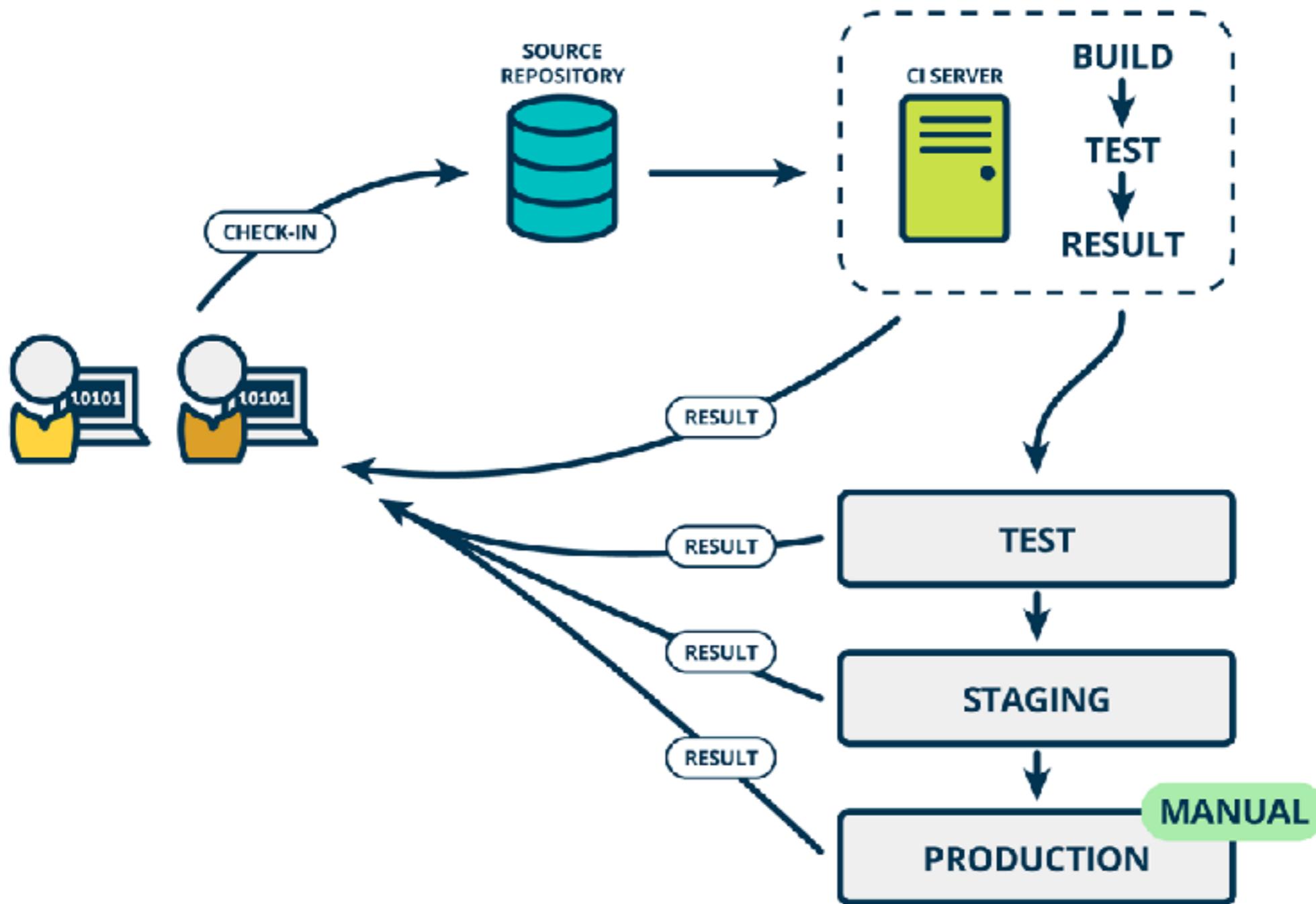
<http://blog.crisp.se/2013/02/05/yassalsundman/continuous-delivery-vs-continuous-deployment>



