



# Microservices

In Practices with Java Technology





Somkiat Puisungnoen

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 สยามชั่นนาคุกิจ 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 Day 1

1. Cloud Native Application
2. Microservices and DevOps
3. The architecture of Microservices
4. How to model Microservices
5. Integrating multiple Microservices
6. Workshop



# Agenda Day 2

1. Testing and Developing Microservices
2. Deploying Microservices
3. Maintaining healthy Microservices
4. Monitoring Microservices
5. Scaling up your Microservices
6. Workshop



**<https://github.com/up1/course-microservice>**



Customers



“The Business”

Product Teams

Platform Teams

Infrastructure Teams

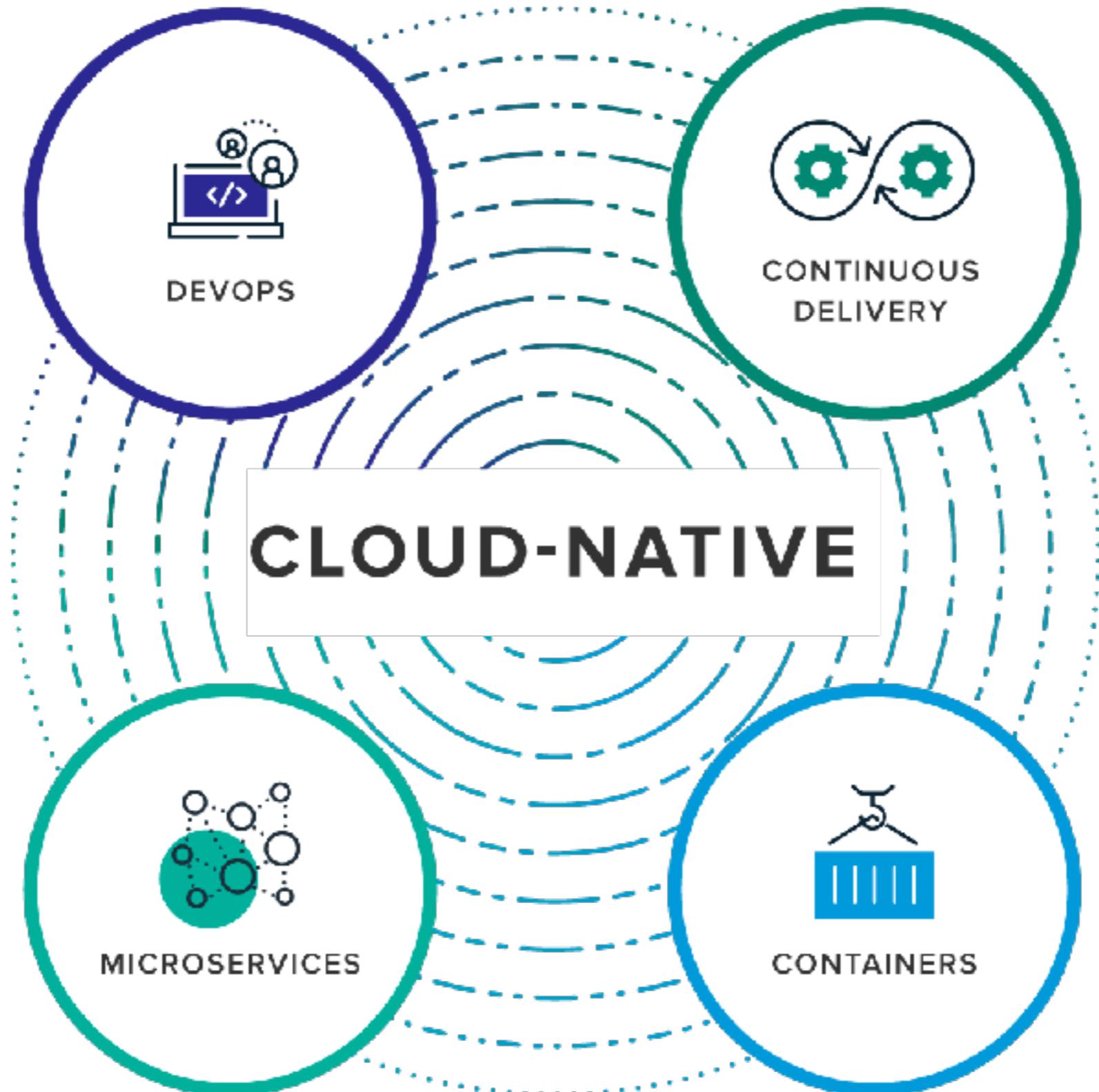
Operations Teams



Google/Amazon

<https://bravenewgeek.com/>





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

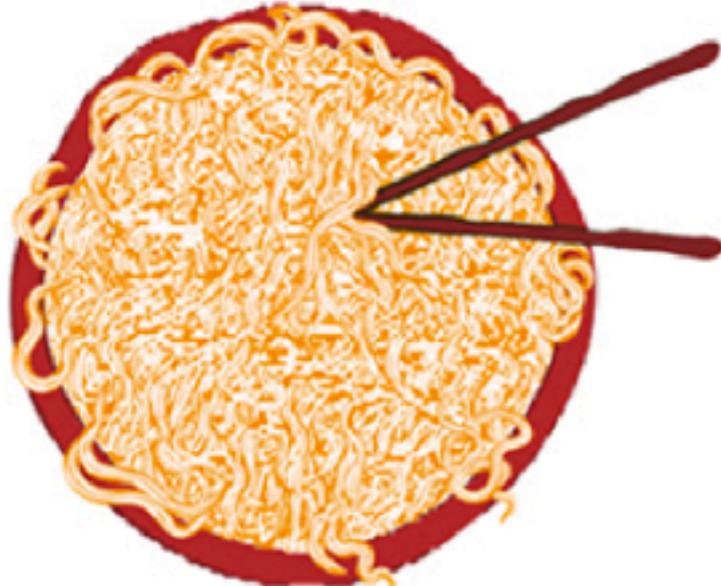


# Evolution of Architecture



### 1990s and earlier

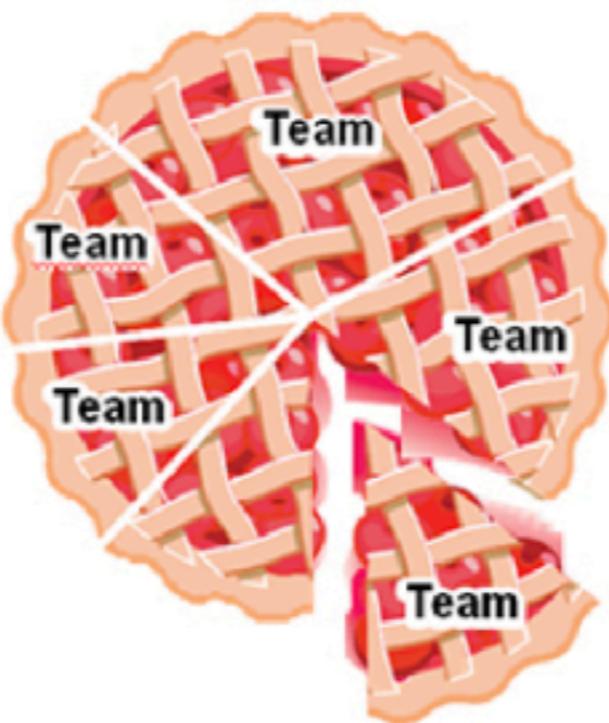
Pre-SOA (monolithic)  
Tight coupling



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

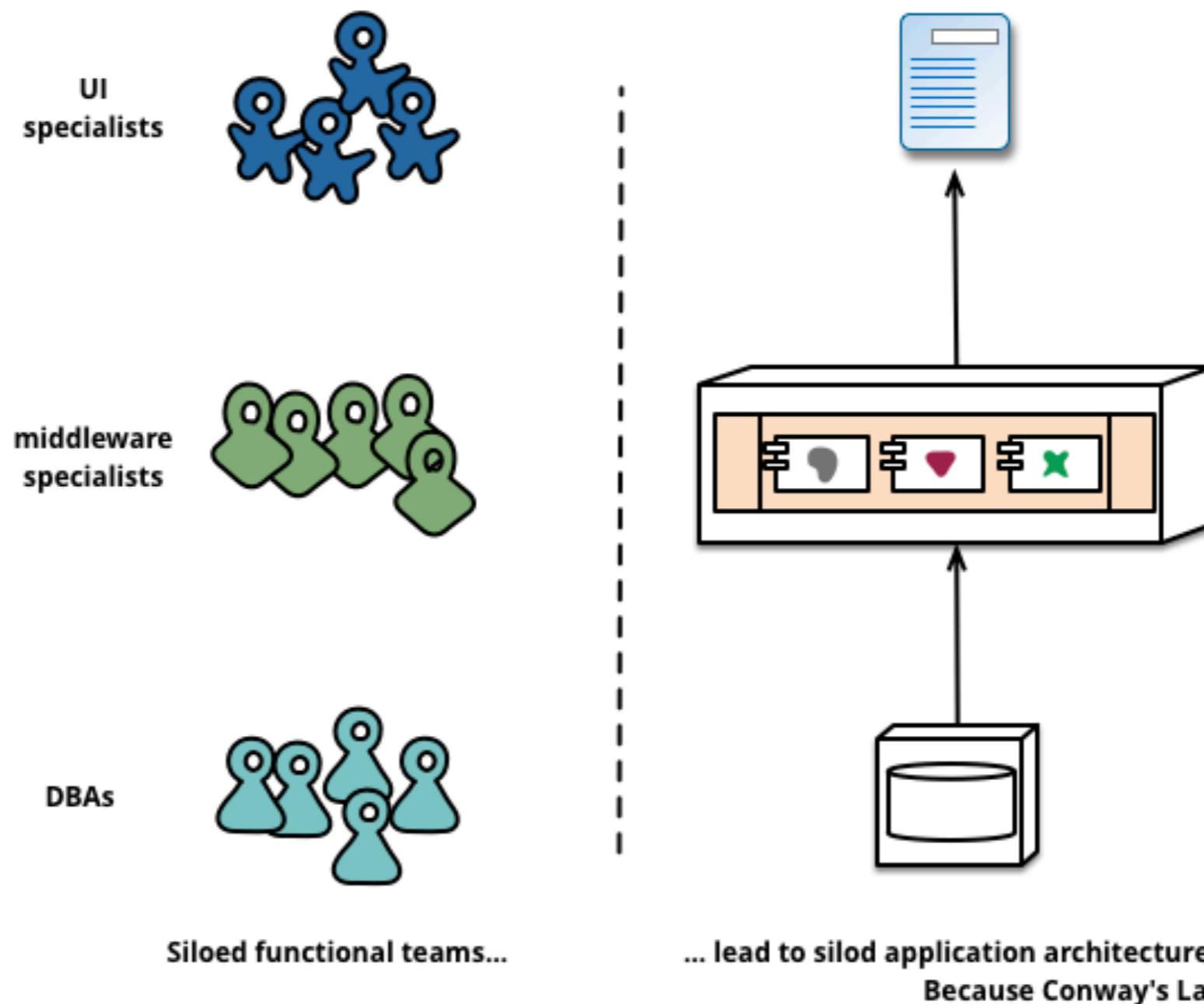
Microservices  
Decoupled



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.



# Conway's Law



# Microservices

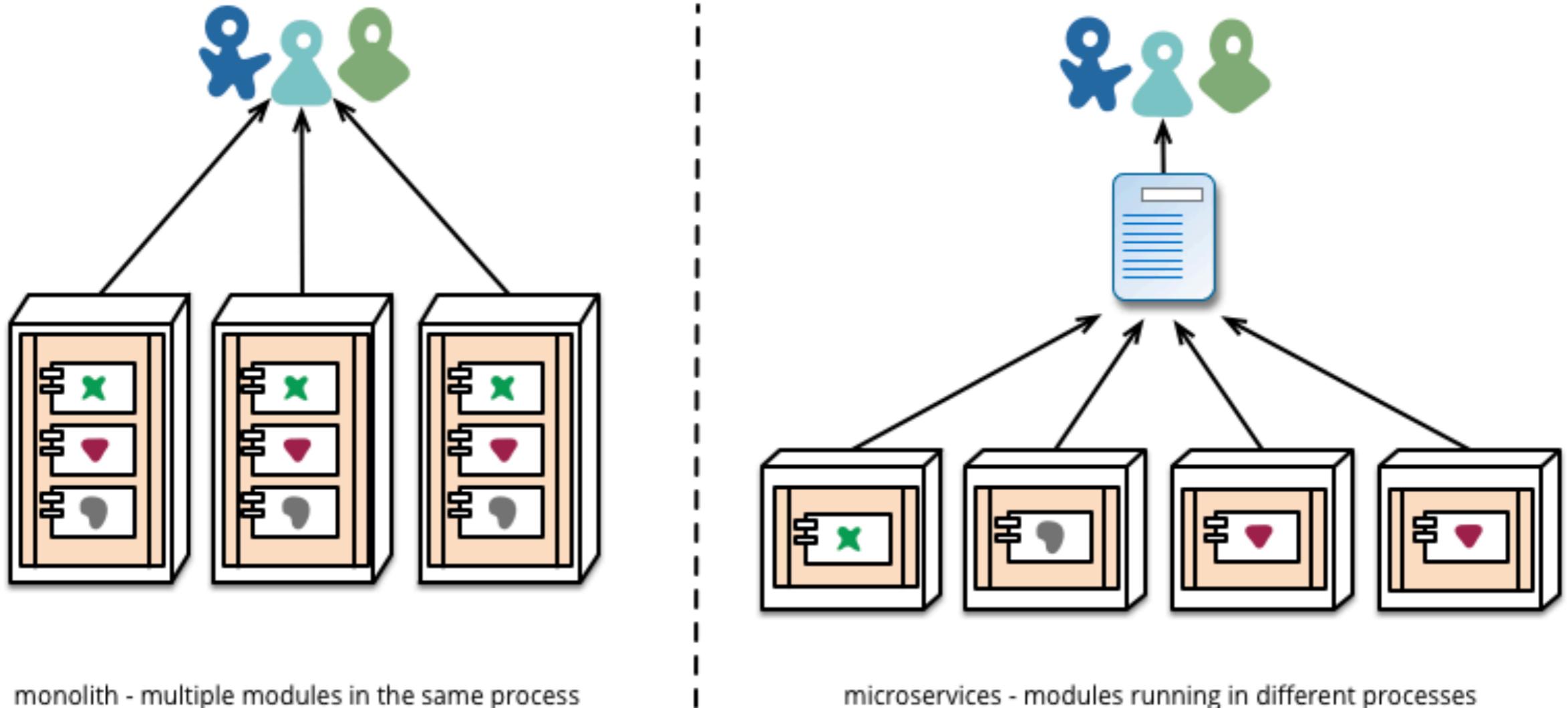


# Microservices

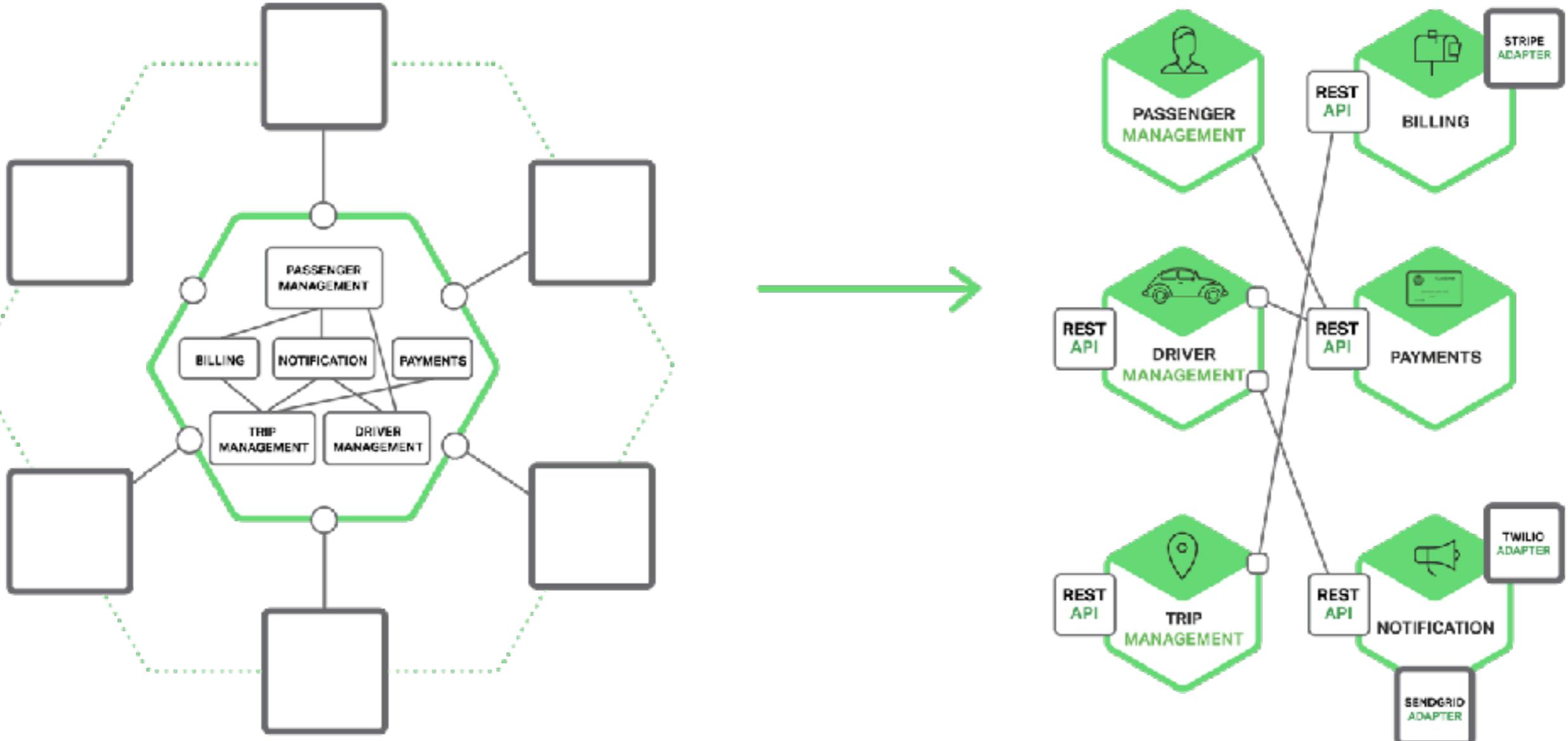
Small, Do one thing  
Modular  
Easy to deploy  
Scale independently