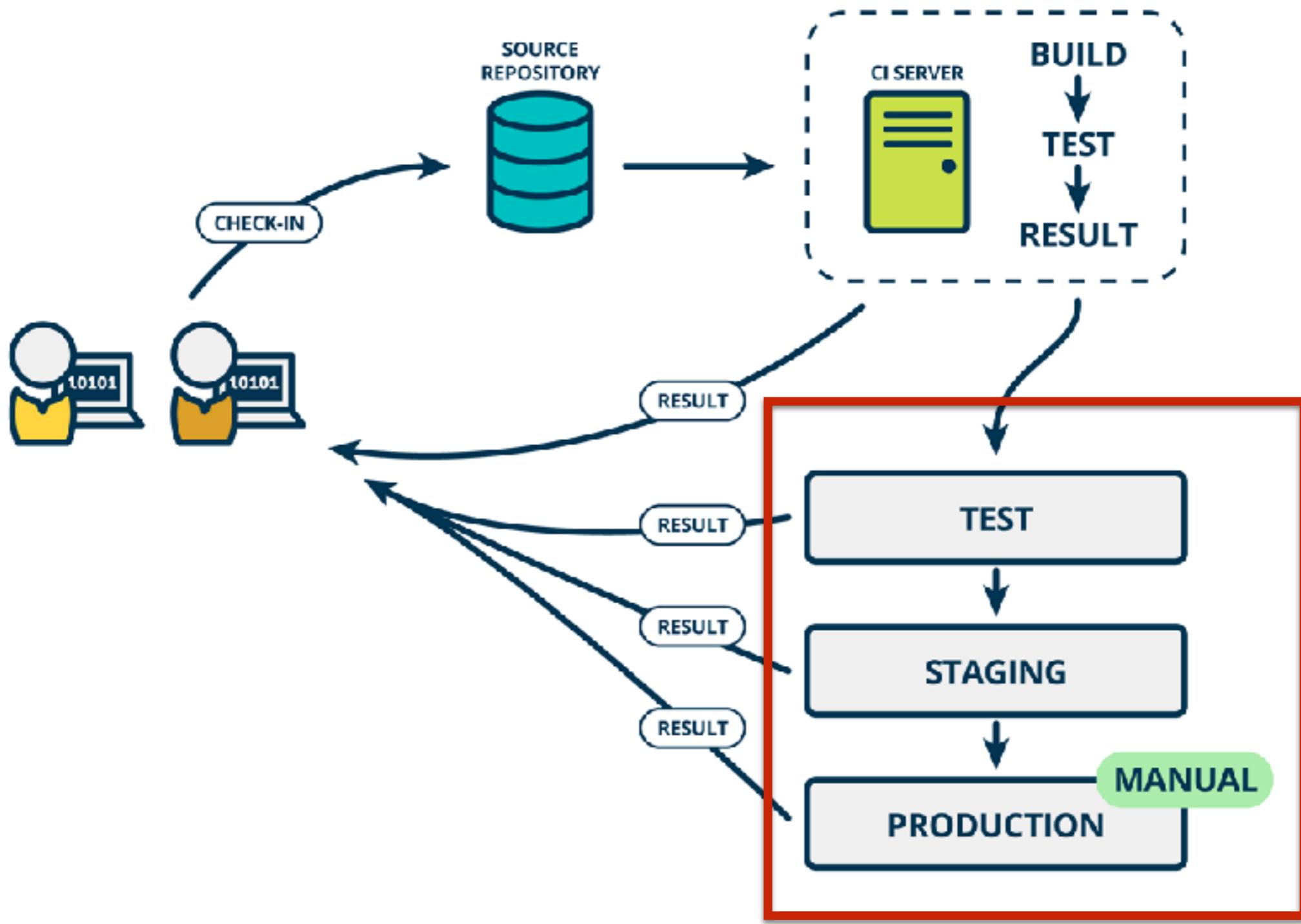
Microservices

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

Continuous Delivery



Rise of DevOps



How to Monitoring ?



How to Configuration ?



How to Scaling ?



Summary



Take to your home

Testing a monolith is easy !!,
think about your **service testing strategy**



Take to your home

Security must bundle in development
workflow but not make slow down



Take to your home

You build it you run it



Take to your home

Focus on **value**, not **quantity**



Suggestion

Don't start with a Microservices



Suggestion

Start with a monolith, keep it modular



Suggestion

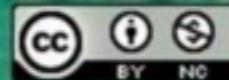
Split it into Microservices when become a problem



Are you too busy to improve?



Created by Håkan Forss @hakanforss <http://hakanforss.wordpress.com>



Thank you Q/A