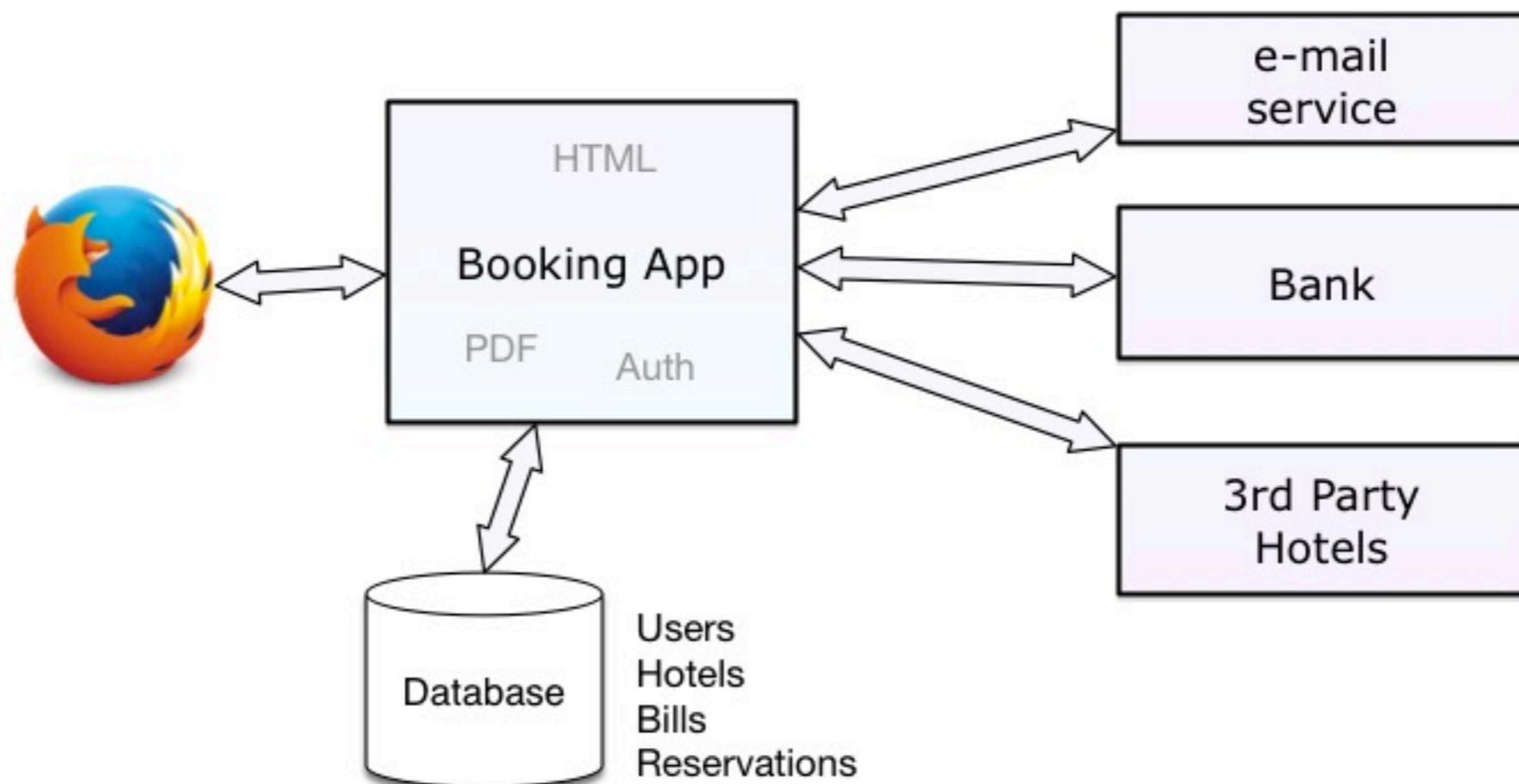
# Conway's Law



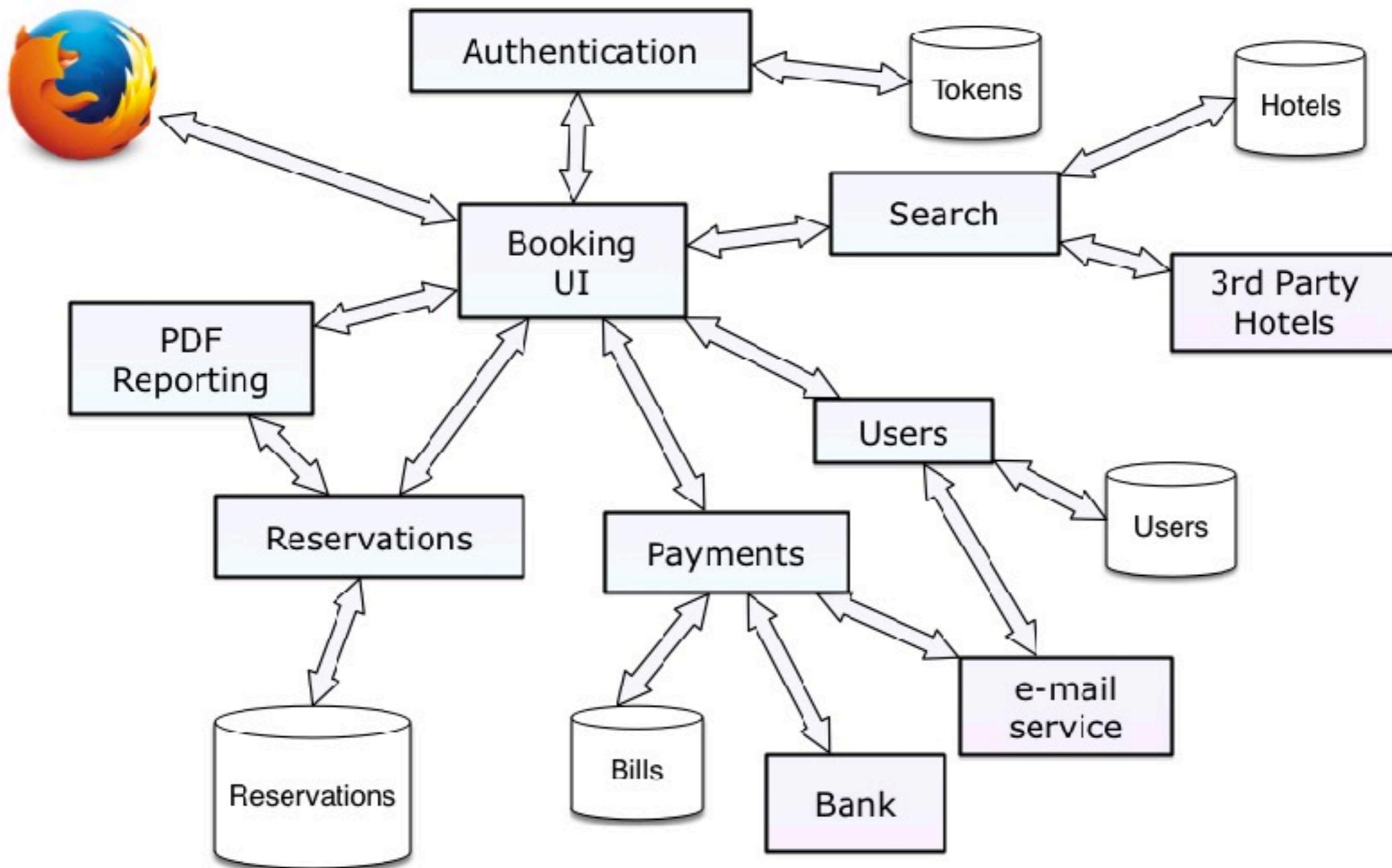
# Conway's Law



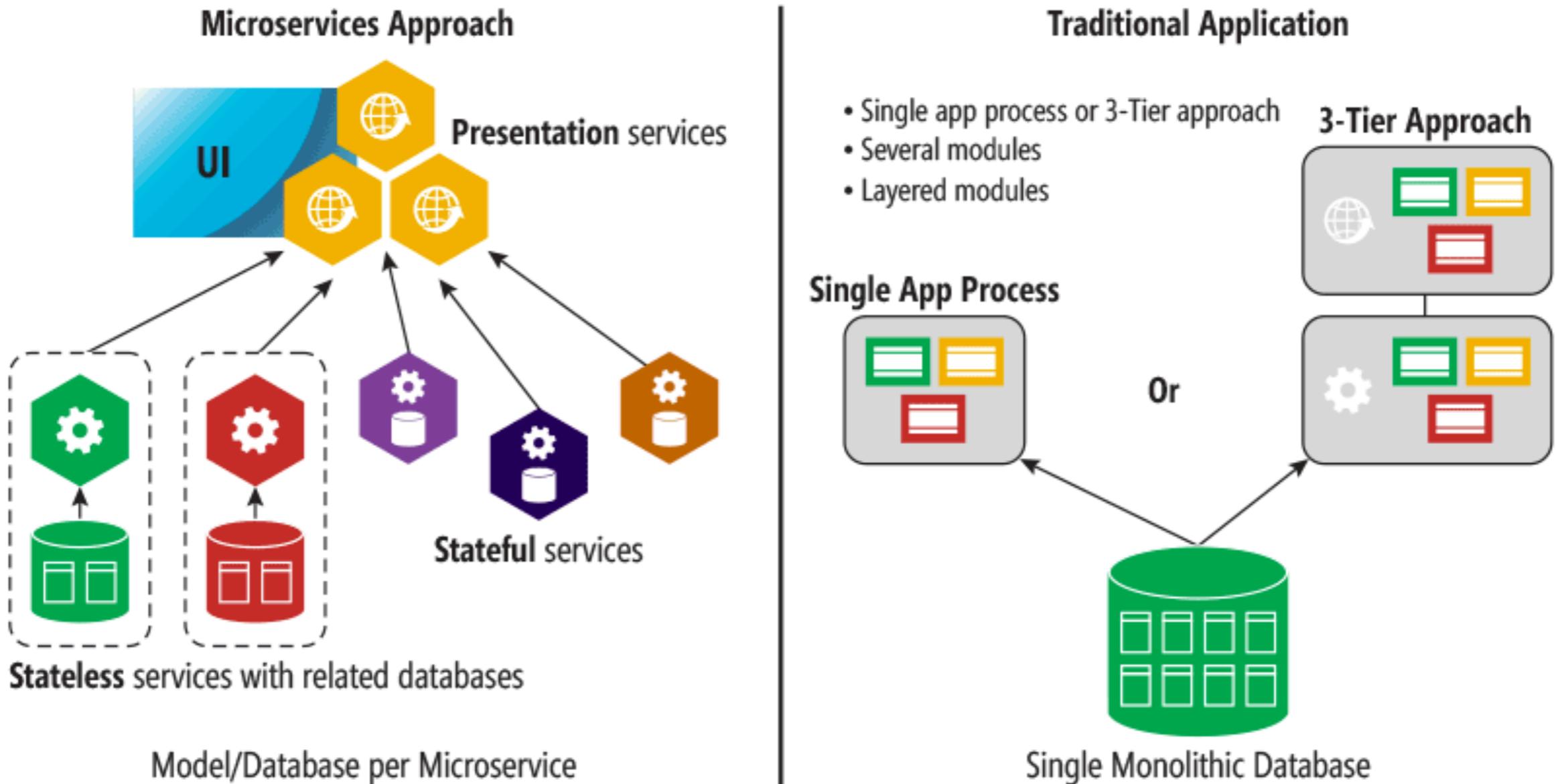
# Monolithic



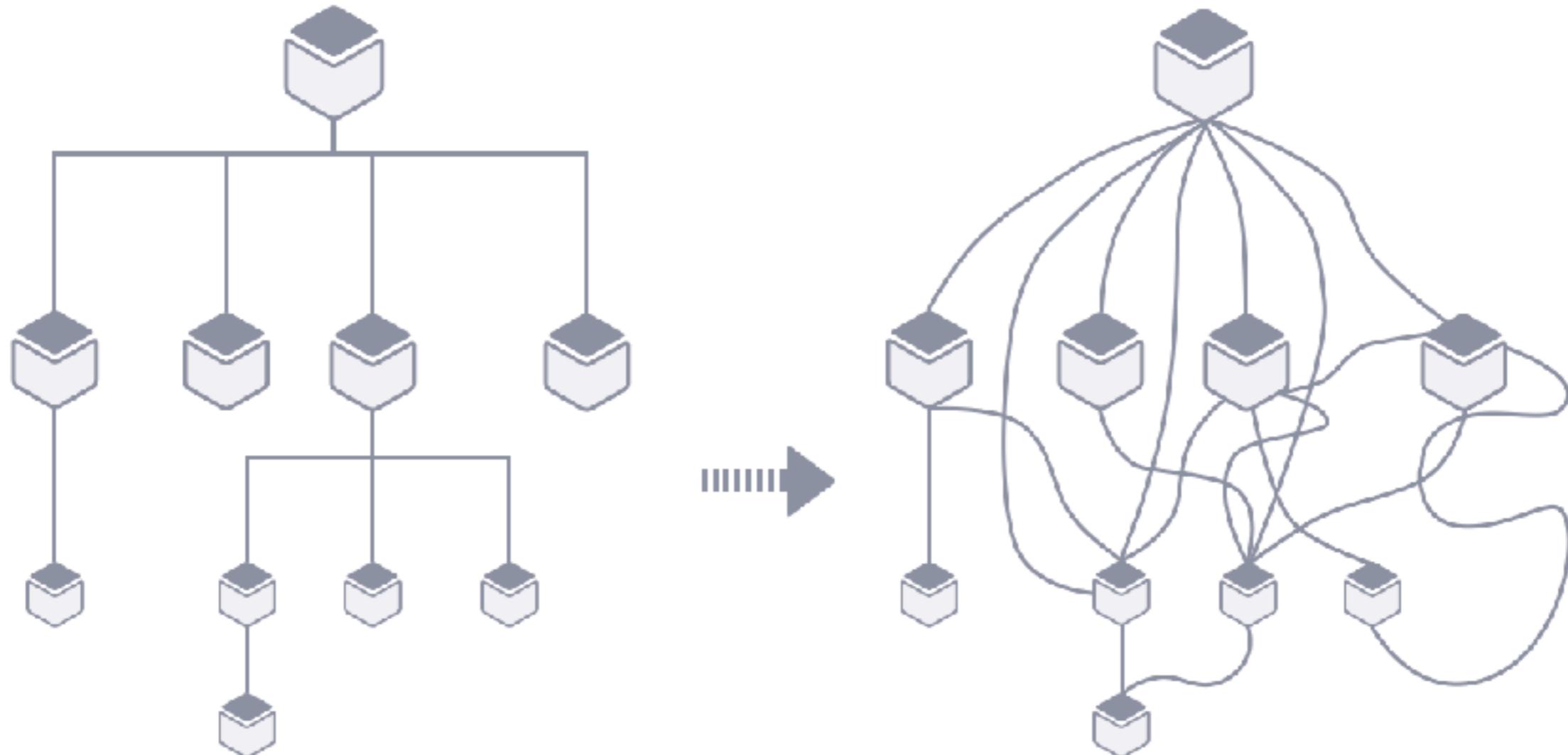
# Microservices



# Microservices

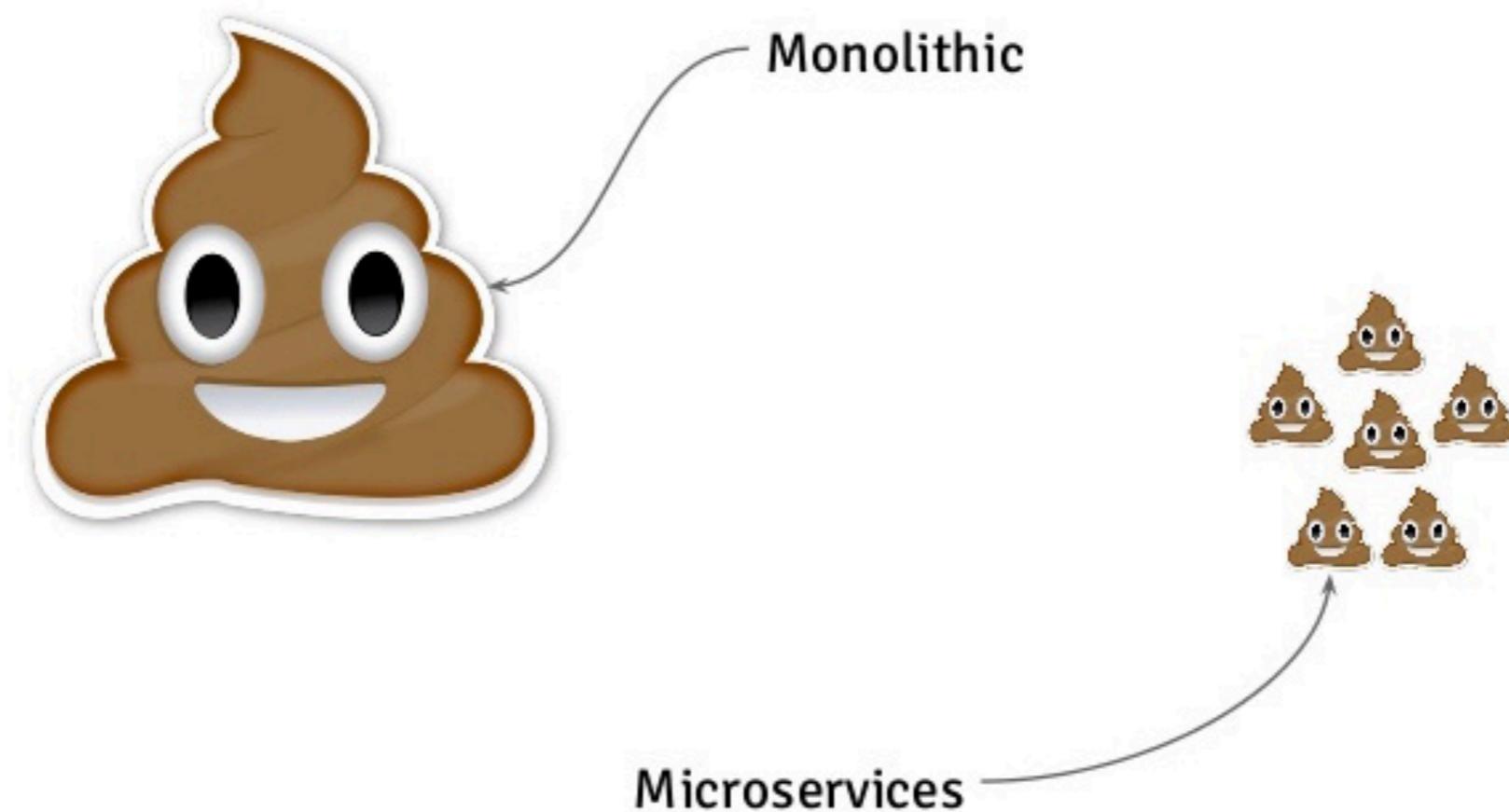


# Microservices spaghetti



# Microservices spaghetti

## Monolithic vs Microservices

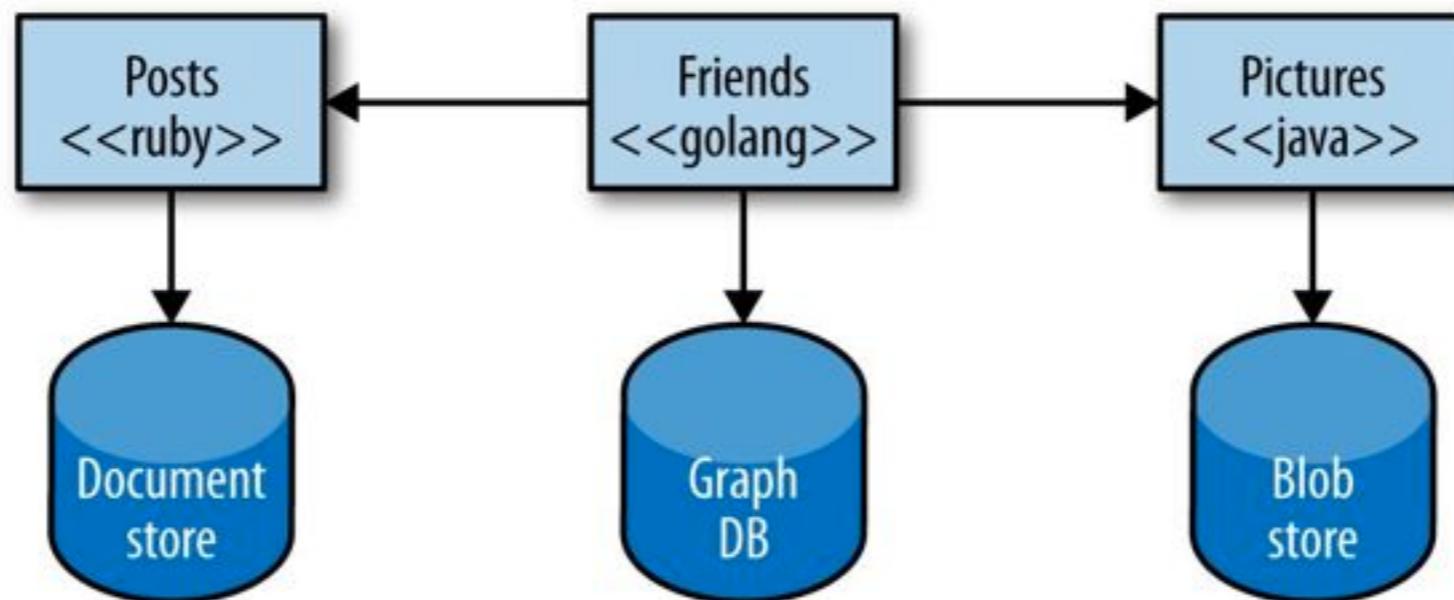


# Key Benefits

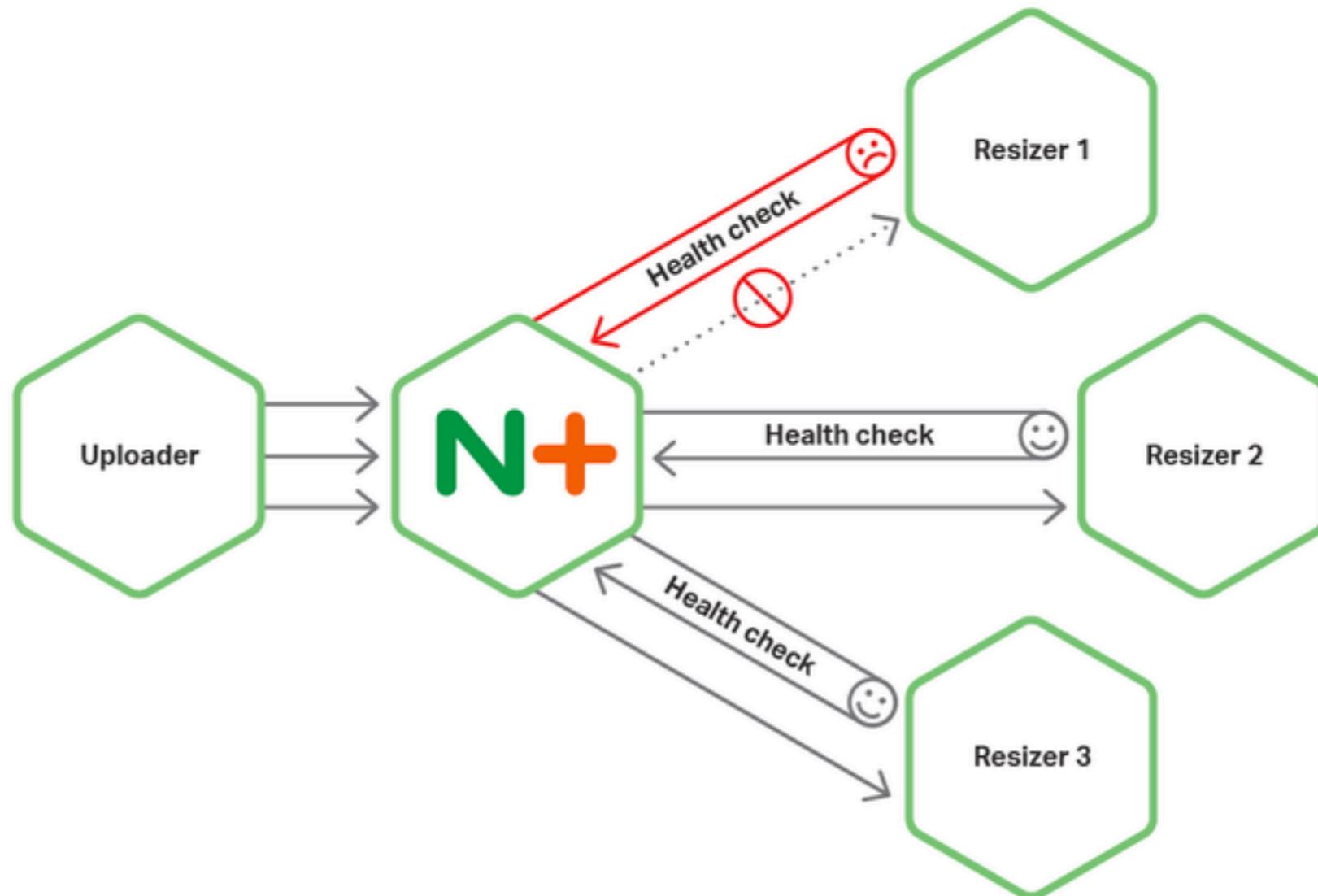


# 1. Technology heterogeneity

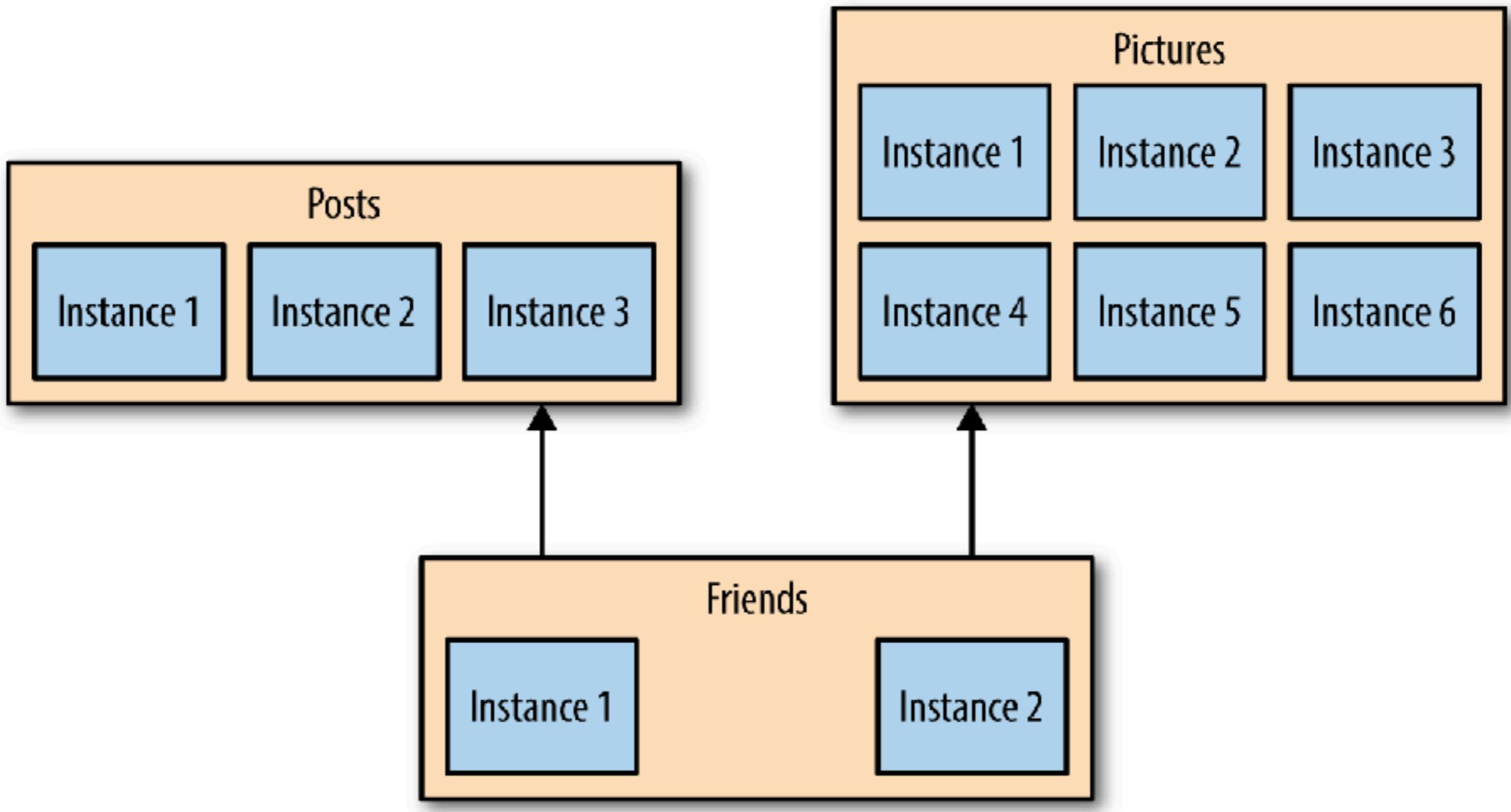
The right tool for each job



# 2. Resilience

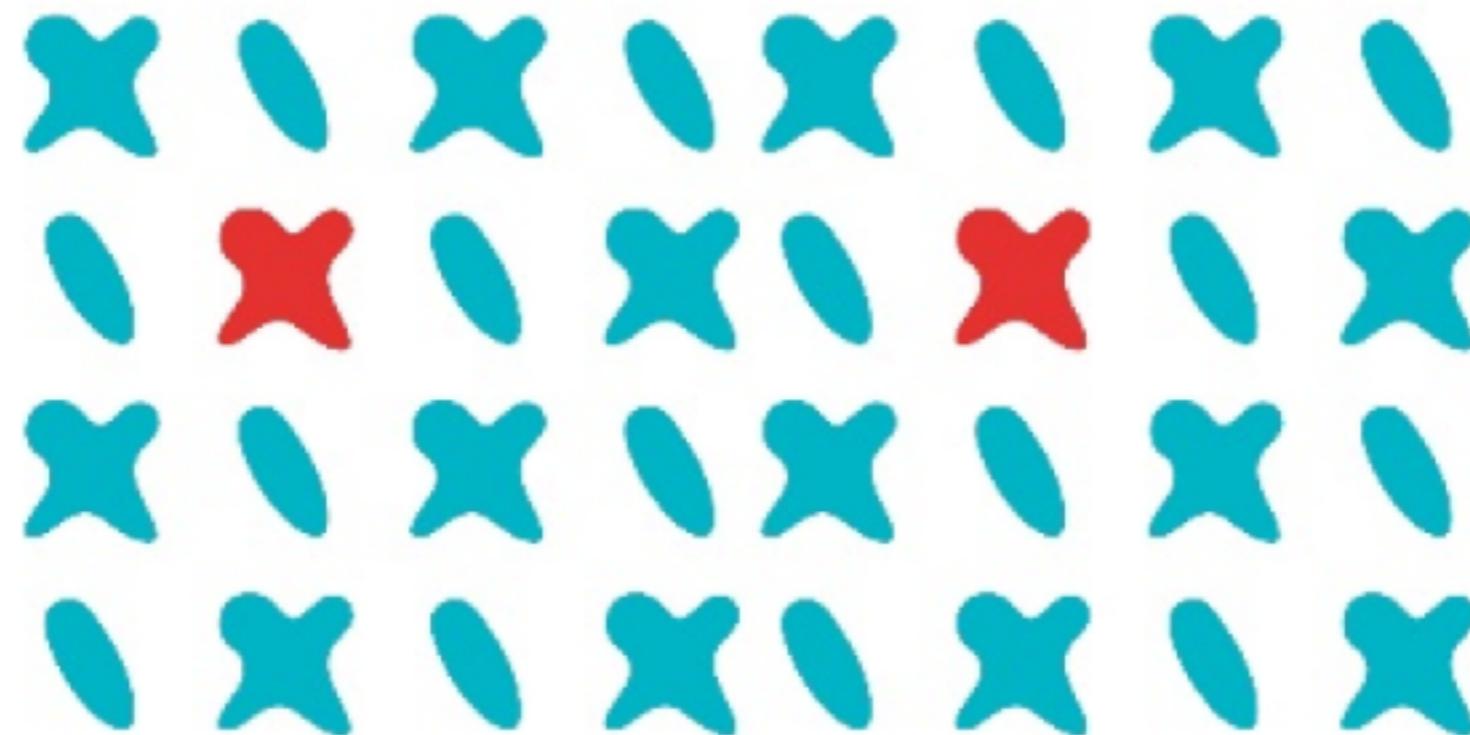


# 3. Scaling



# 4. Ease of deployment

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

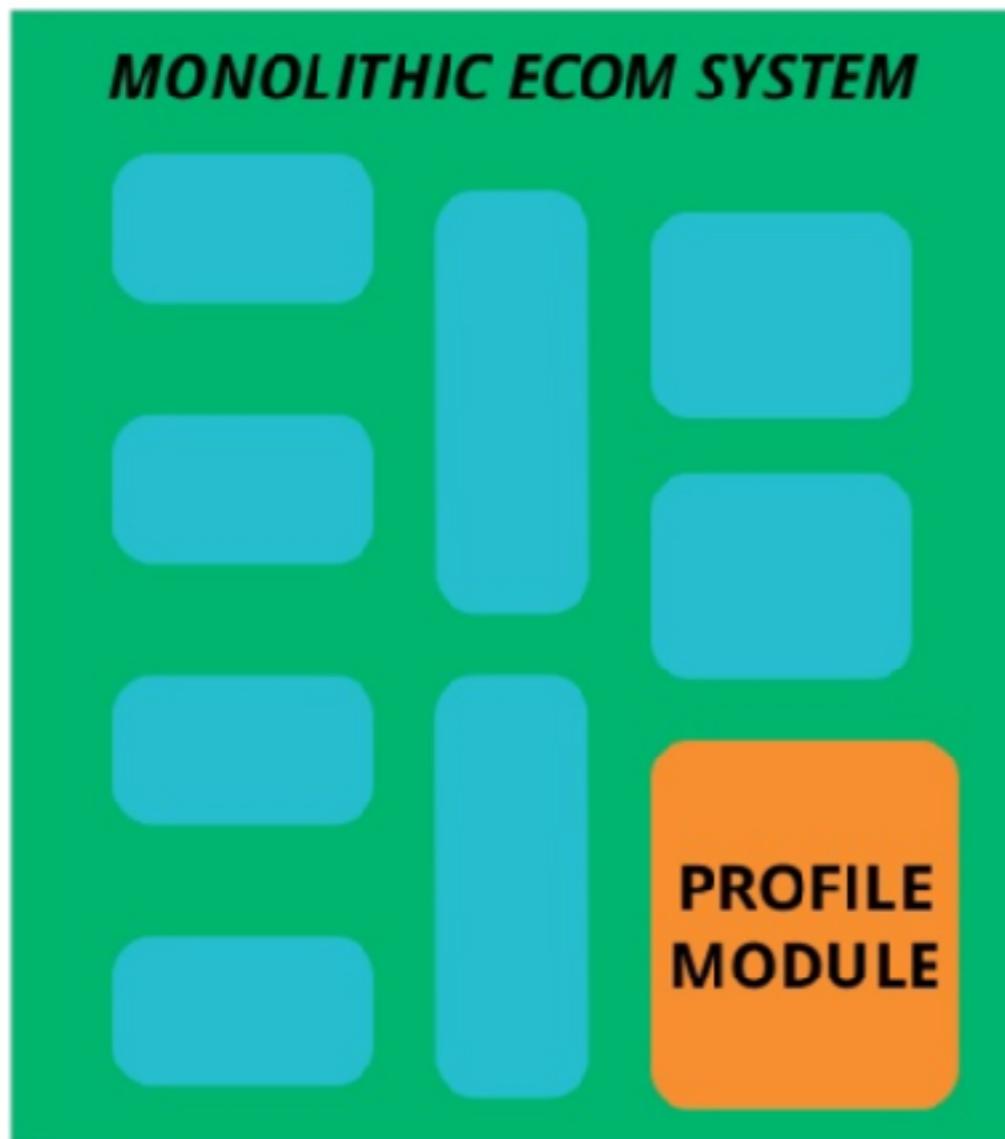


# 5. Organization alignment

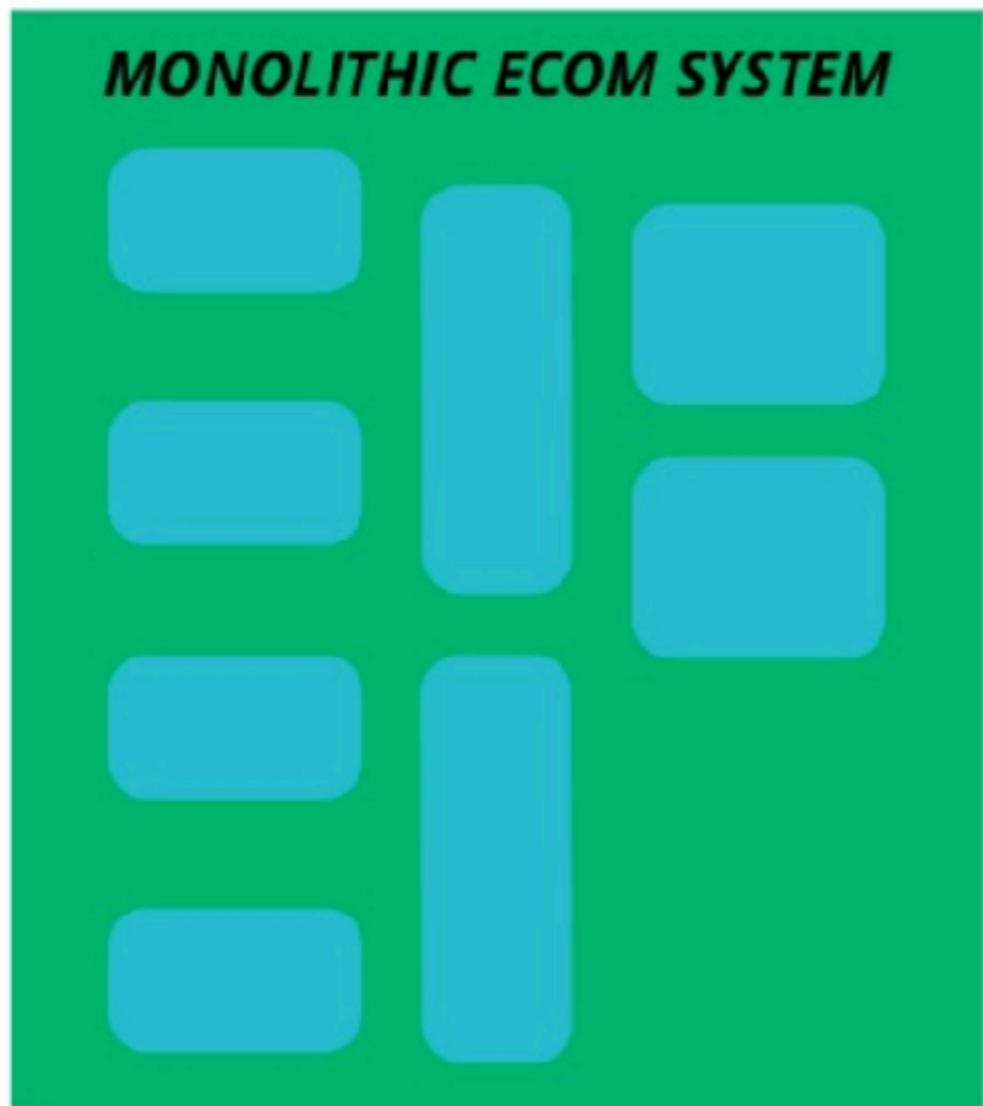
Small teams and smaller codebases



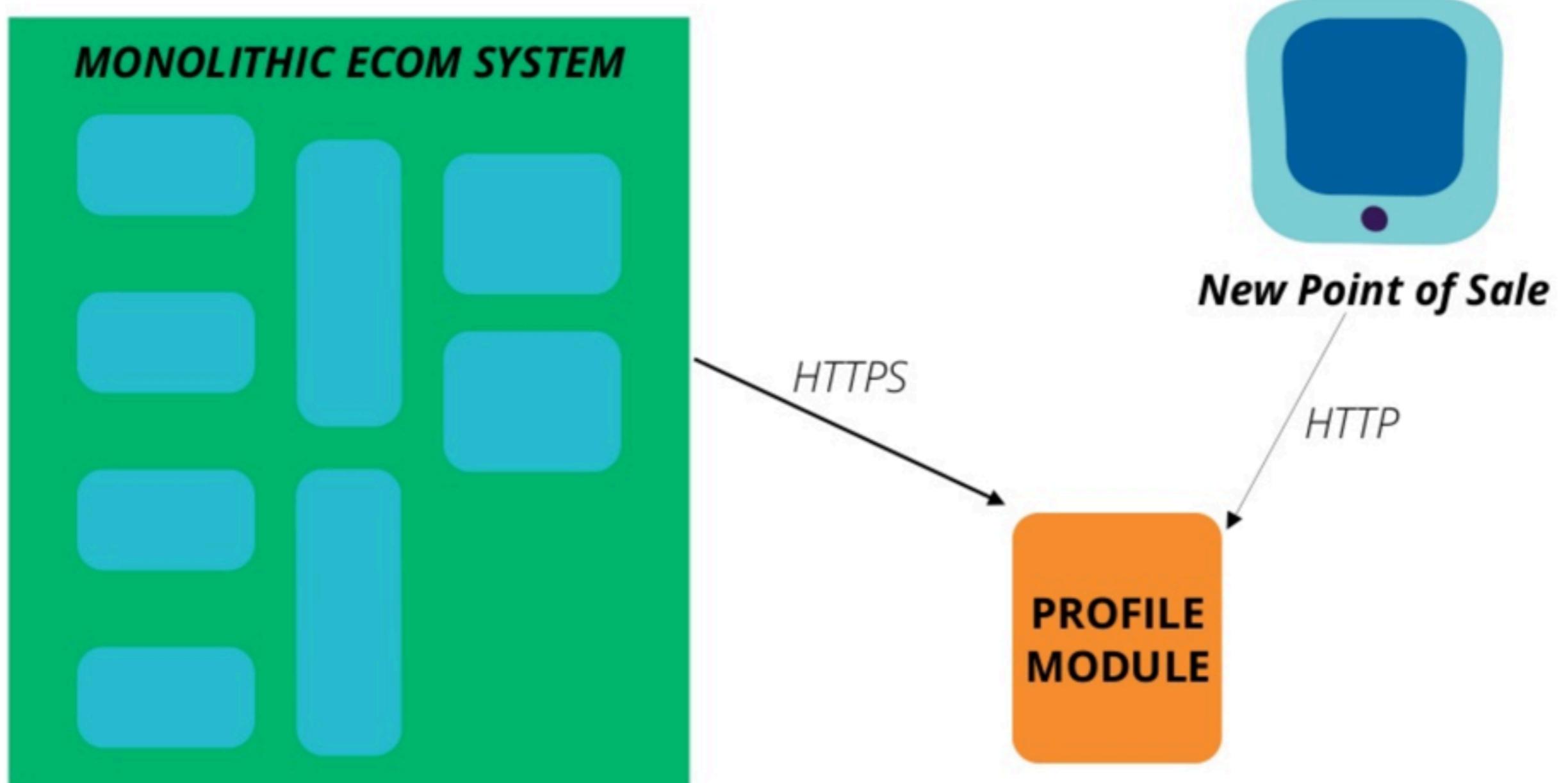
# 6. Composability and replaceability



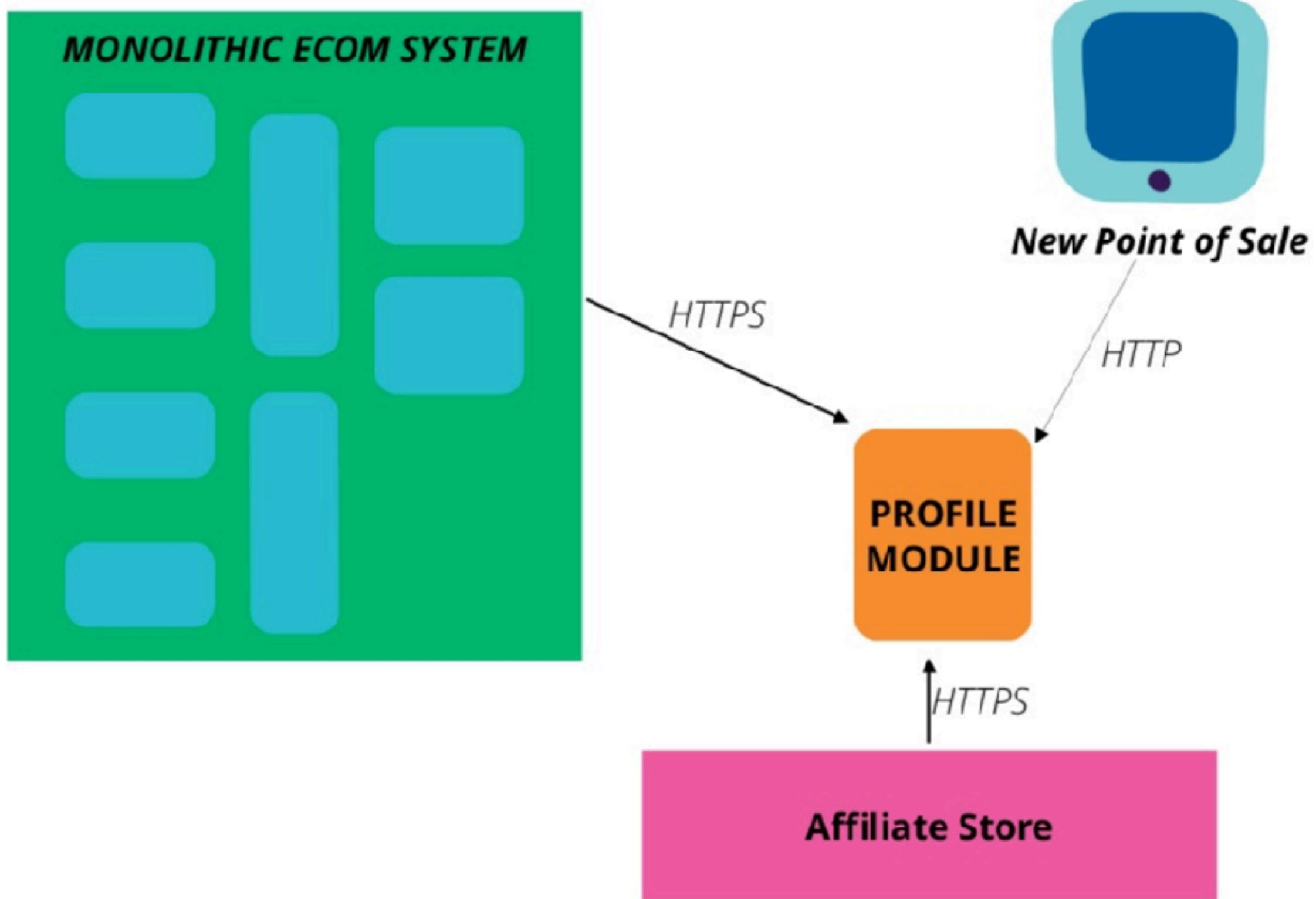
# 6. Composability and replaceability



# 6. Composability and replaceability



# 6. Composability and replaceability



# Characteristics



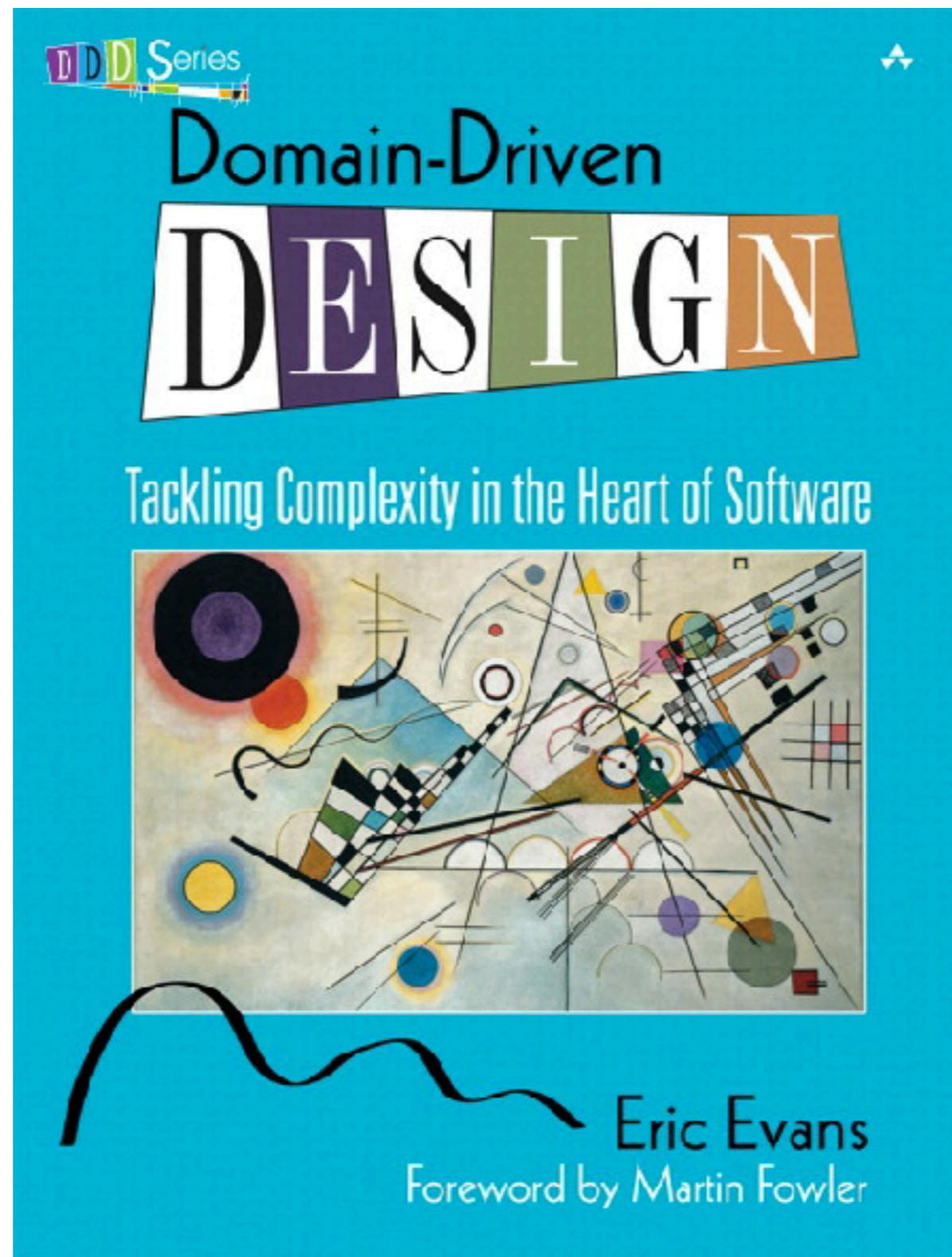
# 1. Responsible for a single capability



# Types of capabilities

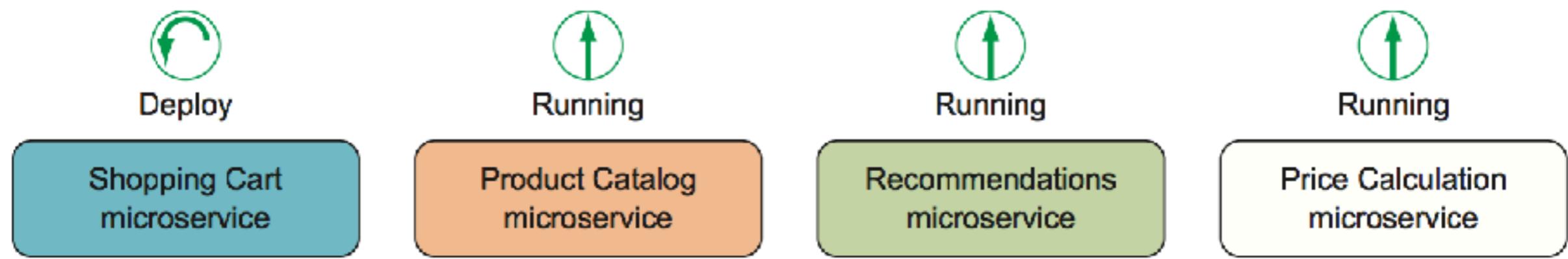
Business capability  
Technical capability





## 2. Individually deployable





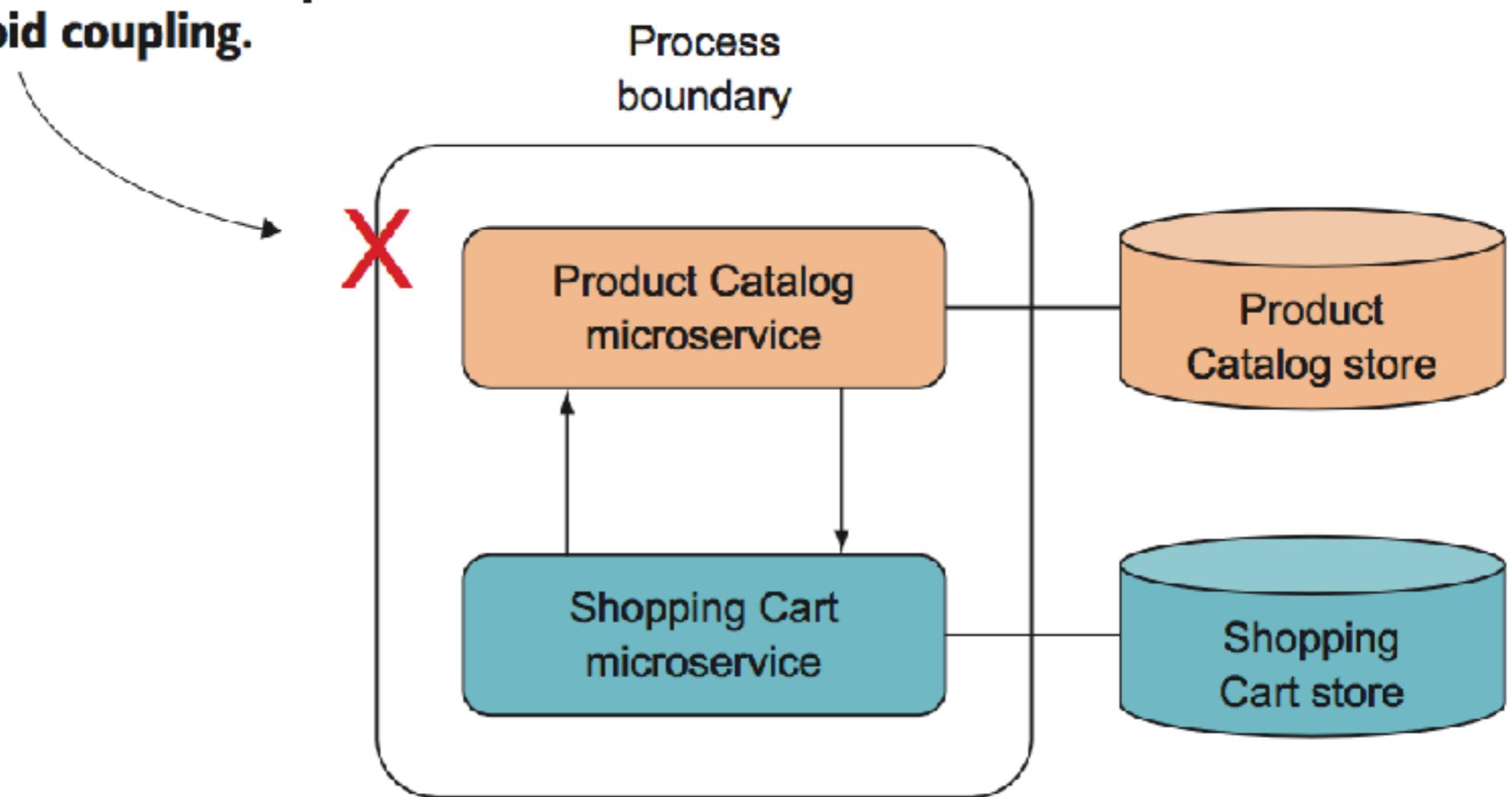
**Figure 1.2 Other microservices continue to run while the Shopping Cart microservice is being deployed.**



**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.

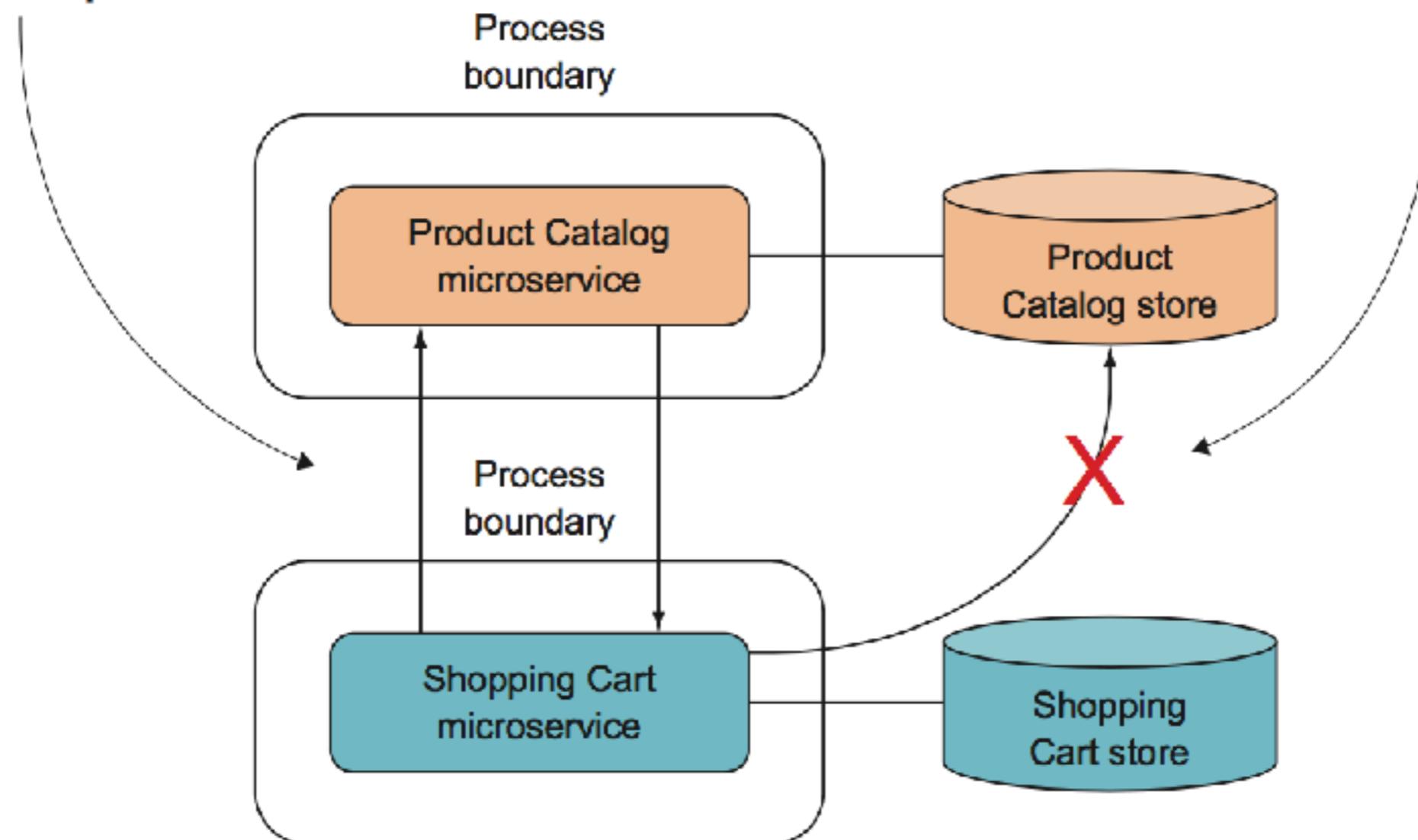


# 4. Own data store



**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.**



# 5. Small team can maintain



# 6. Replaceable



# Enabled system

Flexible  
Scalable  
Resilient



# **Challenges with Microservices ?**



# 1. How to define the boundaries of each microservices ?



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



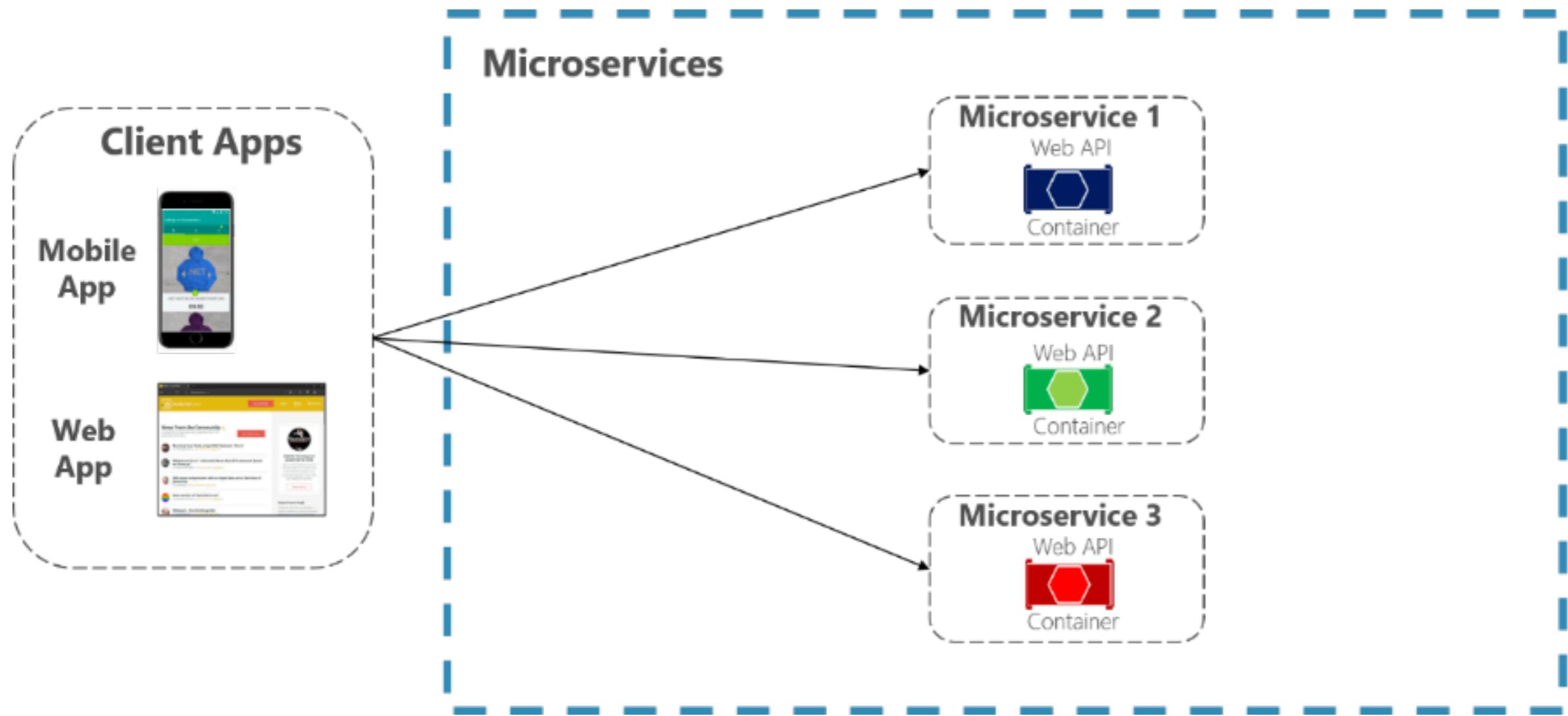
# Popular solutions

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

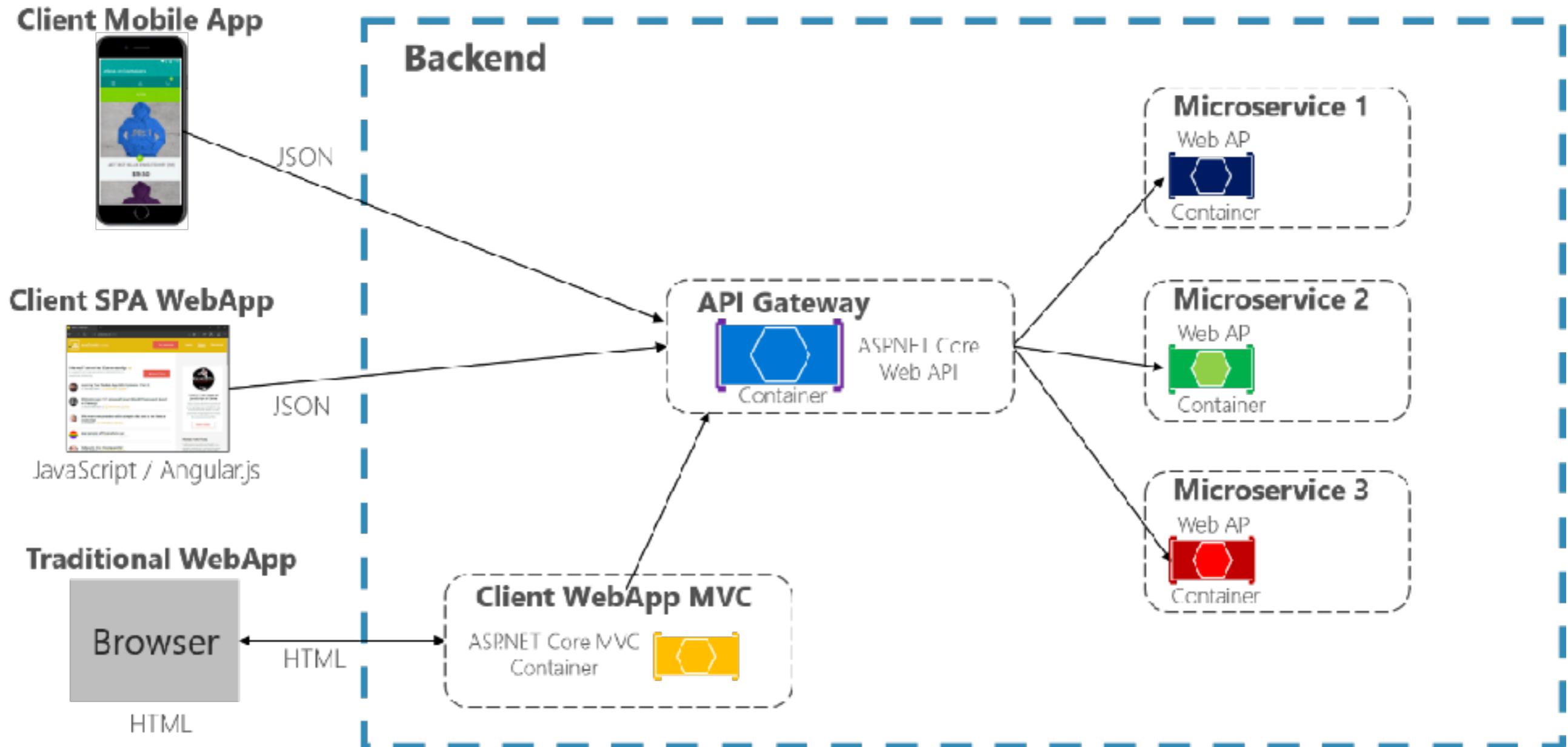


# Direct Client-To-Microservice communication

## 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

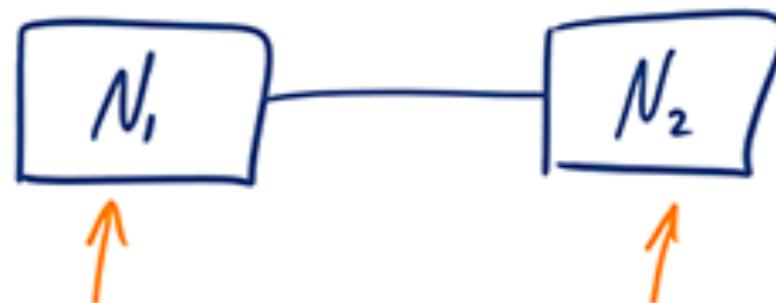


# CAP Theorem

**Consistency**



**Availability**



**Partition Tolerance**



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



# 4. How to design communication across microservices boundaries ?



# Protocols

HTTP and REST  
AMQP  
Messaging



# Communication

Request-Response model  
Observer model

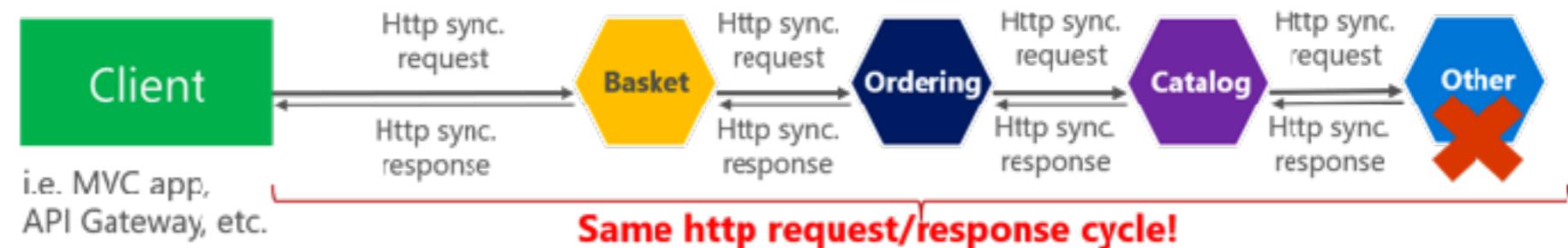


# Communication

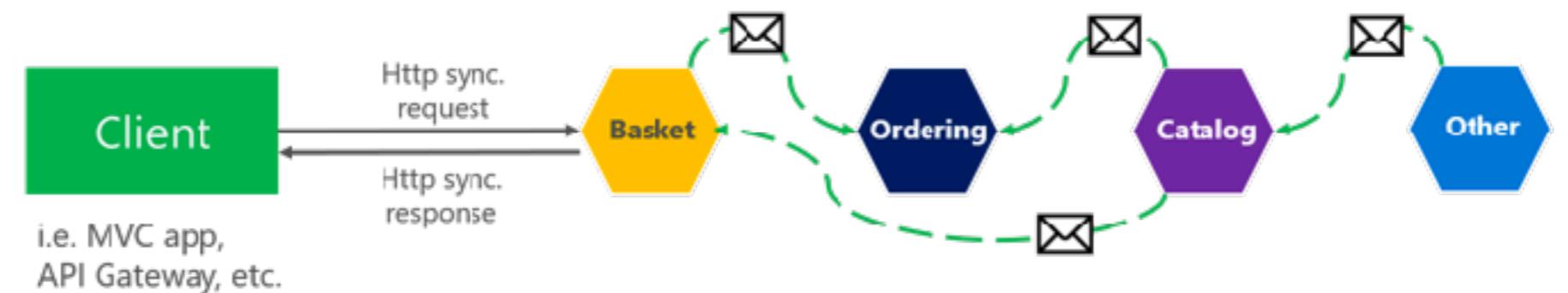
Synchronous vs. async communication across microservices

## Anti-pattern

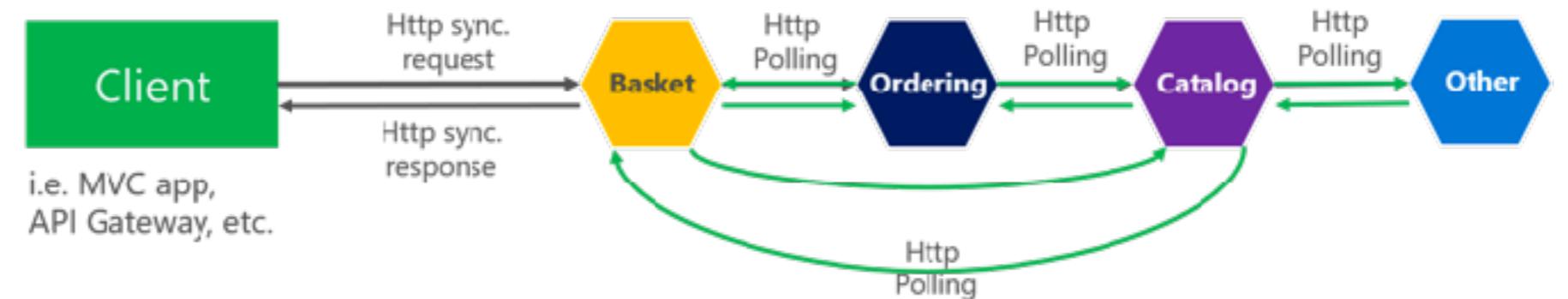
**Synchronous**  
all req./resp. cycle



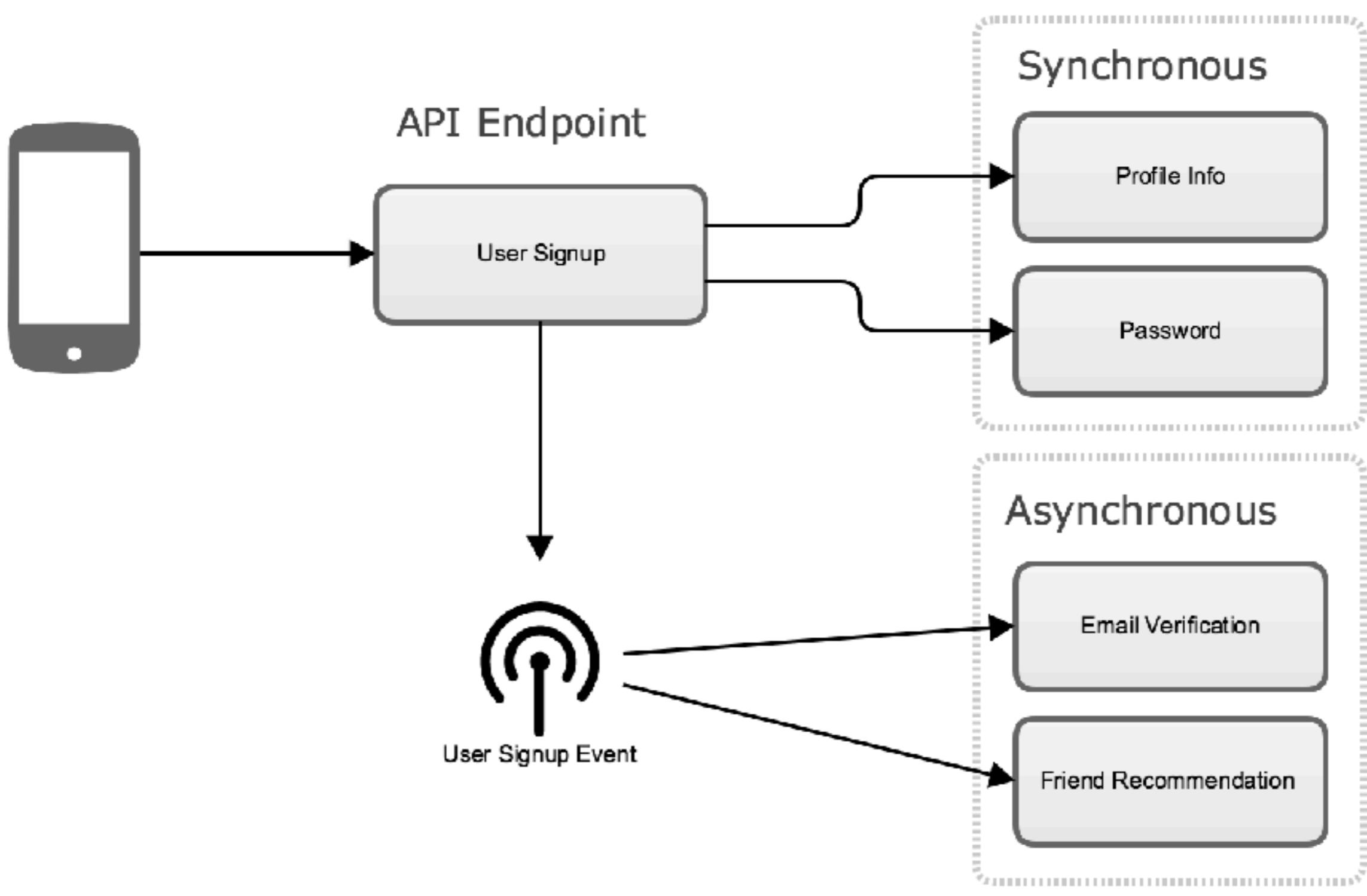
**Asynchronous**  
Comm. across  
internal microservices  
(EventBus: i.e. **AMQP**)



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

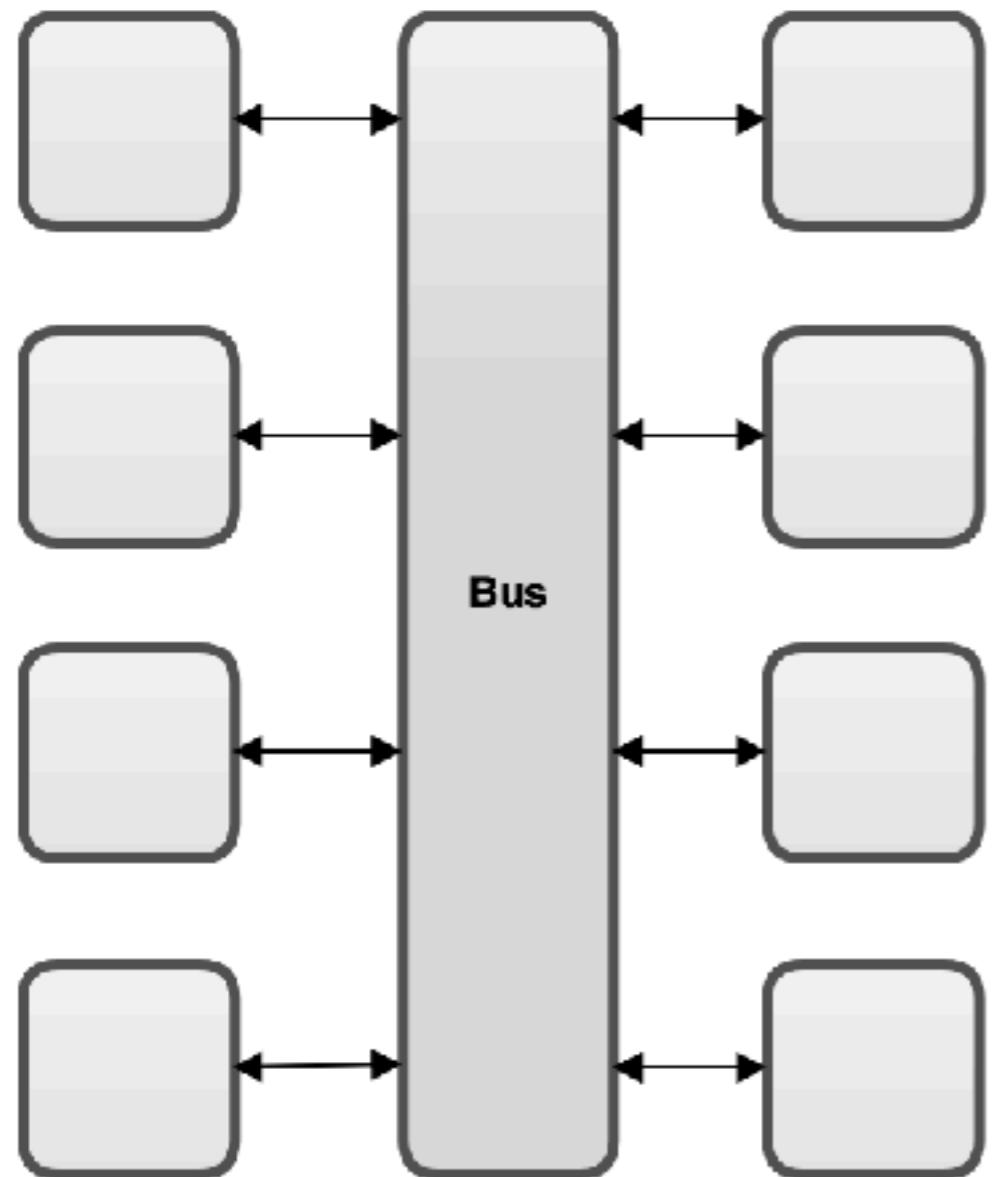


# Communication

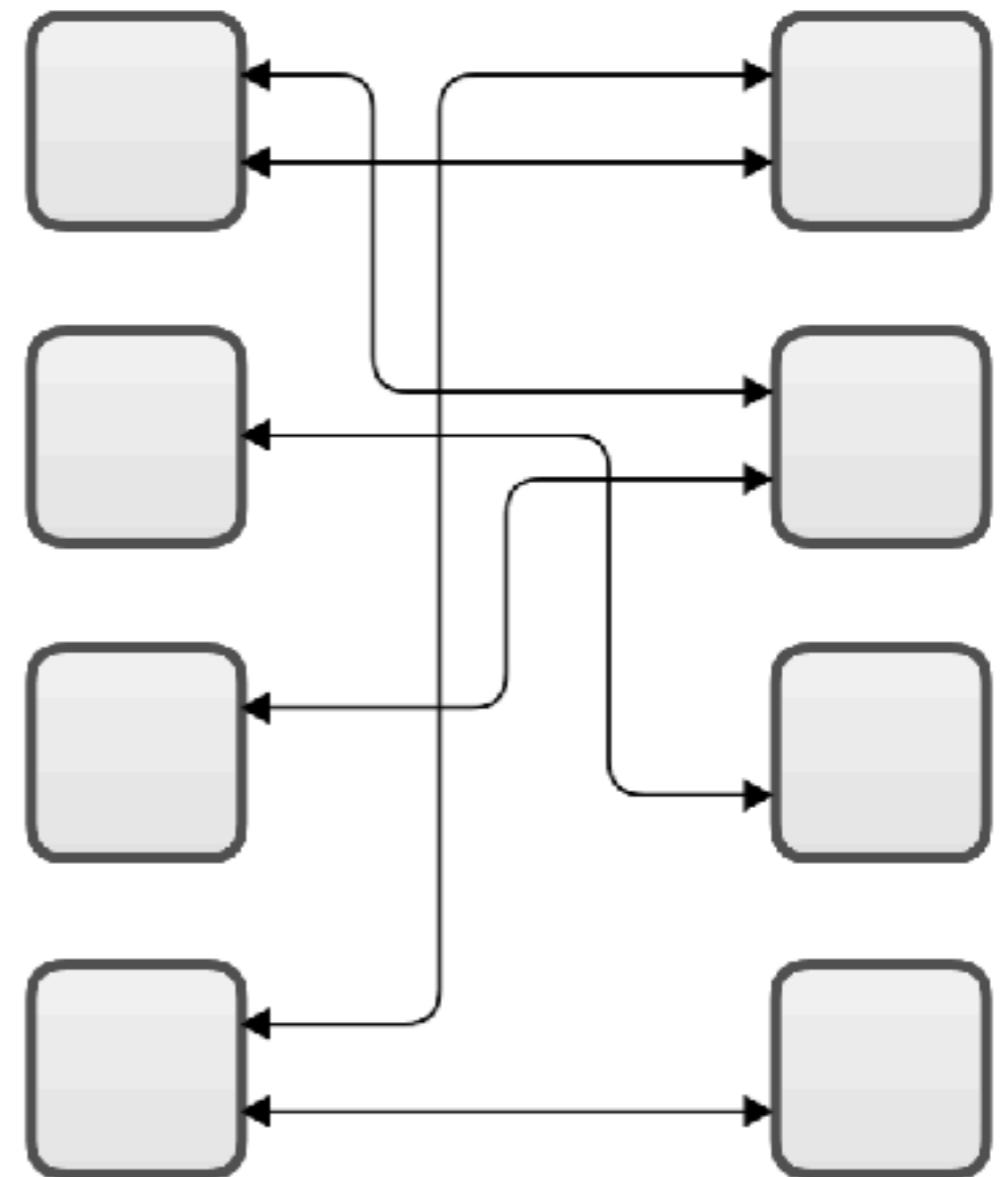


# Anti-pattern :: centralize bus service

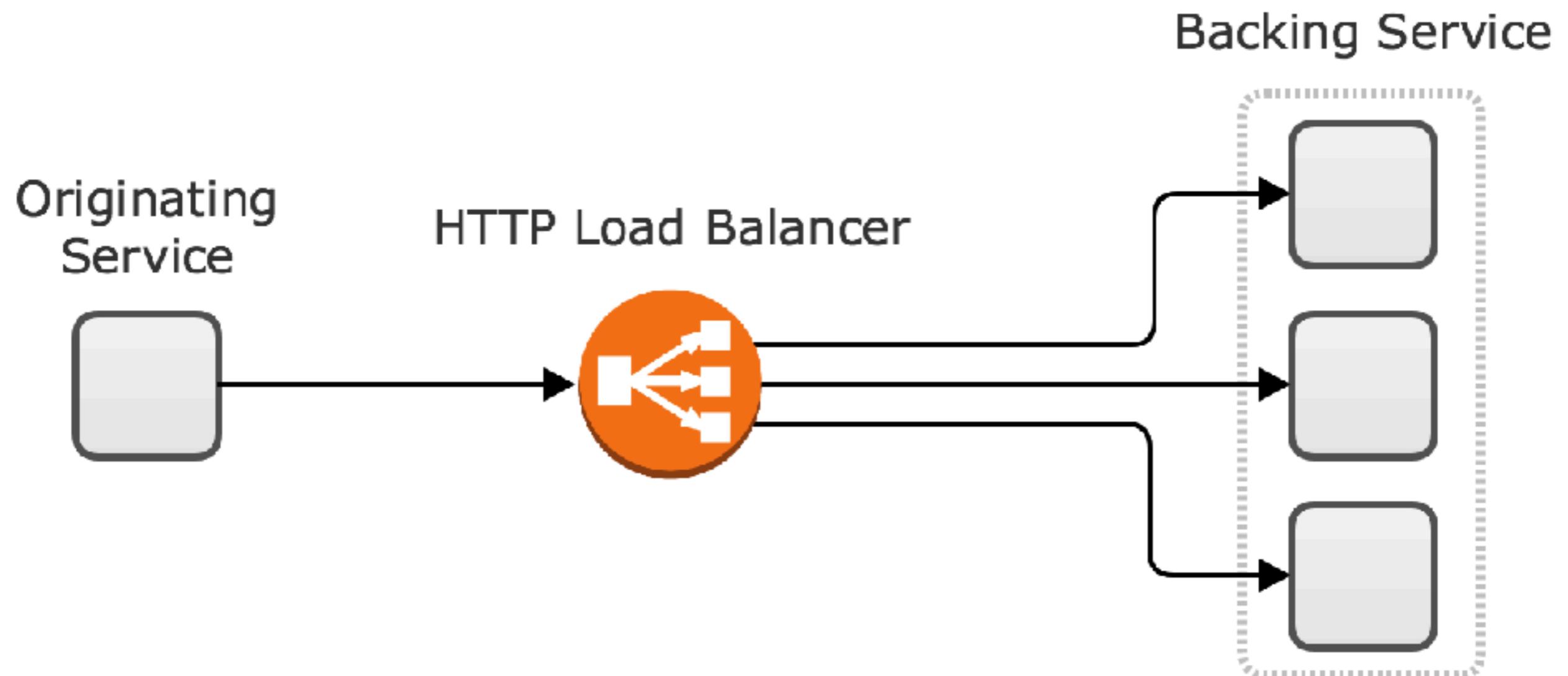
Central Bus



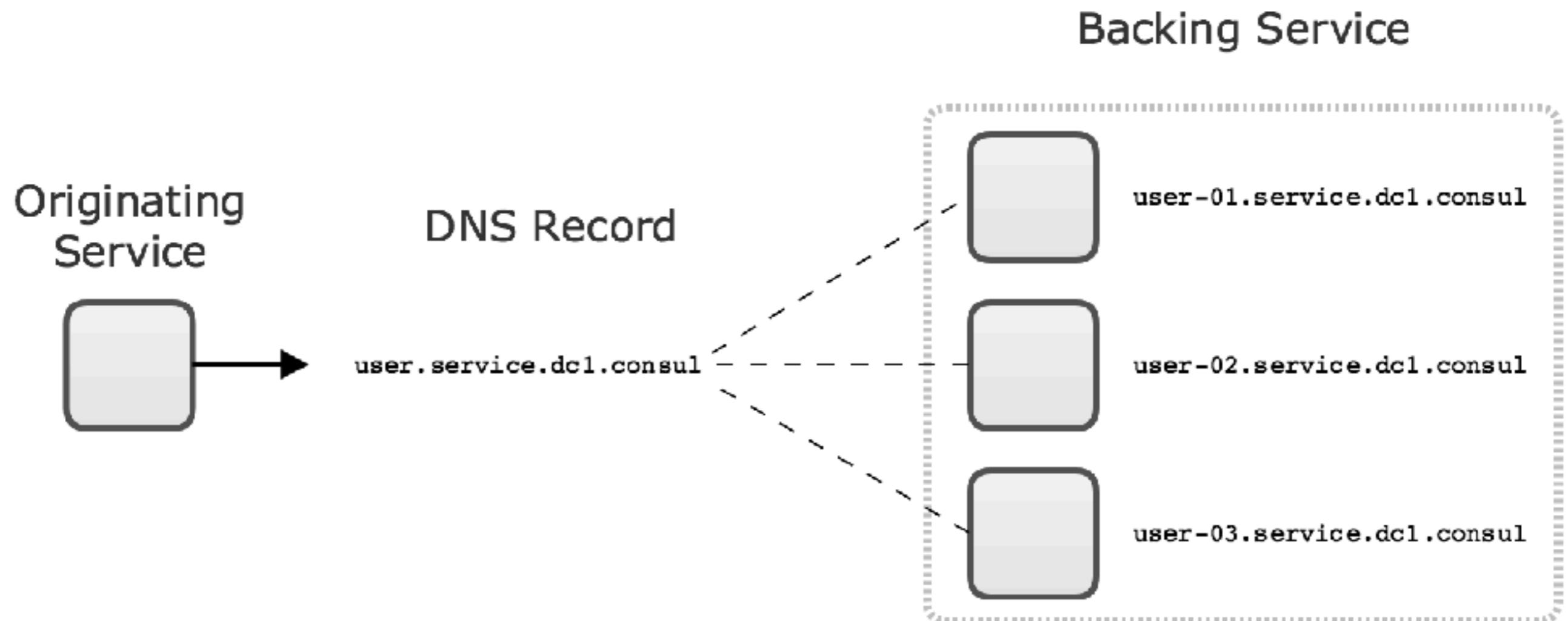
Decentralized



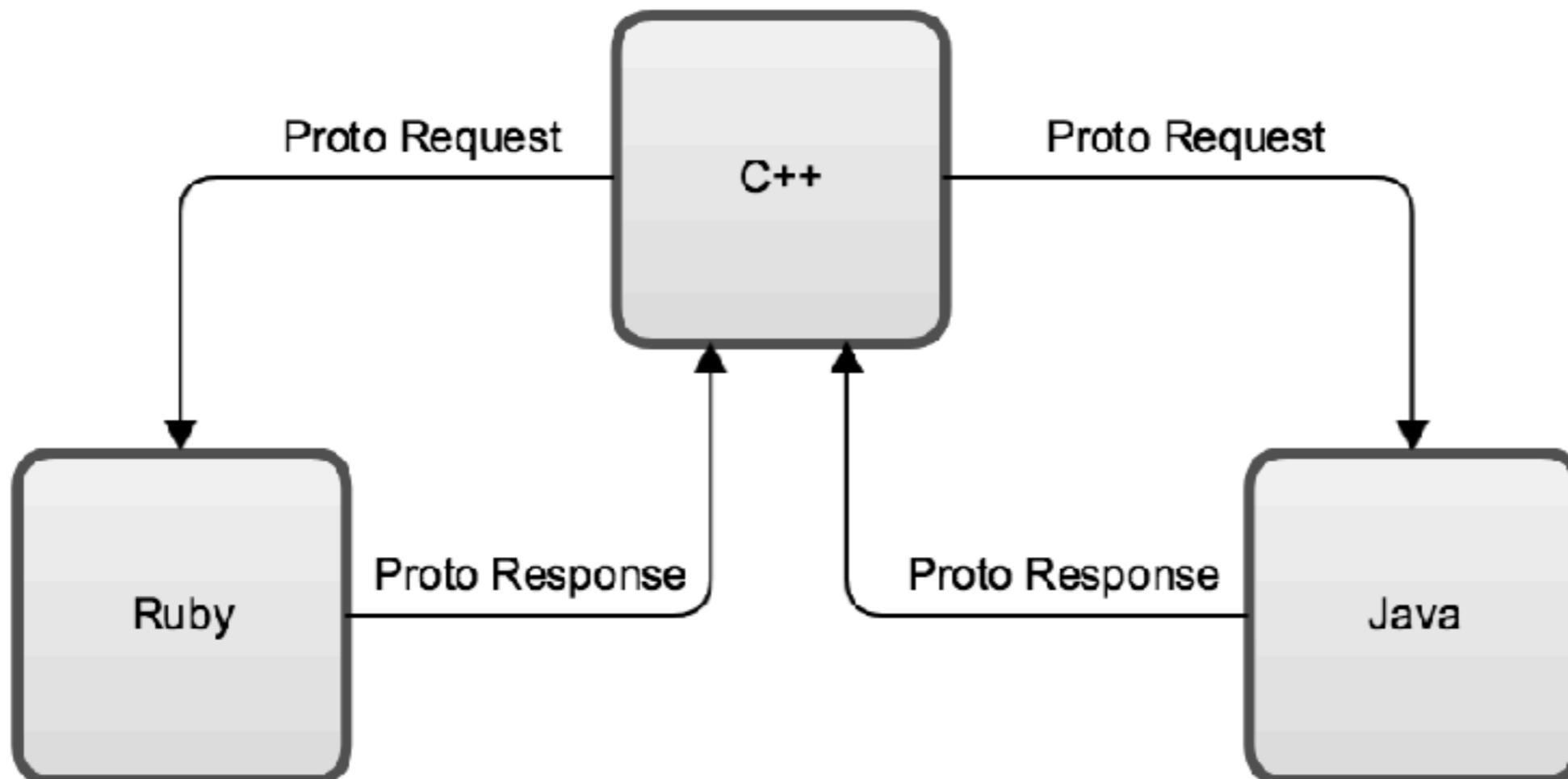
# Request-response model



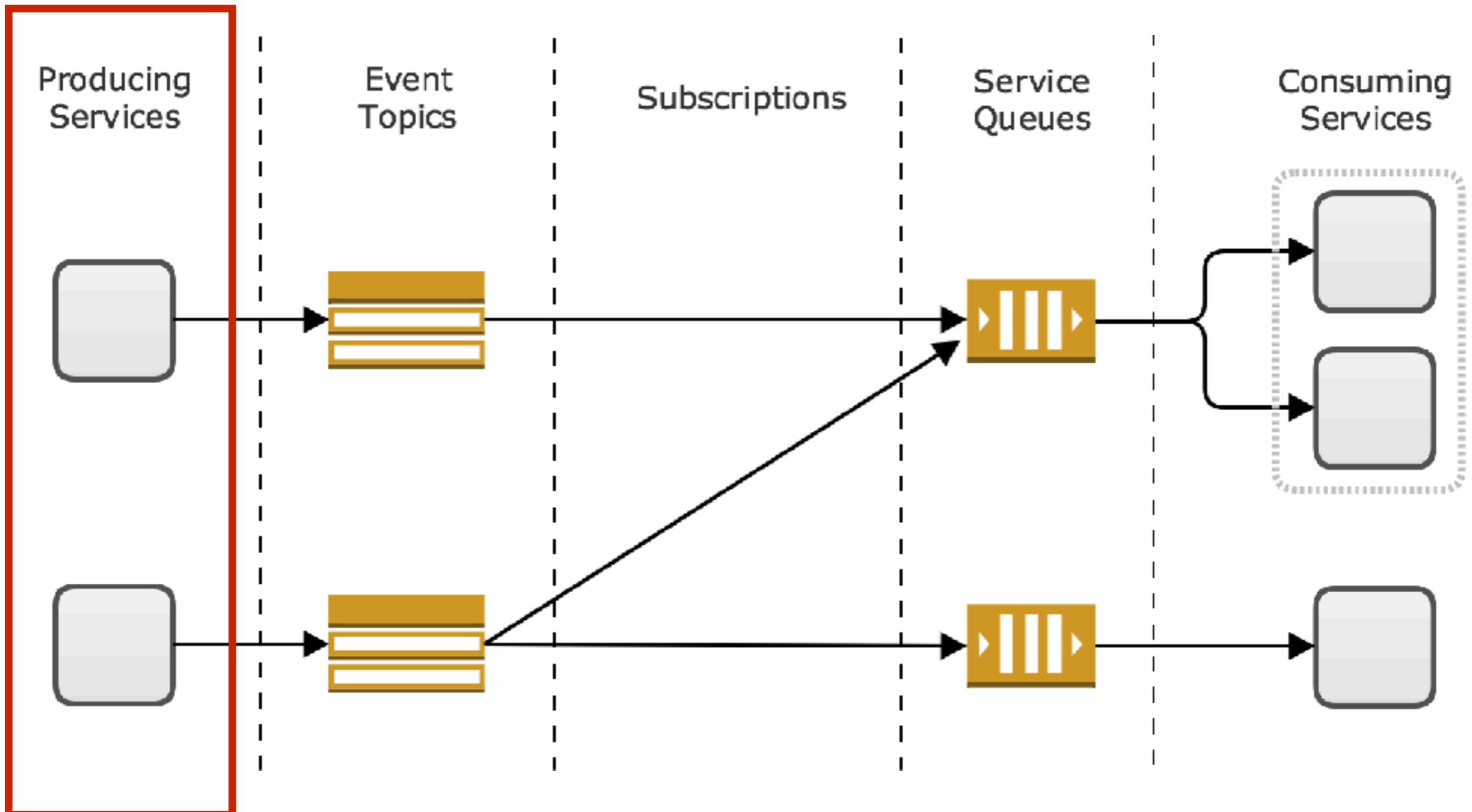
# Request-response model



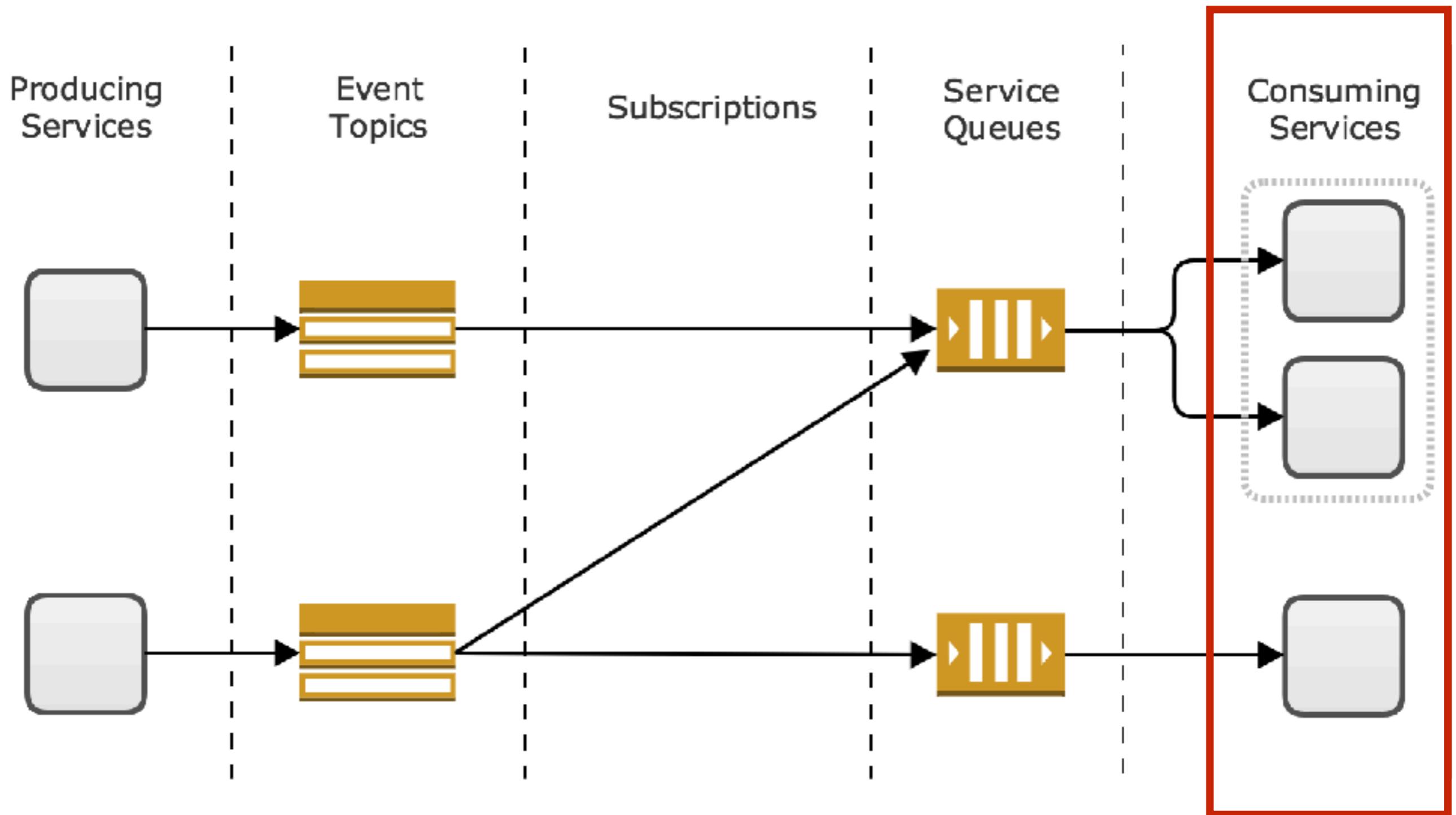
# Request-response model



# Observer model



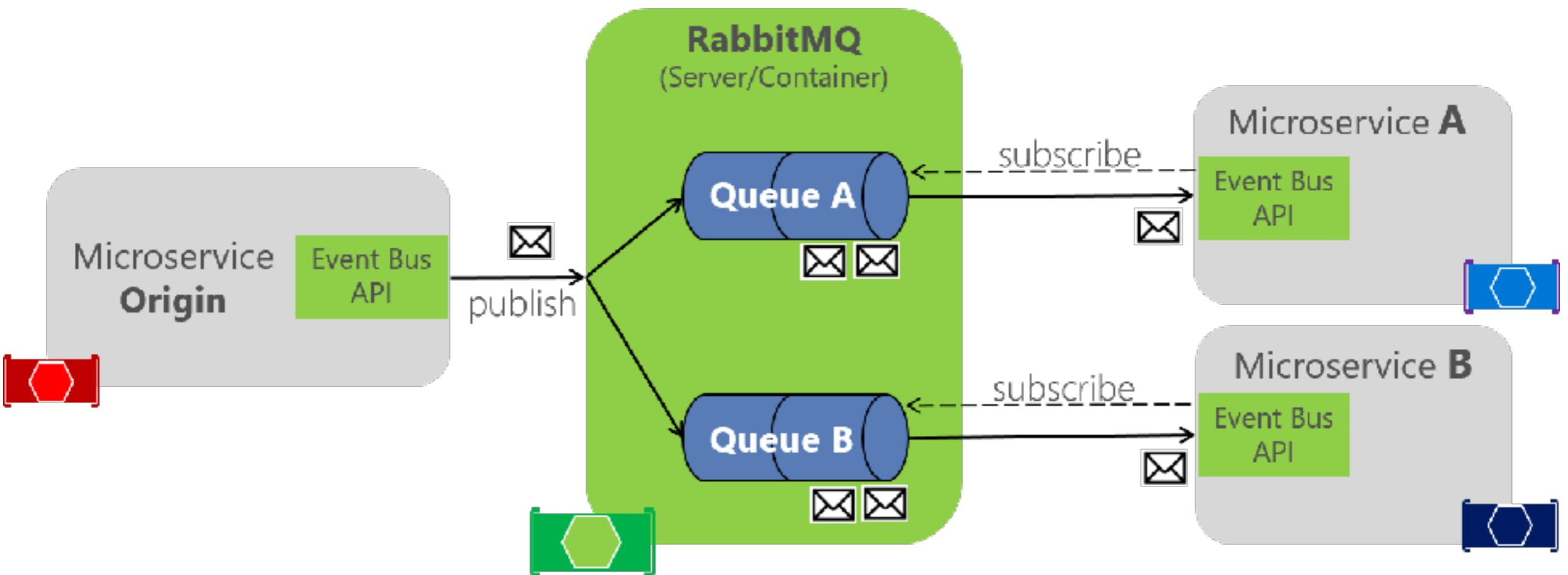
# Observer model



# Observer model

**Message  
Sender**

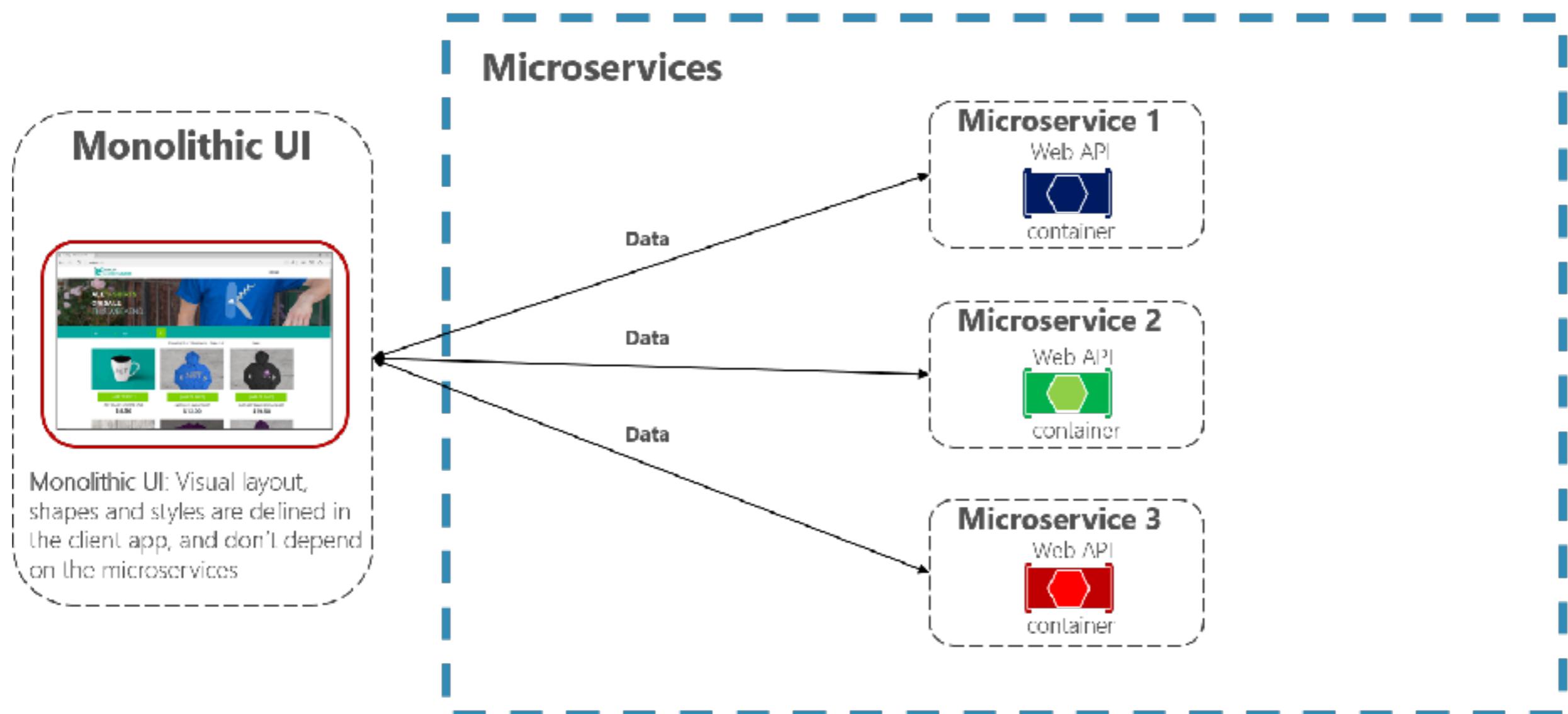
**Message  
Receivers**



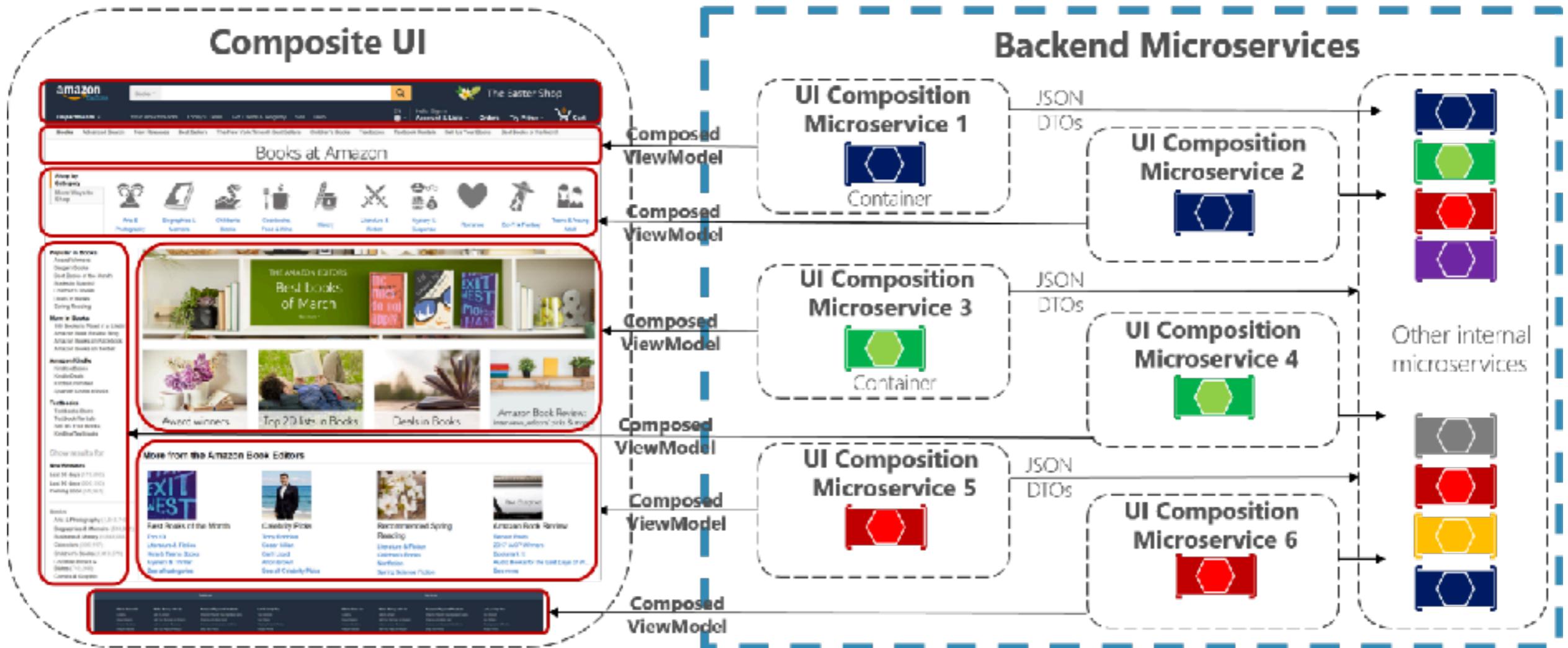
# Integrate with User Interface



# Monolithic UI consuming microservices



# Composite UI generated by microservices



# Microservices pitfalls

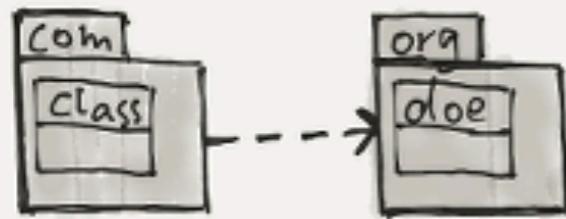
More/Low splitting  
More network interaction  
Data storing and sharing  
Compatibility issues  
Testing  
Operation & Monitoring



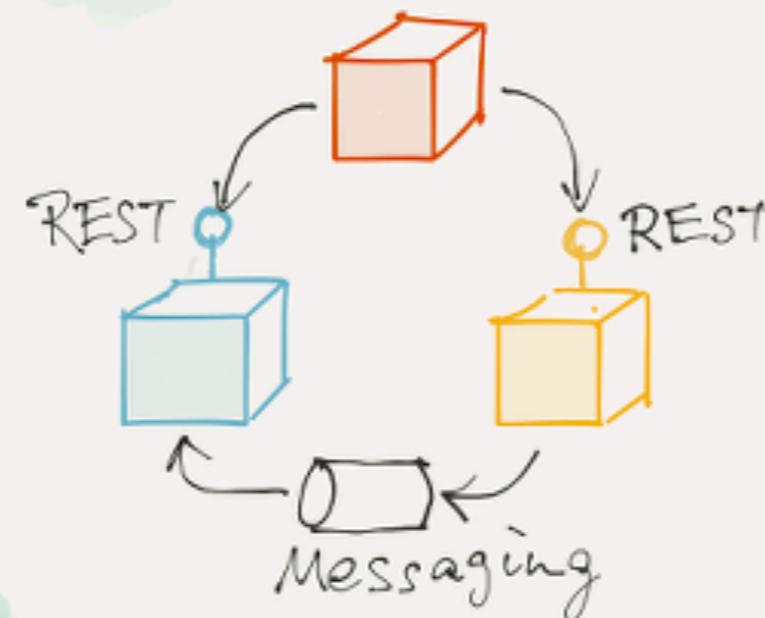




# Architecture



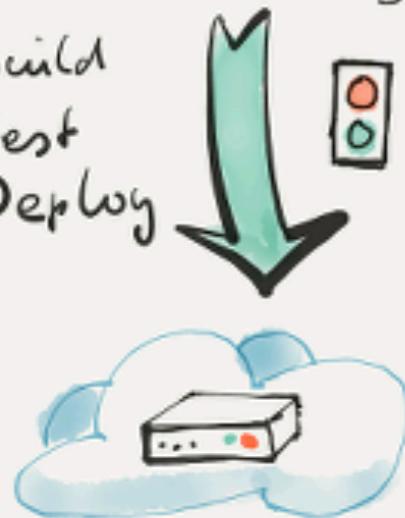
# Microservices



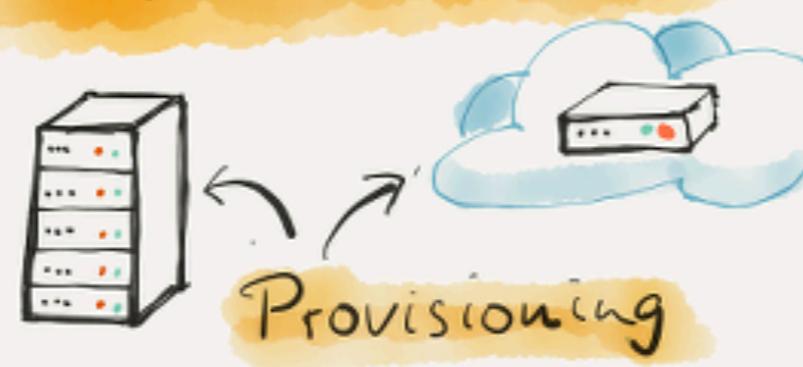
# Deployment

## Continuous Delivery

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



# Infrastructure

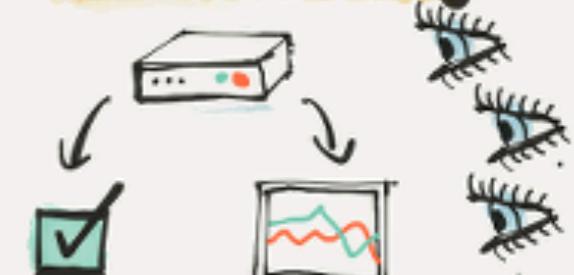


# People & Teams



Communication  
Collaboration

# Monitoring

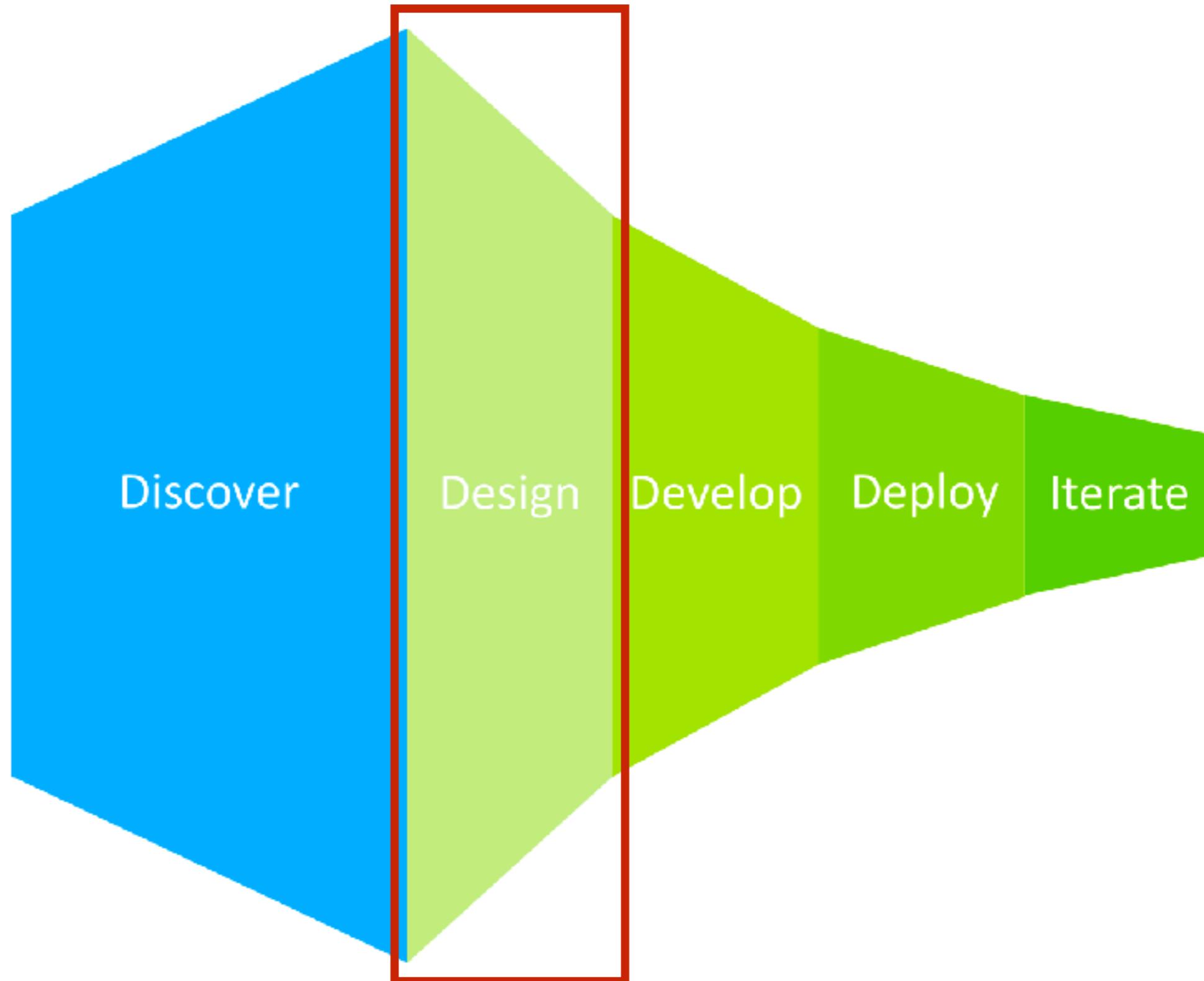


Features & Technology



# Evolution Architecture





# Let's workshop with Design



# E-commerce system



# 1. Search product by name

Adidas NMD

350 ค้นพบสินค้าสำหรับ "Adidas NMD"

เรียงตาม: ความเป็นที่นิยม

จำนวนค่าใช้จ่าย:

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



# 2. Choose a product

Adidas NMD

🔍

🔍 ร้านค้า ทางการ

淘 Taobao คอลเลกชัน

⠇ ไฟฟ์สไตร์ & เติมเงิน

⠇ ใส่โค้ด ลดเพิ่ม

350 ค้นพบสินค้าสำหรับ "Adidas NMD"

เรียงตาม: ความเป็นที่นิยม

จำนวนค่าใช้จ่าย:

	฿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% <span style="color:red">ลด</span> ★★★★★ (70) ลูกค้าปraise
	฿7,990.00 ฿12,000.00 -33% ★★★★★ (1) ลูกค้าปraise



# 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



# 4. Add product to basket

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



# 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 พั้ยิด ทรงเข้ารูป

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**



# 7. Shipping

LAZADA  
CO-TH

1. คำสั่งซื้อ

2. ชำระเงิน

ที่อยู่ที่จะจัดส่ง

Login for speedy checkout

ชื่อและนามสกุล

ที่อยู่

รหัสไปรษณีย์

เมือง

จังหวัด

โทรศัพท์มือถือ

ทางเราจะทำการตรวจสอบเนื้องและจังหวัดของคุณ

เพื่อให้แน่ใจว่าทำการจัดส่งได้

ท่องเที่ยวในประเทศ/ในกำกันภาษี - กรุณาเดือนของการขอข้อมูลเพื่อทำการขอในกำกันภาษี

ข้อมูลการส่งเงินค้า

ชั่งแบบธรรมชาติ: พีวี

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

ดำเนินการต่อ

สูปการสั่งซื้อ (1 items)

สินค้า	จำนวน	ราคาร
POCA SHOE NMD Sneakers Fashion รองเท้า ล่าสุด แนว บู๊ก-สีขาว แมทช์ ราคาถูกสุดๆ Sport Unisex รุ่น PSN-Black/White ขนาด: EU:40	1	399
ยอดรวมค่า		399 บาท
ยอดสุทธิ		399 บาท

มาตรฐานมากยิ่งขึ้นค่าเพิ่ม (ถ้ามี)

 คุ้มครองคุณภาพ 100%





# 8. Payment

LAZADA  
.CO.TH

✓ 1. ค่าซื้อขั้นต่ำ

2. ชำระเงิน

**เลือกคัวเลือกสำหรับการชำระเงิน**

บัตรเดบิตหรือ เทิร์บเงินปลายทาง	ชำระเงินผ่าน เดบิตหรือ	PayPal/Amex	มอนชาระ	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	ราคารวม
POCA SHOE NMD Sneakers Fashion รองเท้า ลั่วสังฆ่า ใบไม้ ผู้หญิง-ผู้ชาย แฟชั่น ราคาถูกสุดๆ Scott Unisex รุ่น PSN-Black/White ขนาด: EU:40	1	399 บาท

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

กรอกคุณปวงส่วนลดที่มี  **ขึ้นชั้น**

มูลค่าสินค้า **ค่าซื้อขั้นต่ำ** 399 บาท  
ยอดสุทธิ **รายการรวมภาษีมูลค่าเพิ่ม (มีภาษี)** 399 บาท

ทุมควรอุปกรณ์ 100%



# 9. Confirm to order

LAZADA  
.CO.TH

✓ 1. ค่าซื้อขั้นต่ำ

2. ชำระเงิน

**เลือกคัวเลือกสำหรับการชำระเงิน**

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

หมายเหตุบัตร

ชื่อบนบัตร  Somkiat Puisungnoen

วันที่บัตรหมดอายุ  mm  yy      CCV / CVV  ?

ข้อมูลใบสำคัญไม่สามารถเปลี่ยนแปลงได้หลังการสั่งซื้อสินค้า

**ล็อก 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 ลำลอง ถ้าใบ สีฟ้า-ฟ้าเข้ม ราคากลางๆ Scott Unisex รุ่น PSN-Black/White ขนาด: EU:40	1	399 บาท

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

กรอกคุณปวงส่วนลดที่นี่  **ขึ้นชัน**

มูลค่าสินค้า  
**ค่าซื้อขั้นต่ำ** 399 บาท  
บาท

**ยอดสุทธิ** ราคารวมภาษีมูลค่าเพิ่ม (มีภาษี) 399 บาท

**ทุมดาวลูกค้า 100%**

**Lazada Security Approved JUN-2016**



Microservices

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

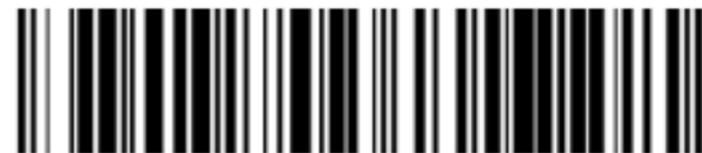
# 10. Summary



## ใบแจ้งการชำระเงิน(PaySlip)

Counter Service Co., Ltd.

เลขที่ใบแจ้งสั่นด้า/Invoice No:	3779254692
ผู้ชำระเงิน/Payer:	Somkiat Puisungnoen
วันที่รายการ / Transaction Date:	25/02/2018 23:33
กำหนดชำระเงิน / Expired Date:	<b>27/02/2018 23:33</b>
เพื่อเข้าบัญชี / Payee:	www.lazada.co.th Tel: <b>020180000</b>
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)



[Back to merchant](#)

[Print](#)

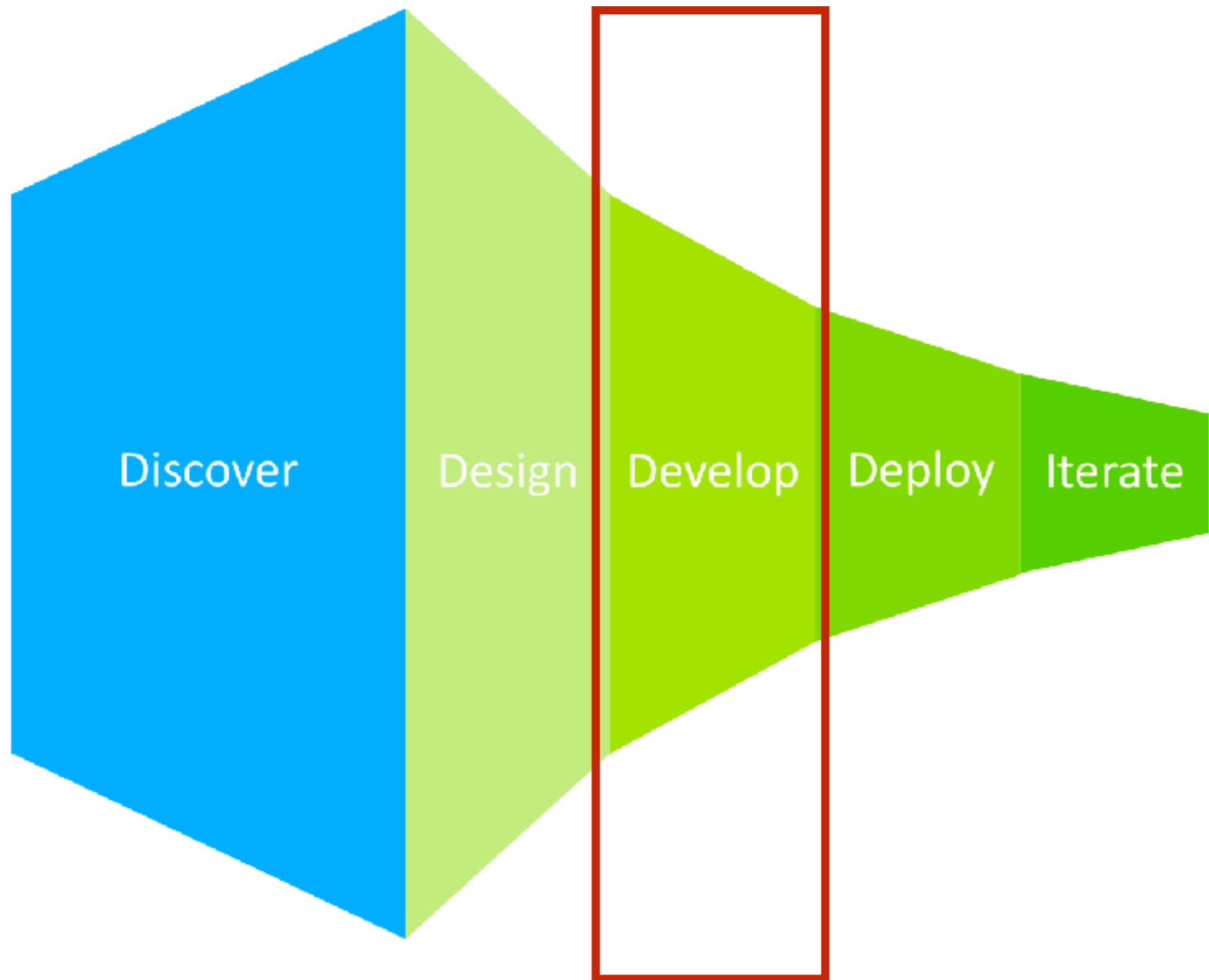


Microservices

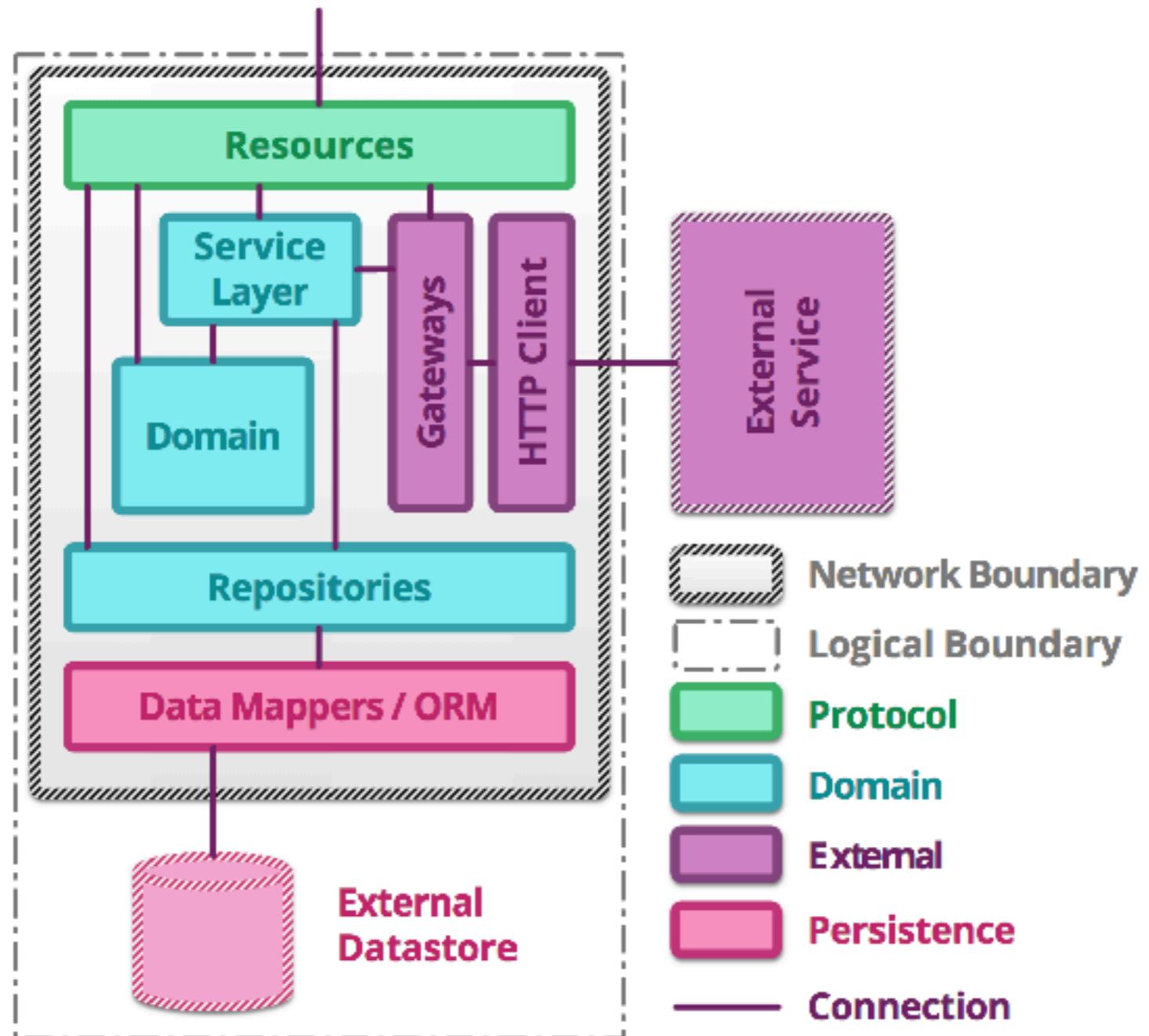
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

# Try to design system





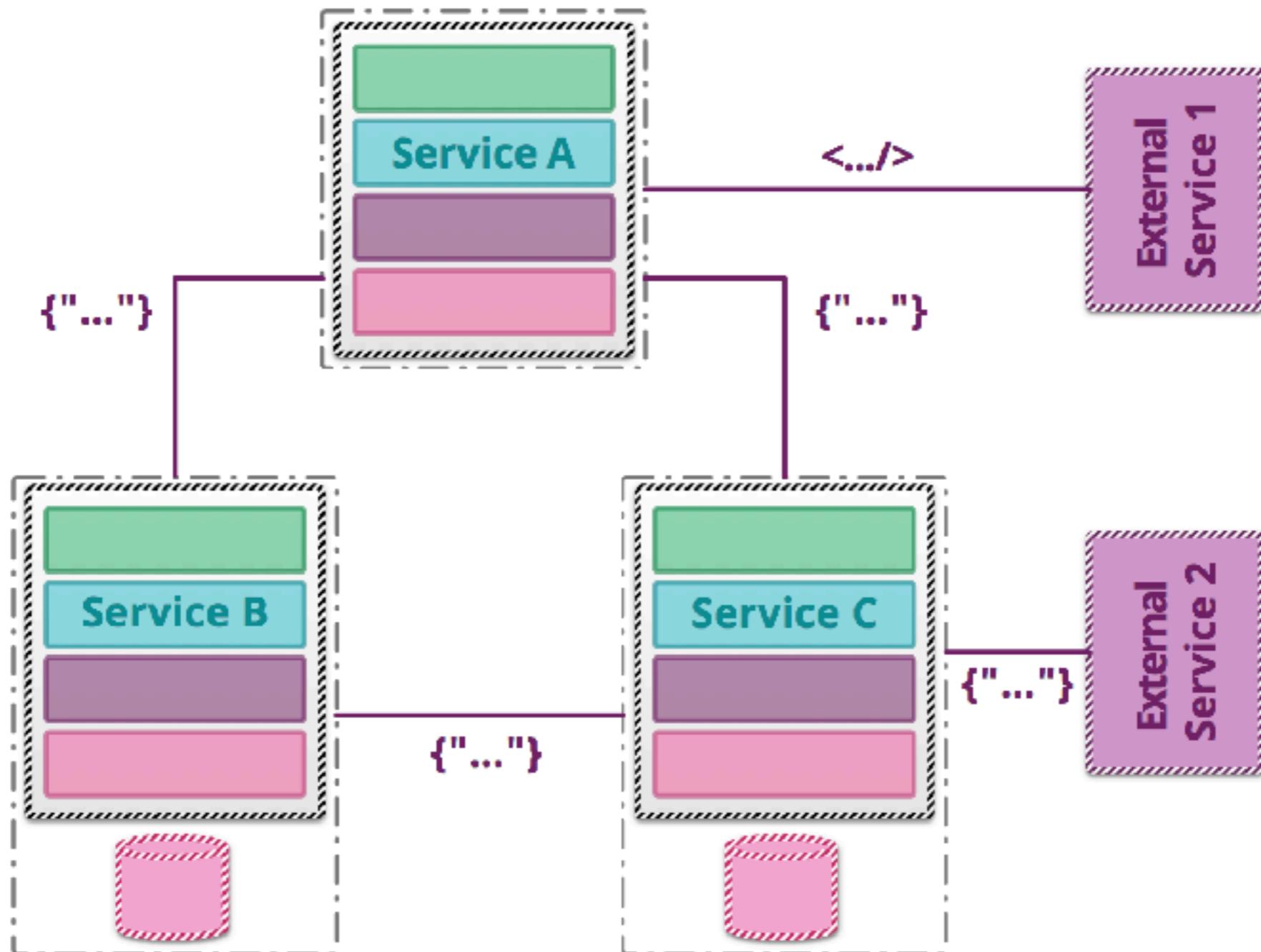
# Service structure



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



# Multiple services

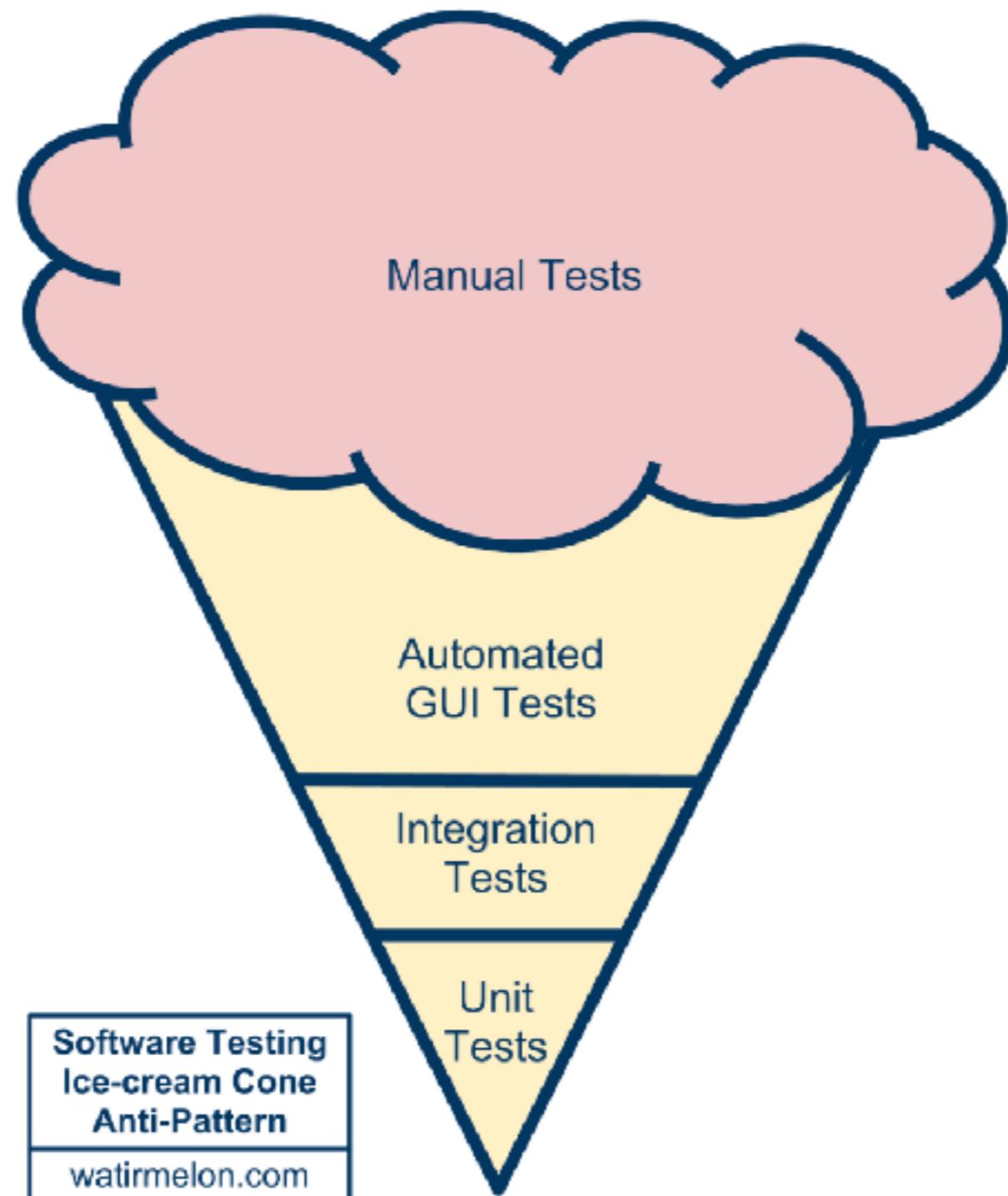


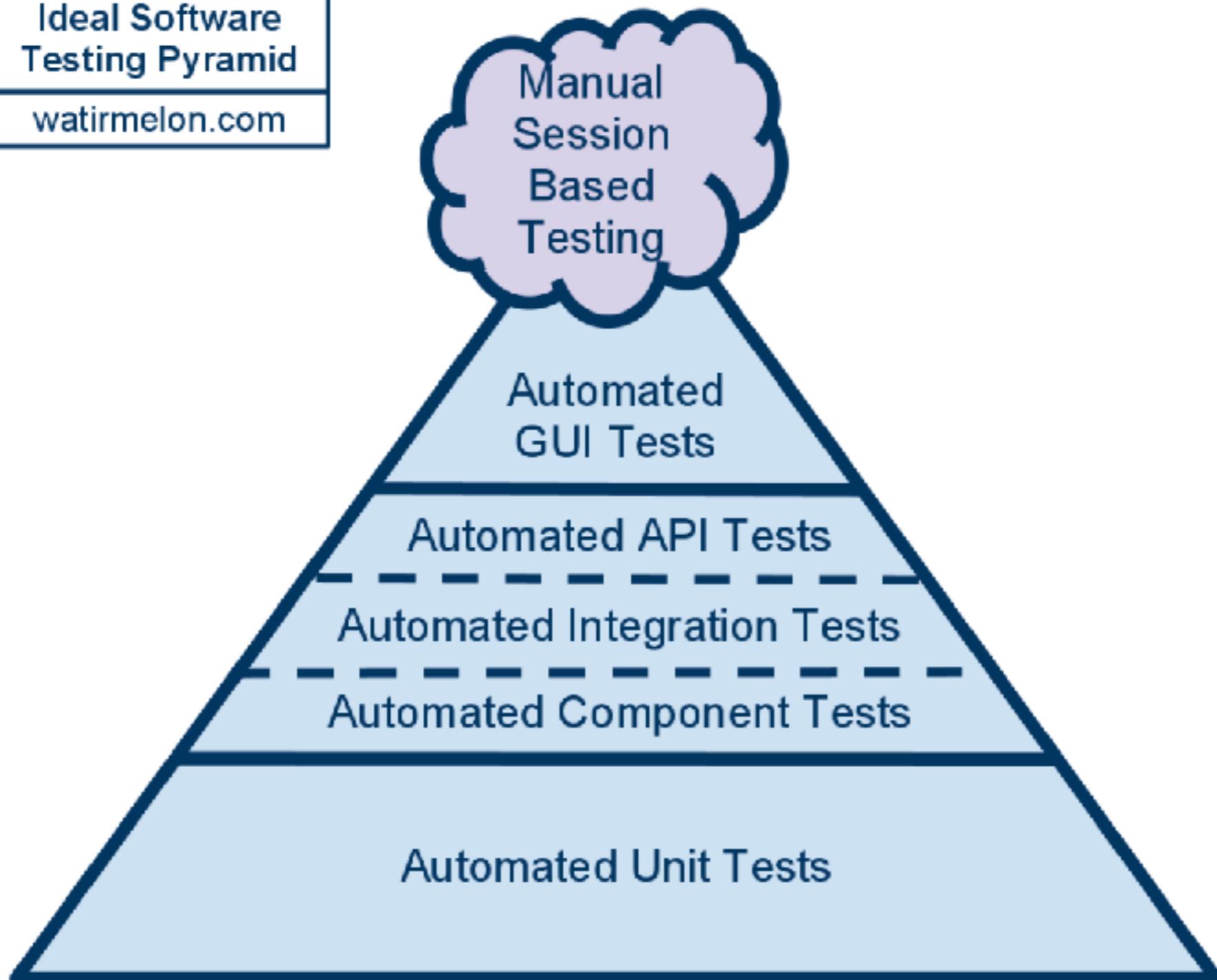
<https://martinfowler.com/articles/microservice-testing>

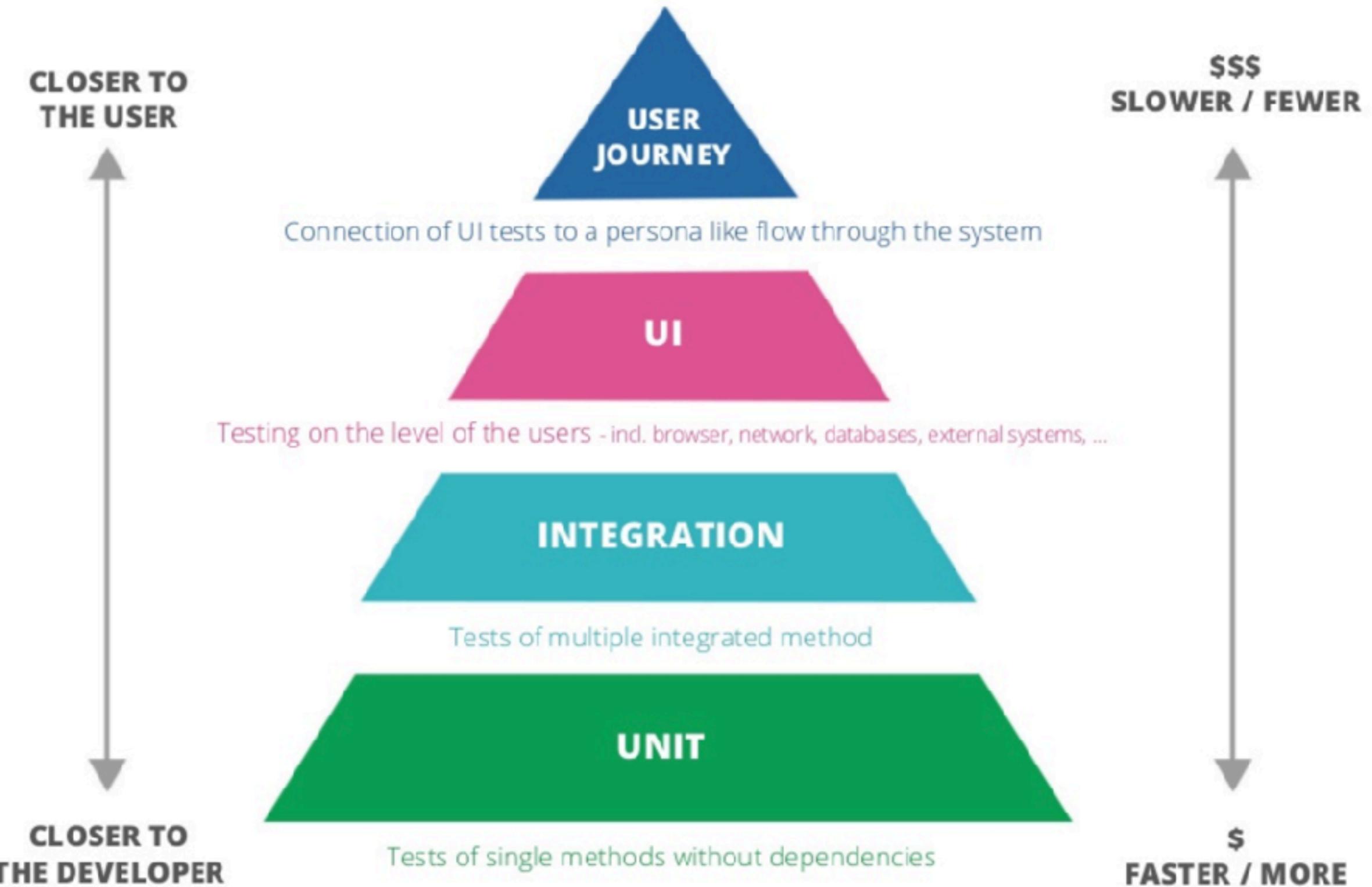


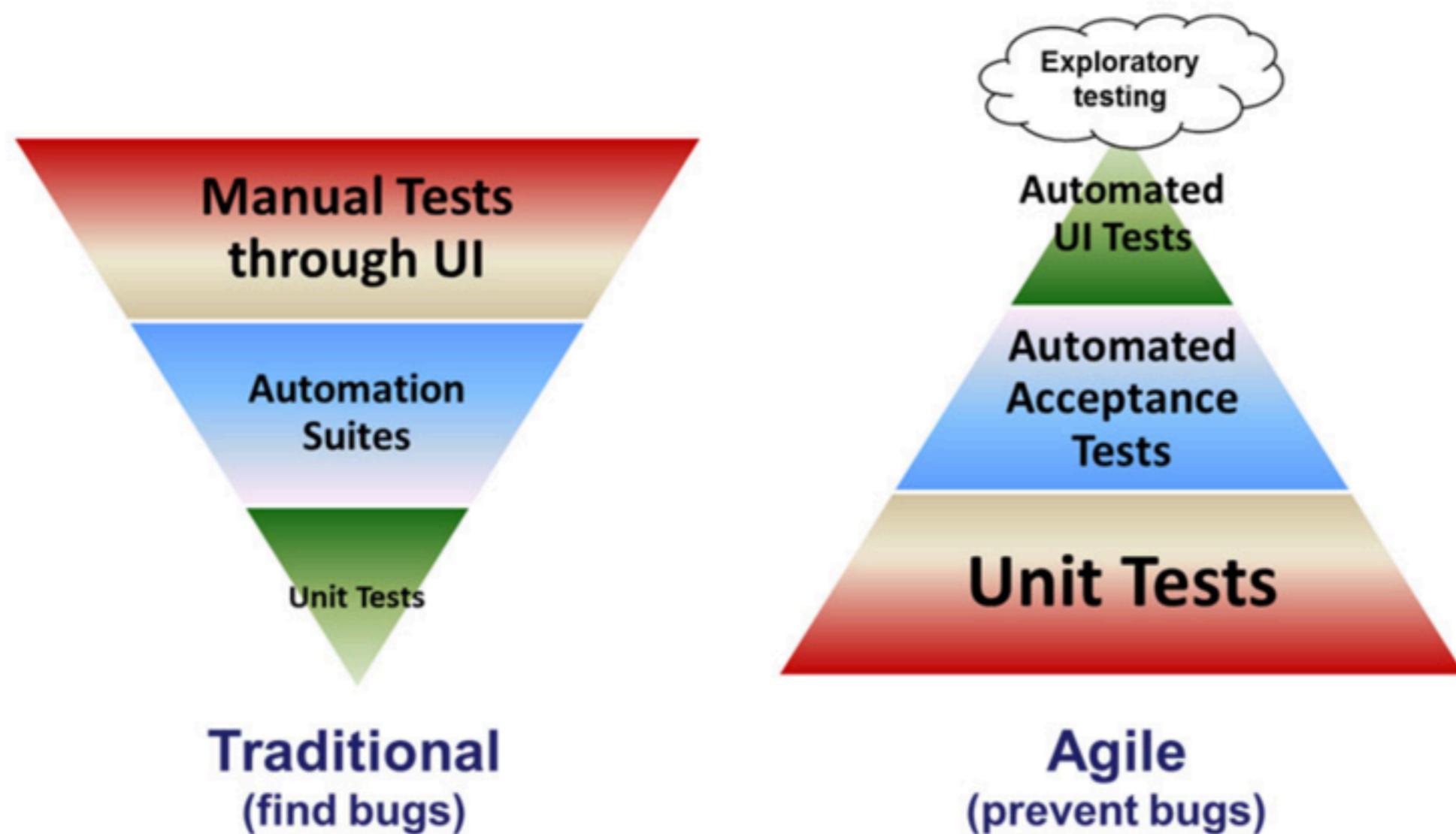
# Microservice Testing

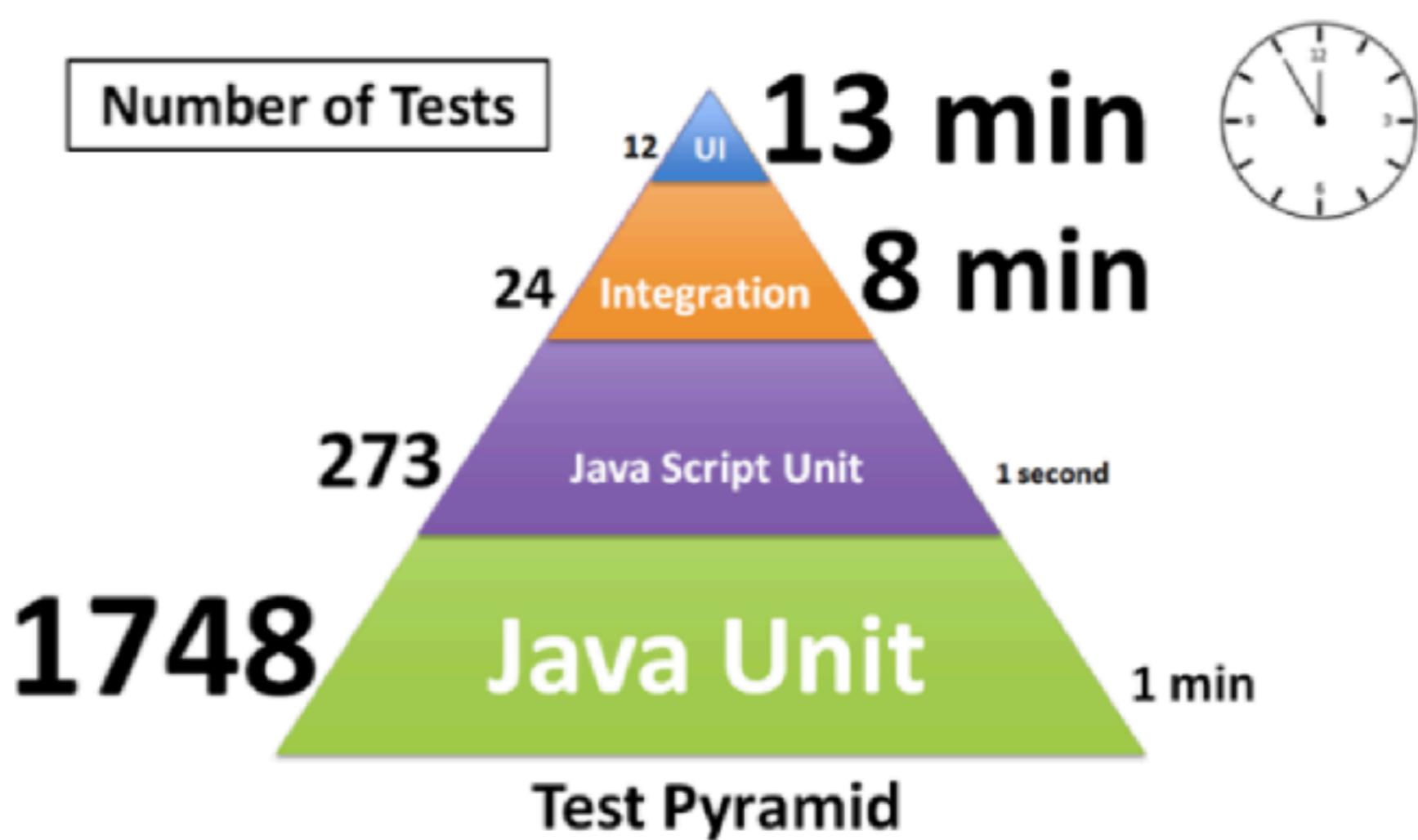


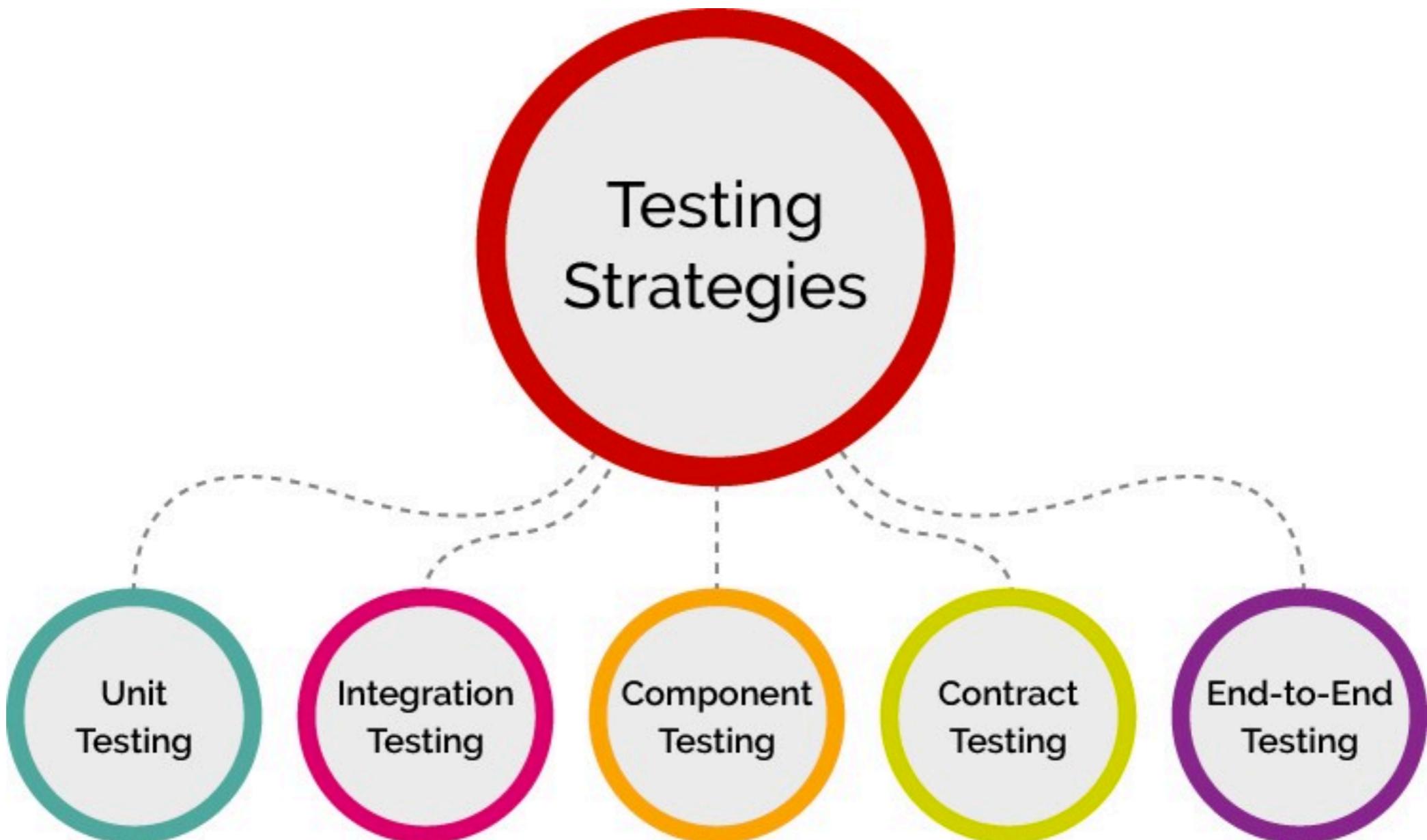




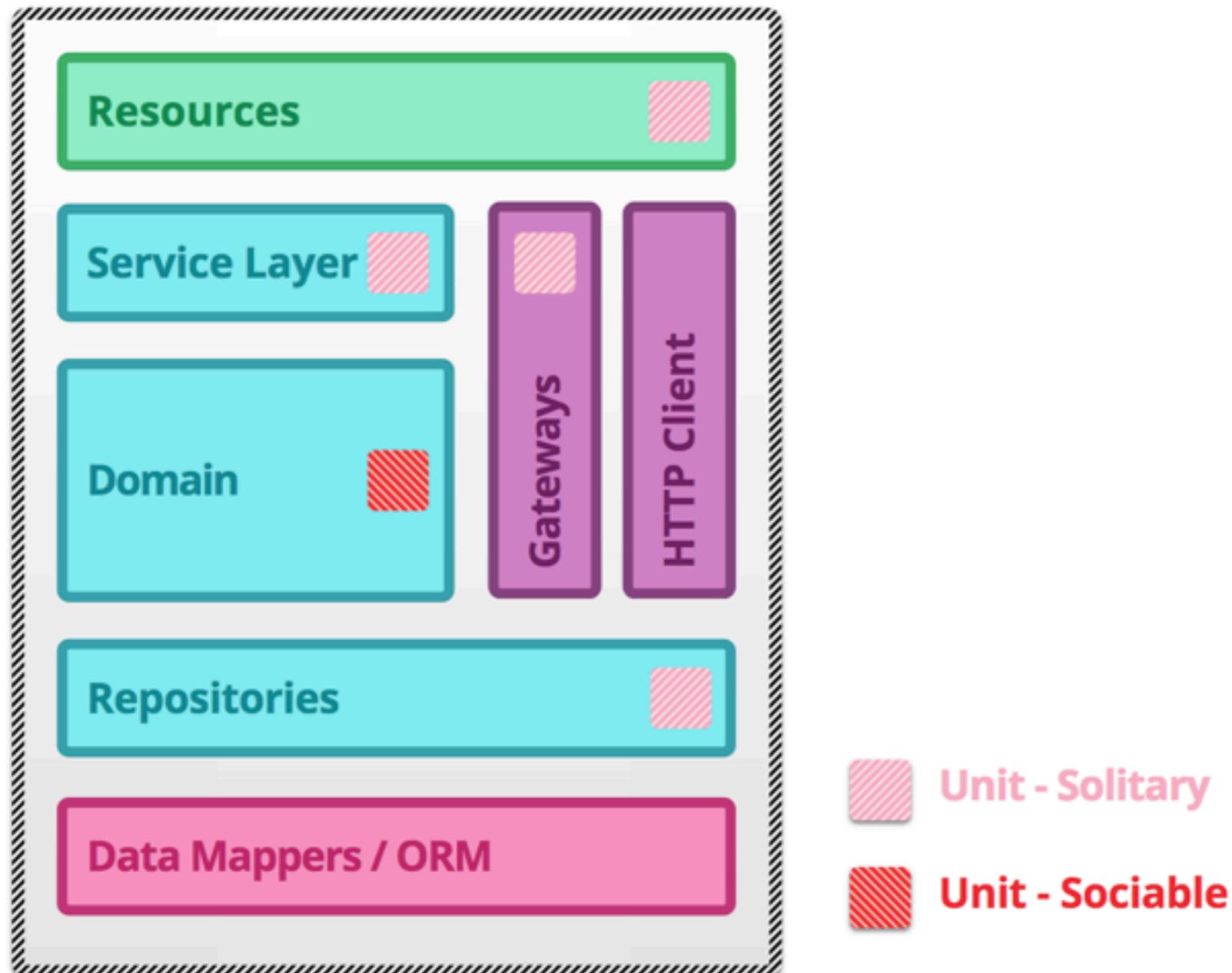




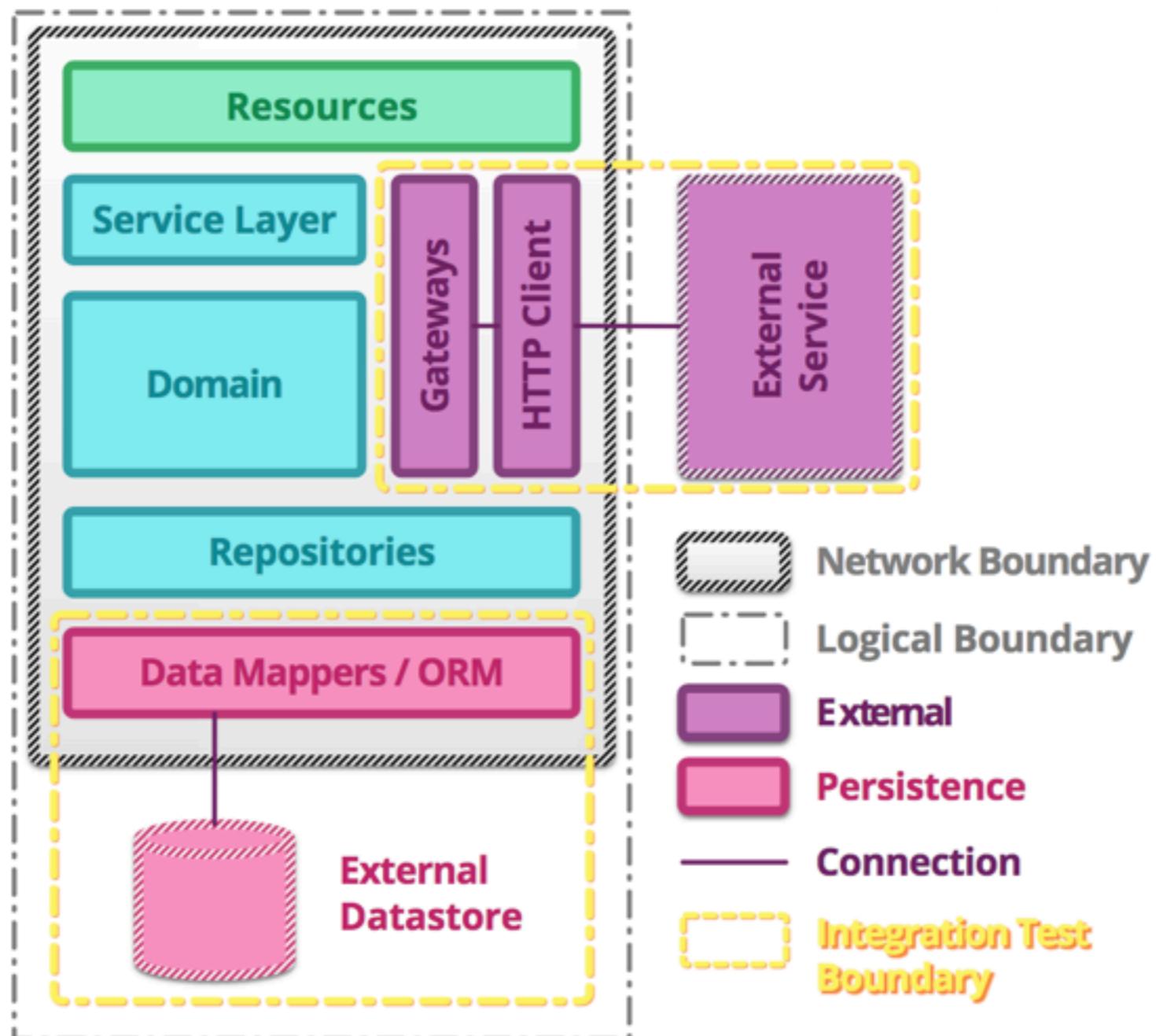




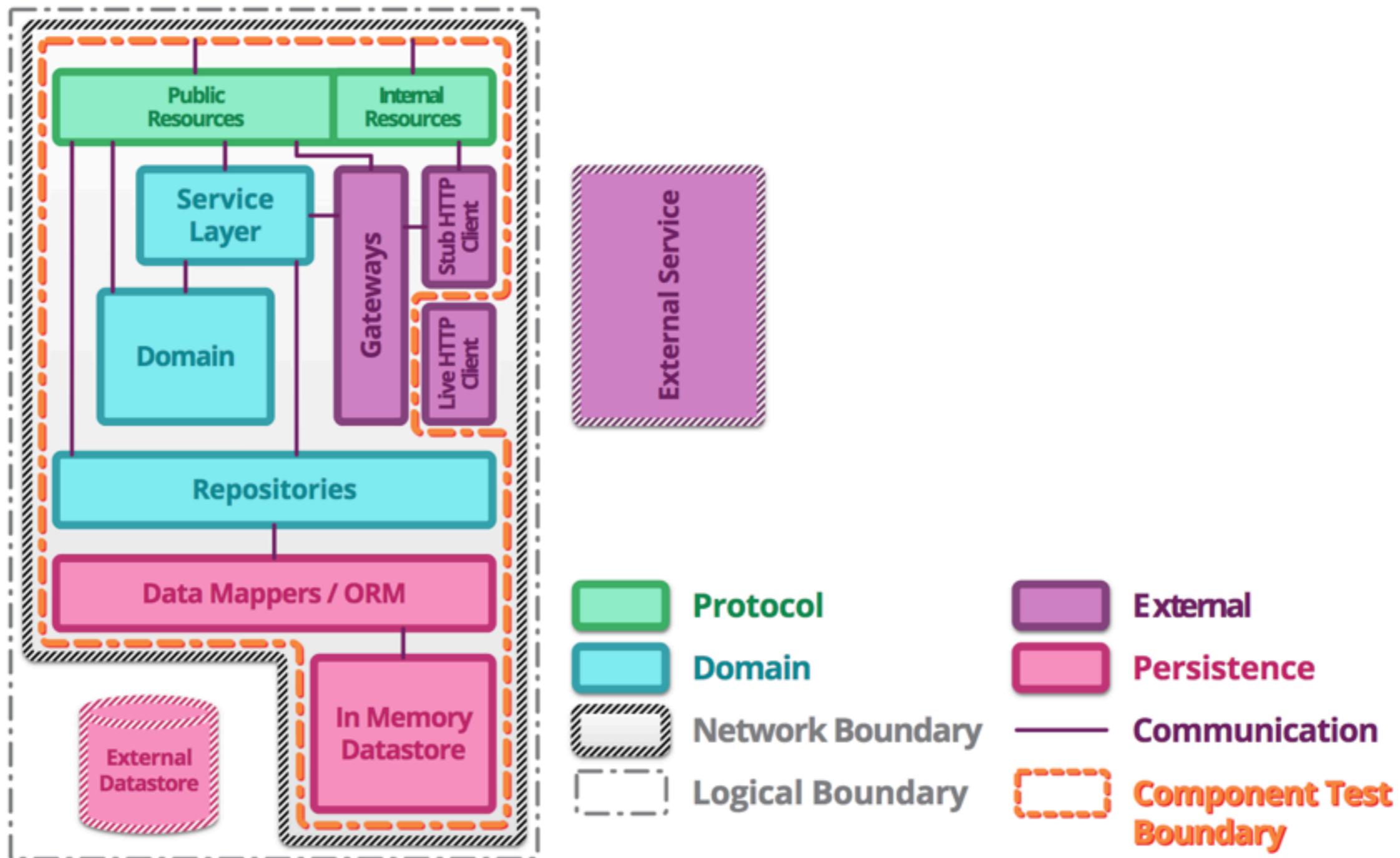
# Unit testing



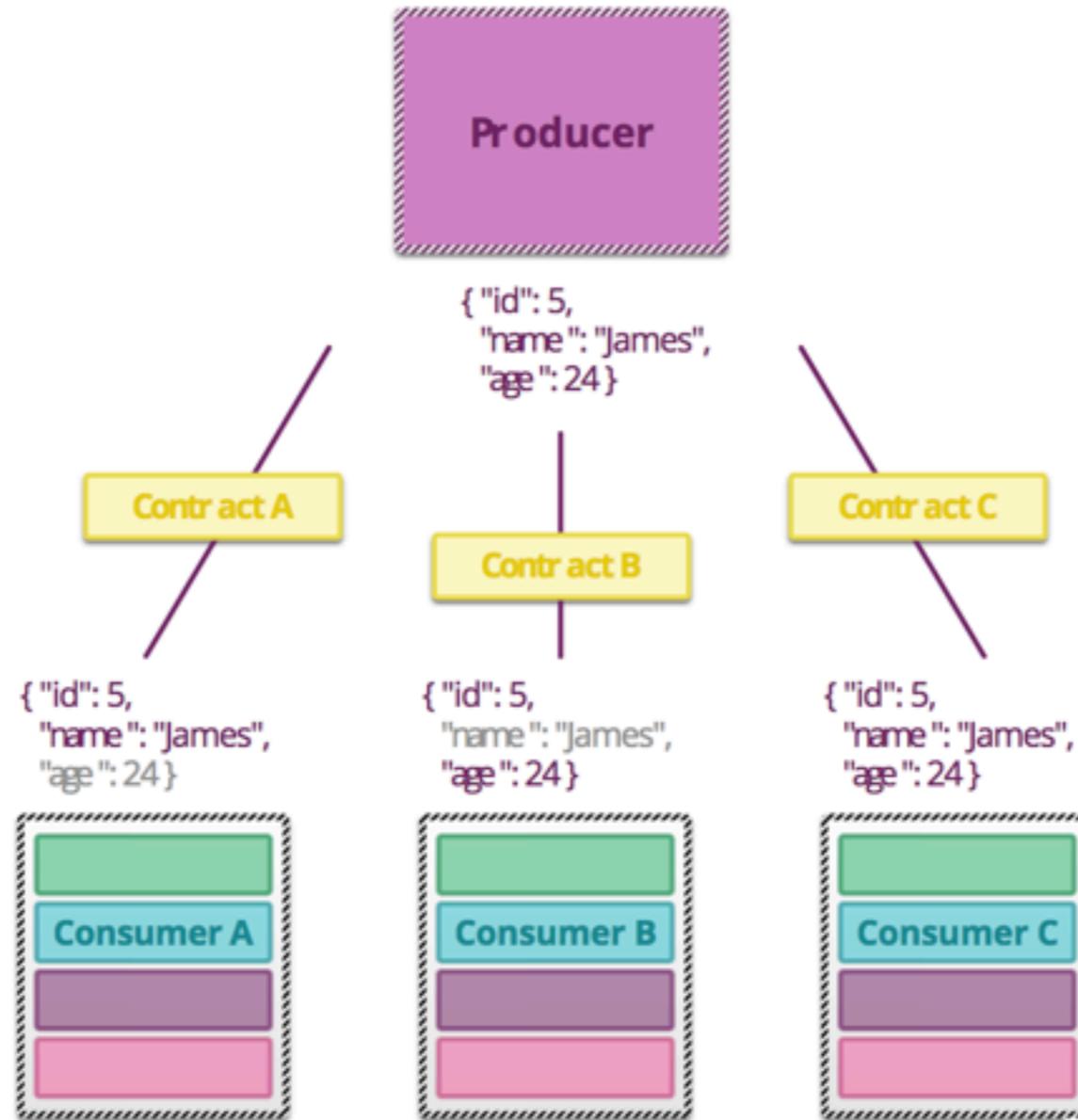
# Integration testing



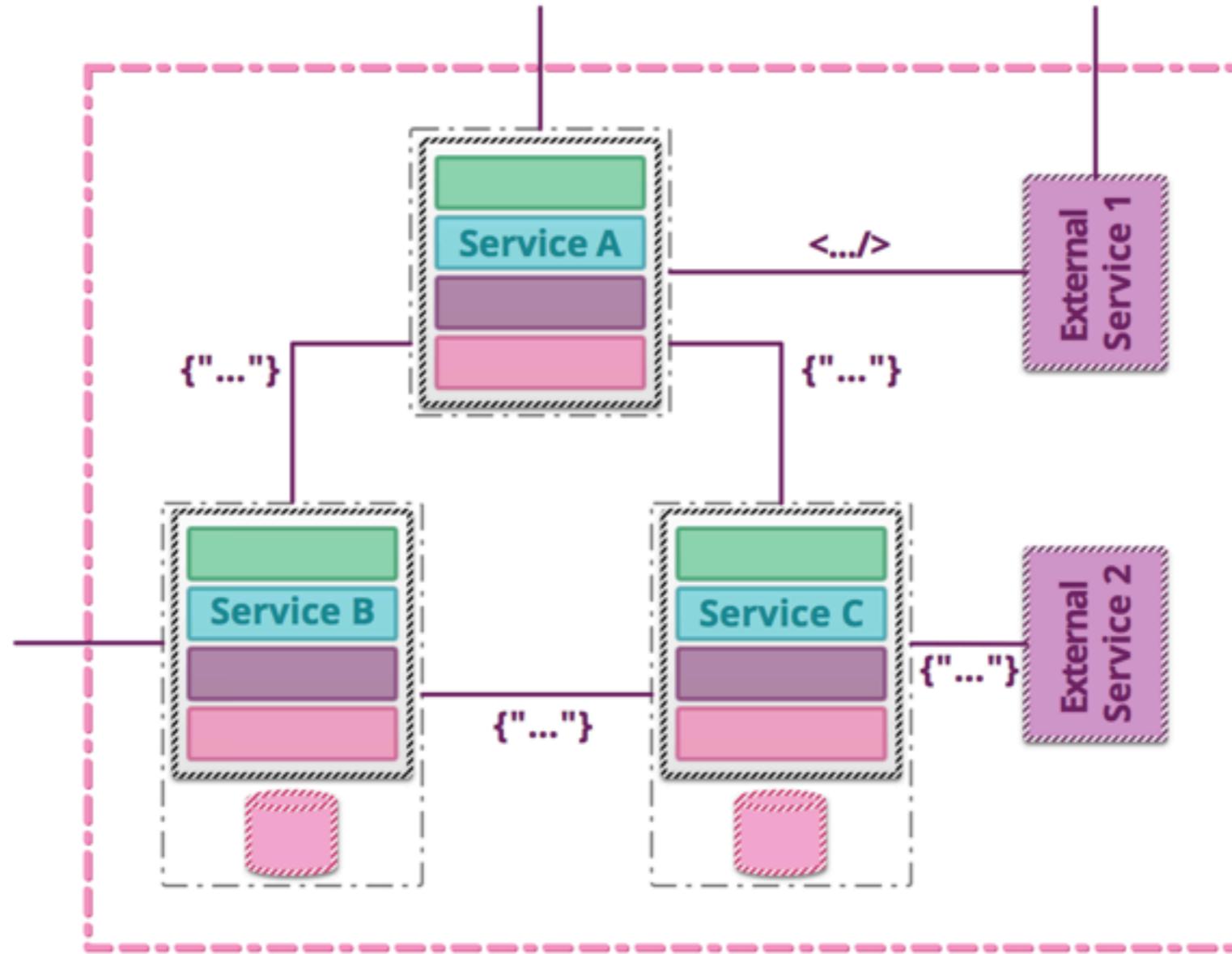
# Component testing



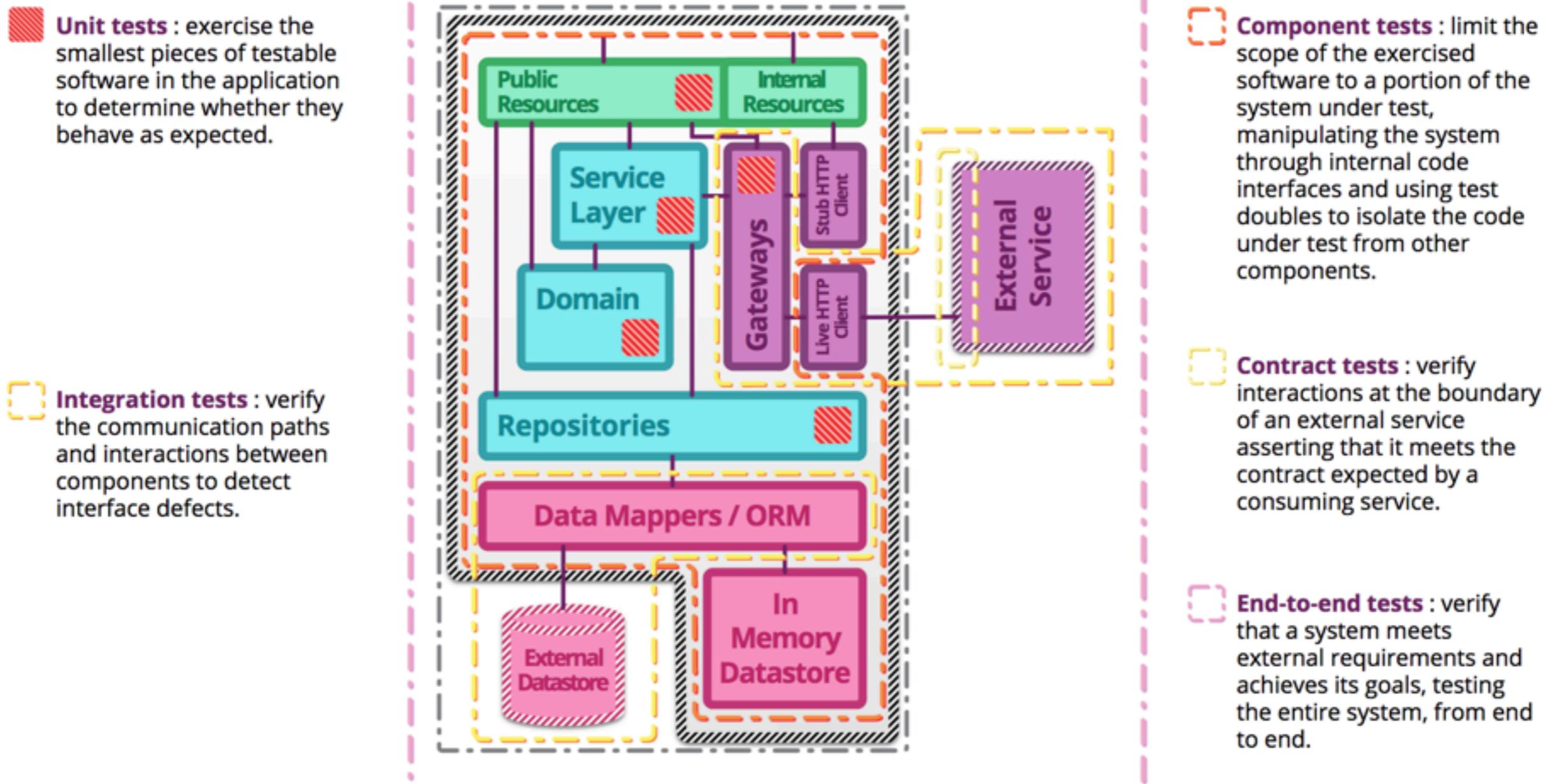
# Contract testing



# End-to-End testing



# Summary



# What is your testing strategy ?

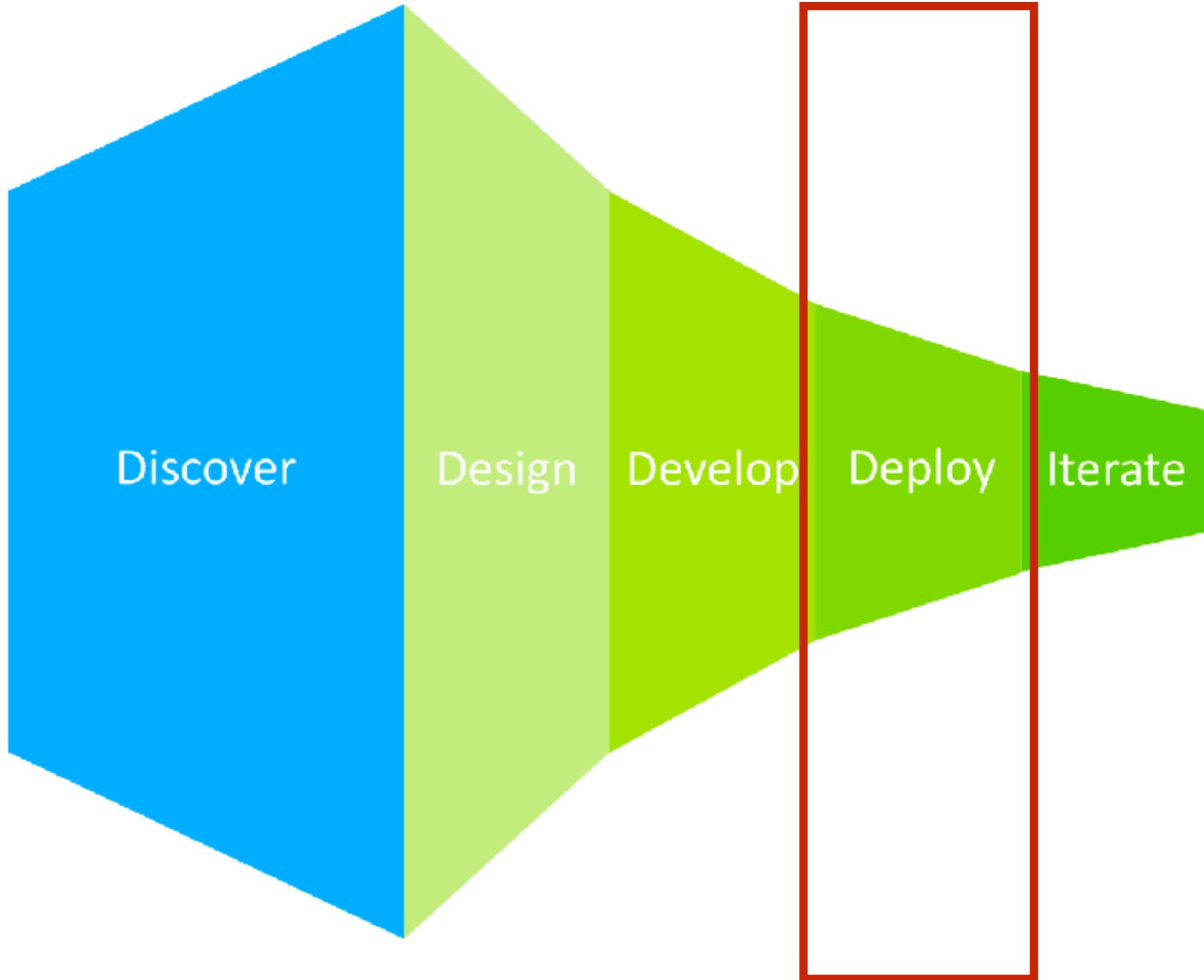


# Workshop



# Workshop





# Deployment





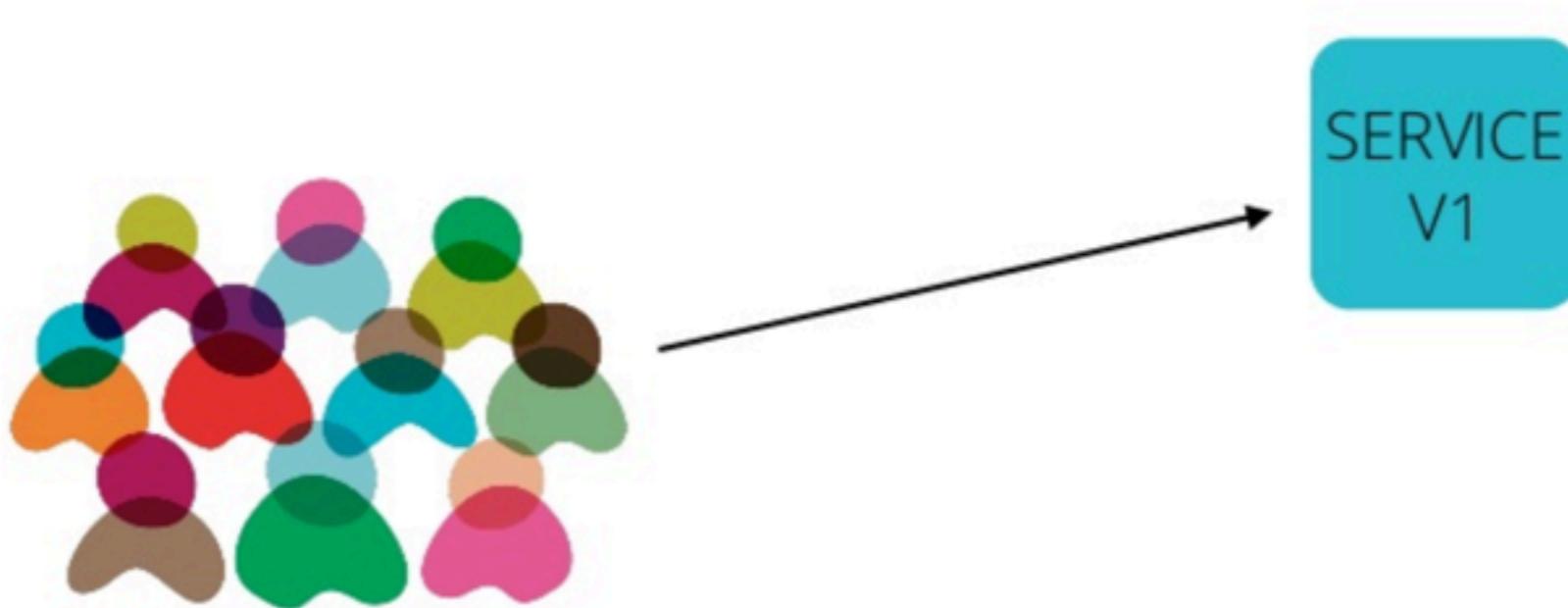
# Deploy vs Release



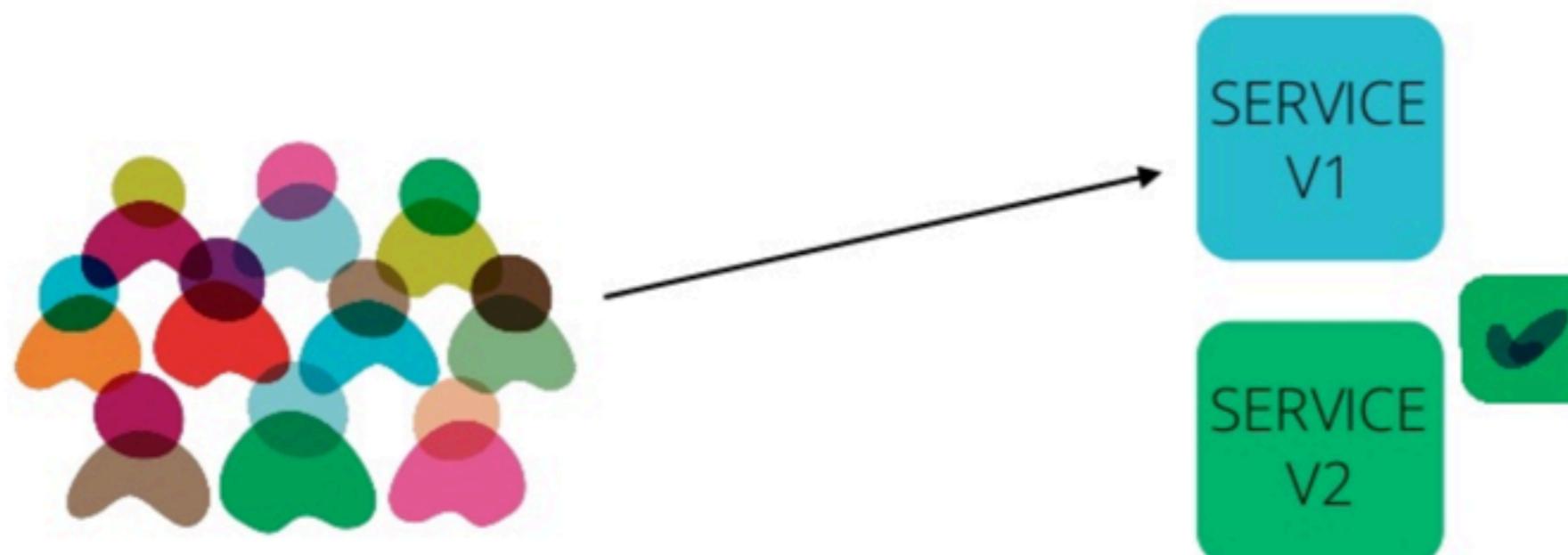
# Blue Green Deployment



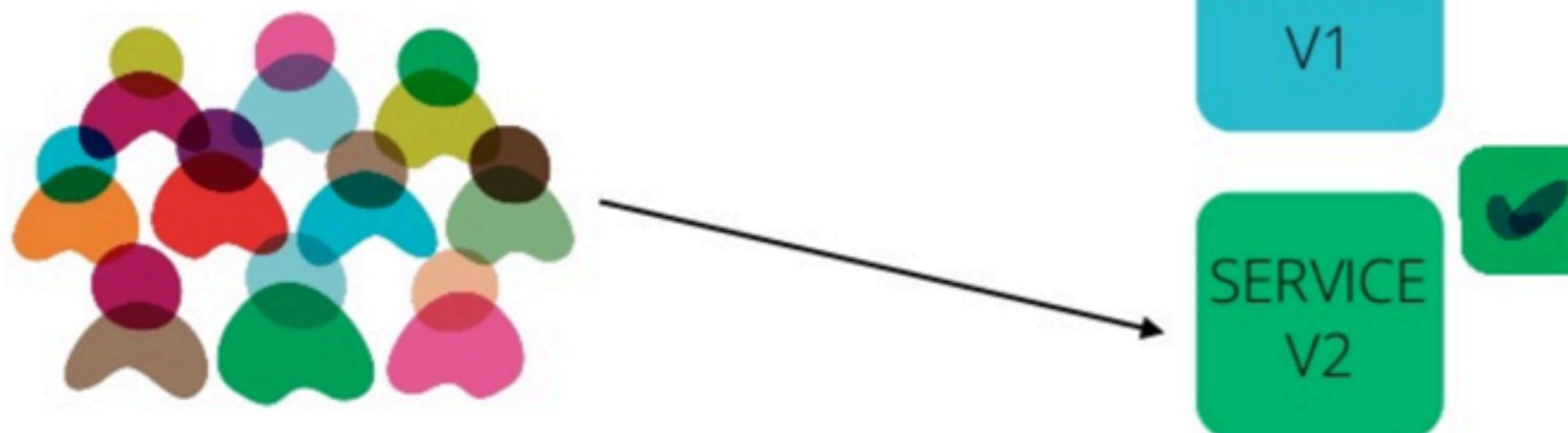
# Blue Green Deployment



# Blue Green Deployment



# Blue Green Deployment



# Canary Release



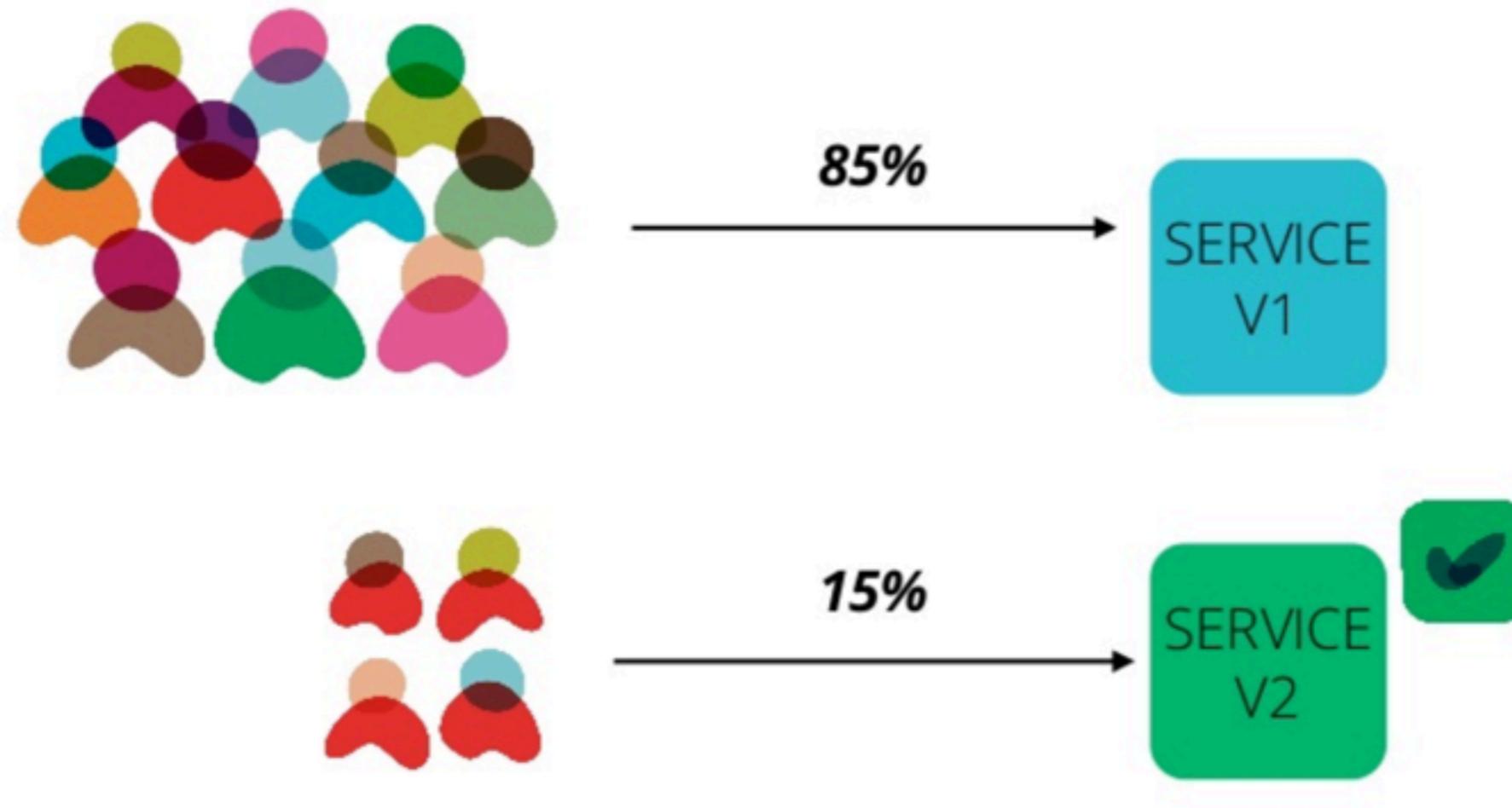
# Canary Release



# Canary Release



# Canary Release



# Mean Time to Recover (MTTR)



# Mean Time to Recover (MTTR)

**Tests** are very important to reduce amount of defects in your systems. However, it's important to acknowledge that bugs will always happen in production.



# Mean Time to Recover (MTTR)

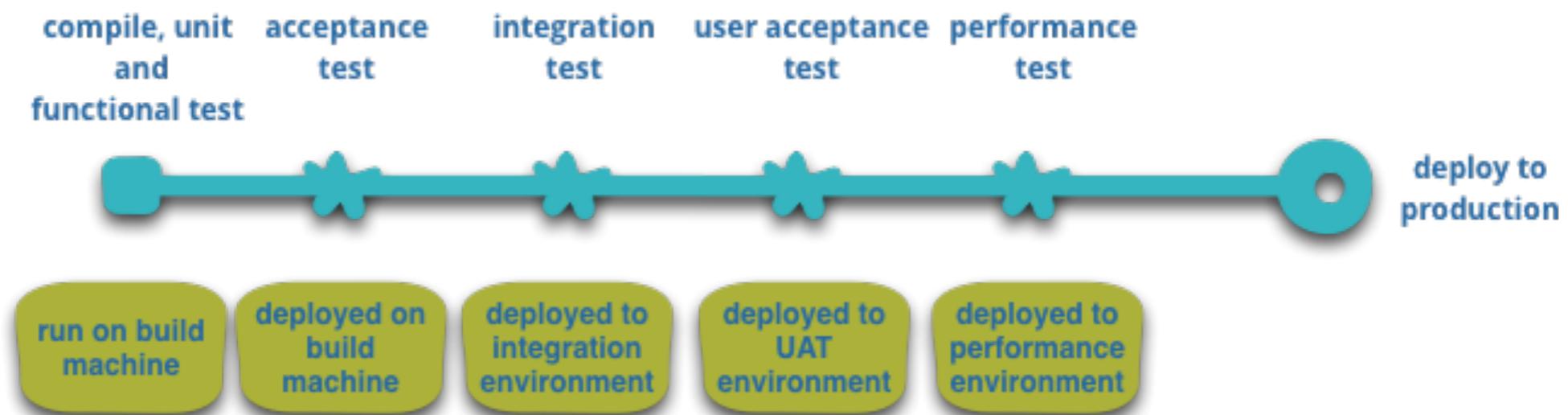
How **fast** to recover from them will help determining our success !



# Current situation !!

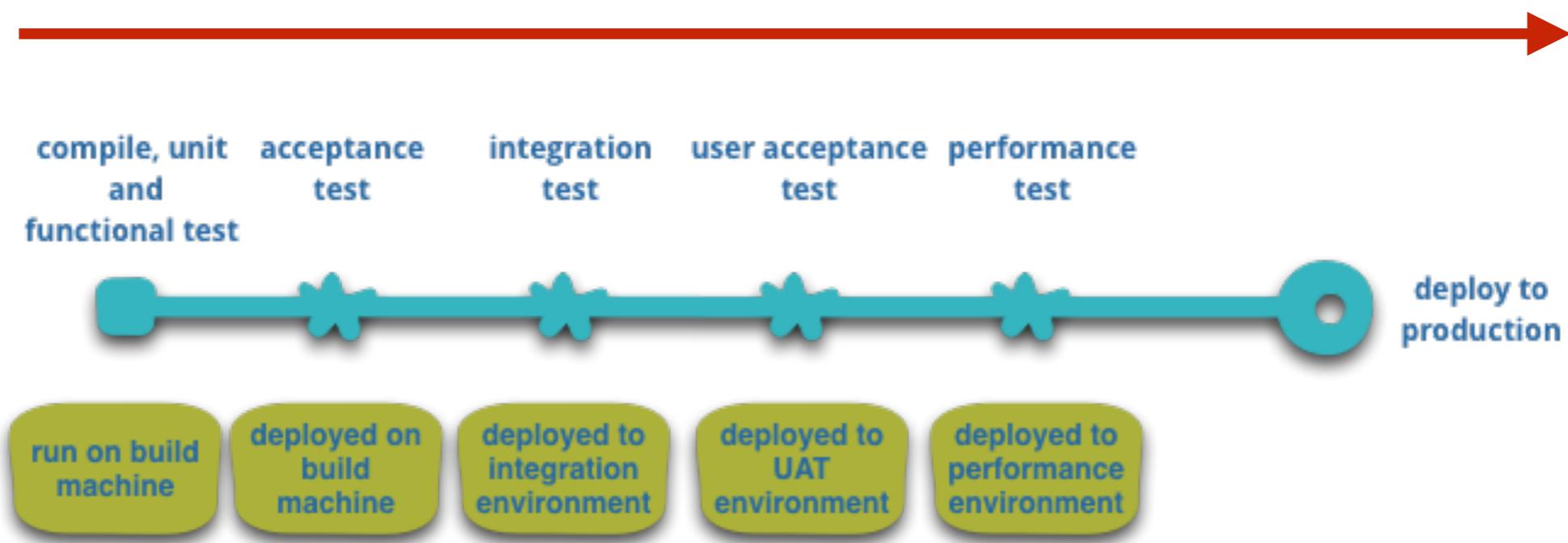


# Infrastructure Automation

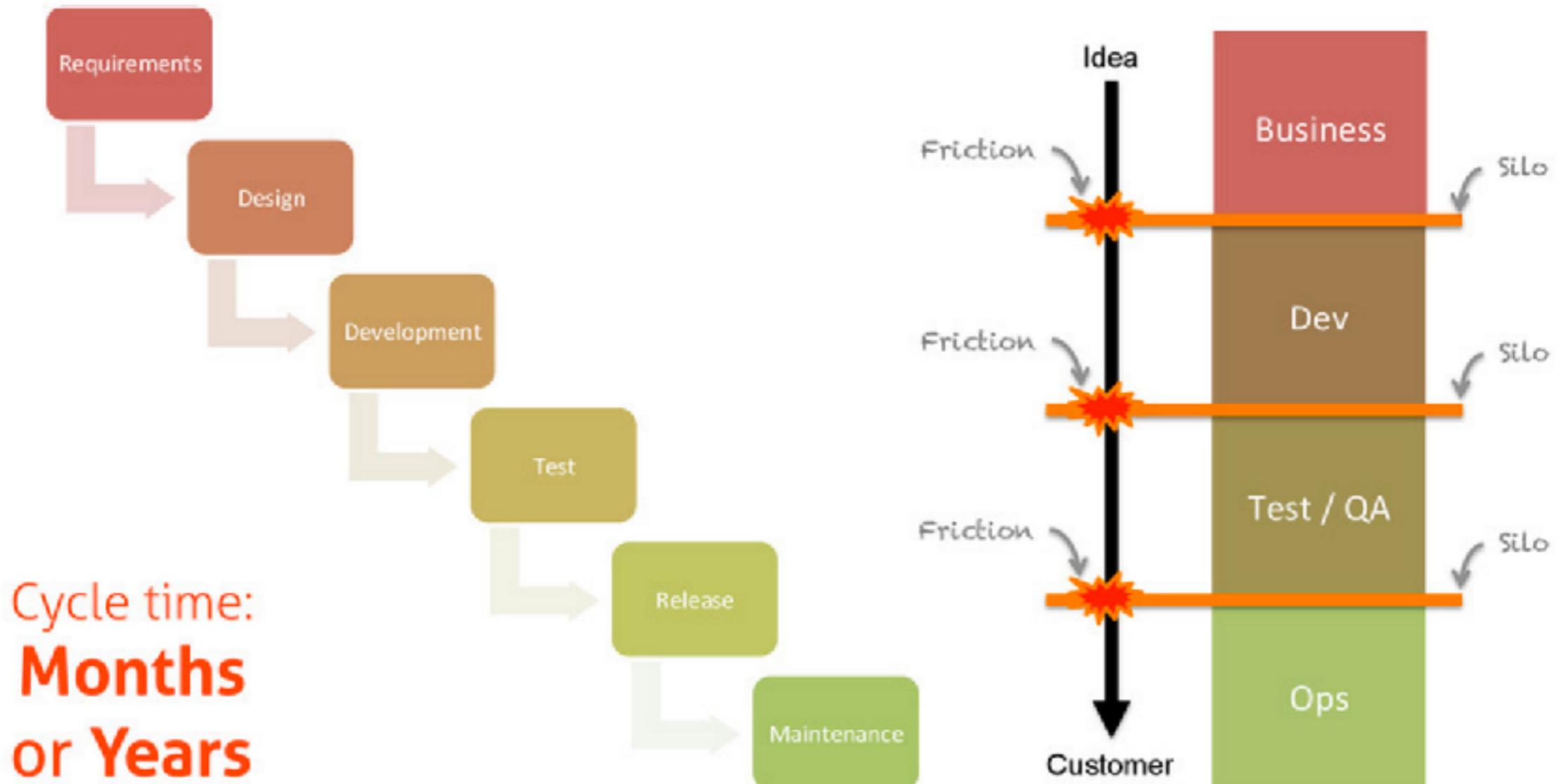


# Infrastructure Automation

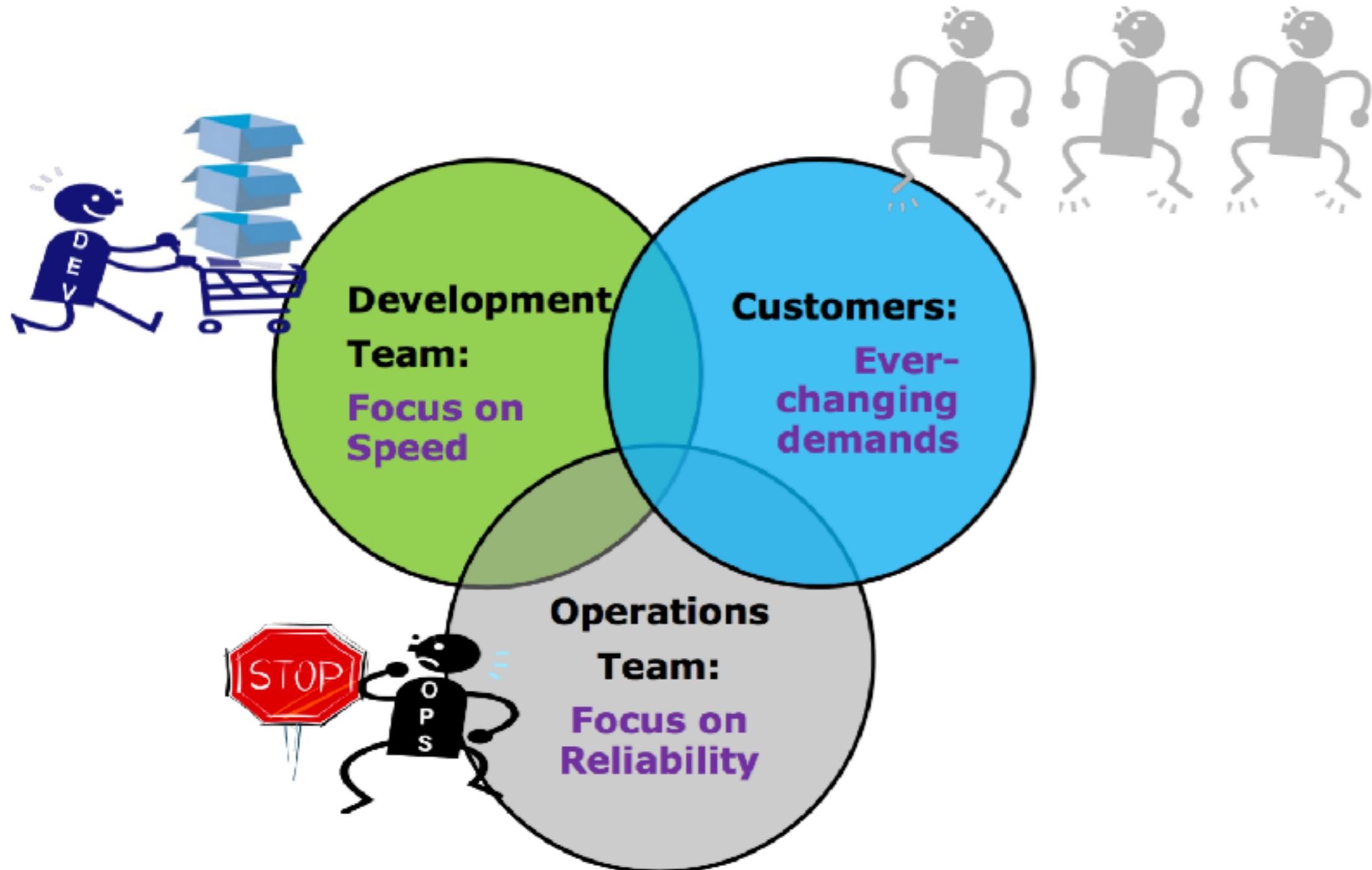
## Lead time ?



# Traditional development



# Conflict of Interest



# Conflict of Interest

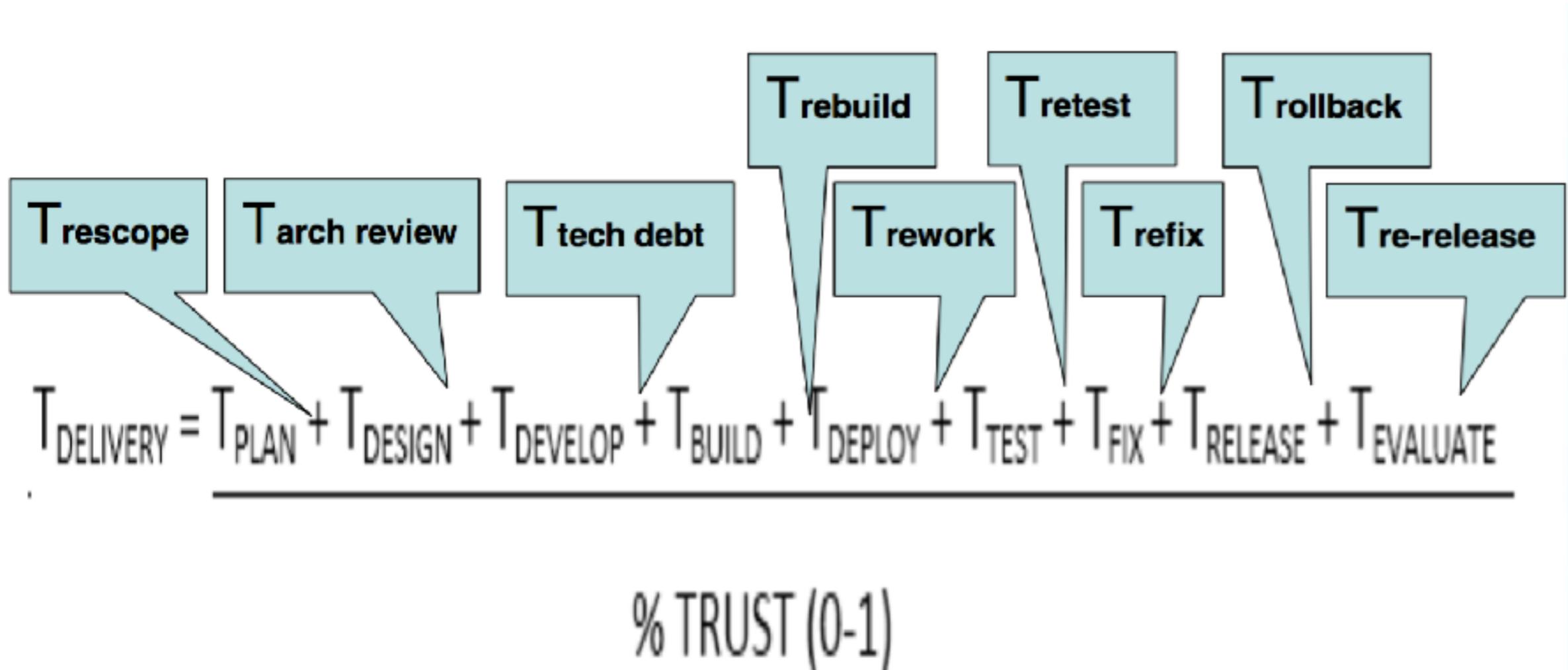


# Conflict of Interest

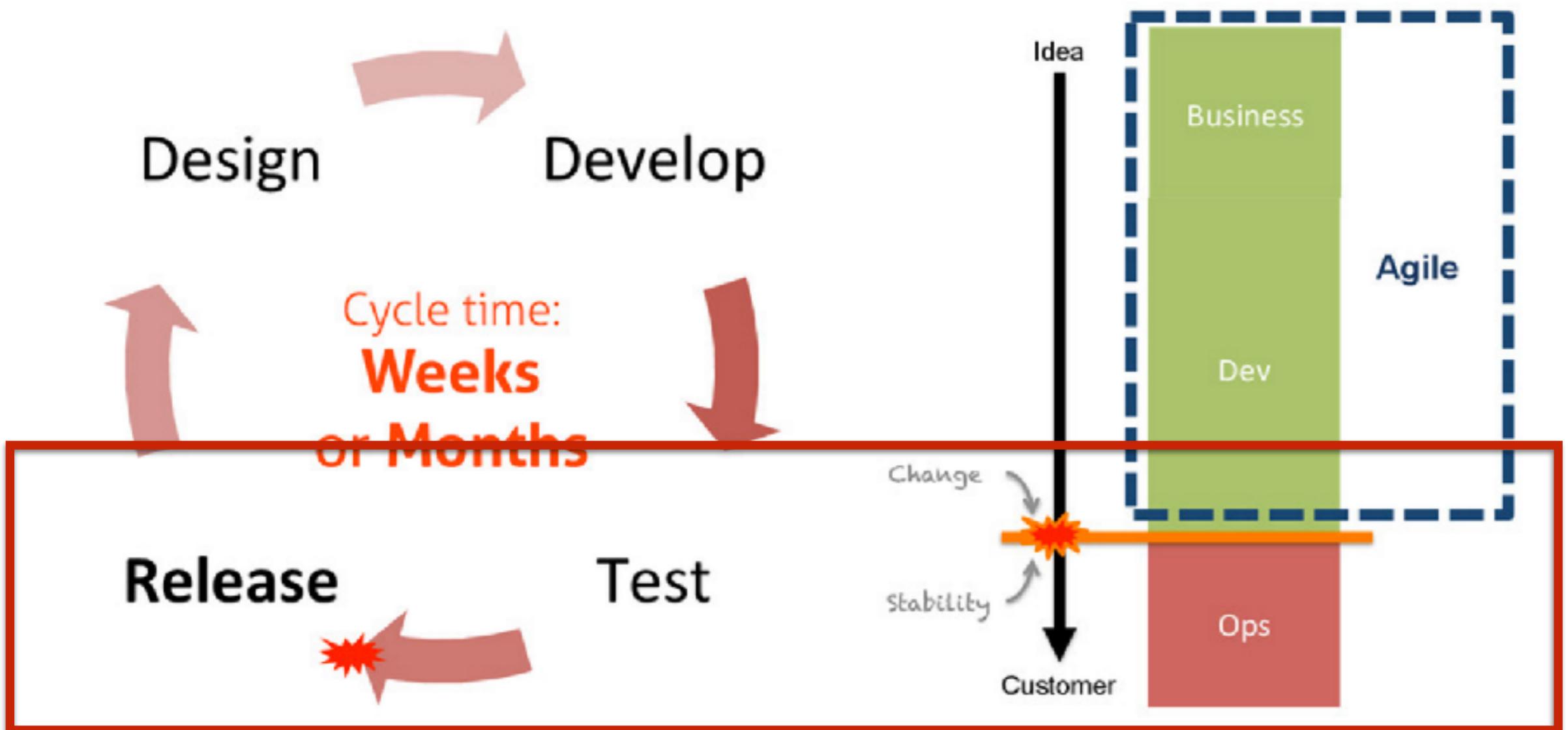




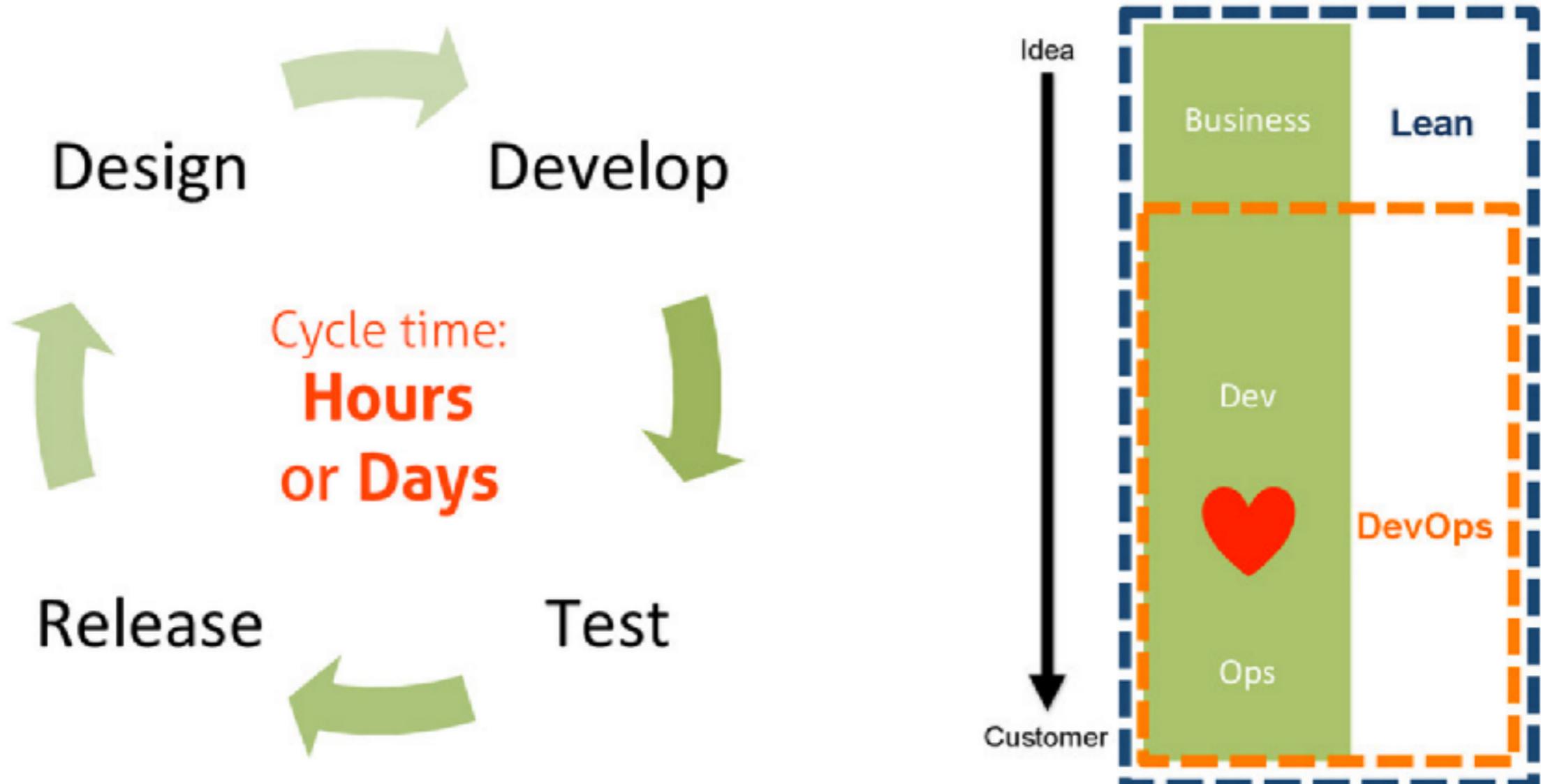
# Low trust create extra steps

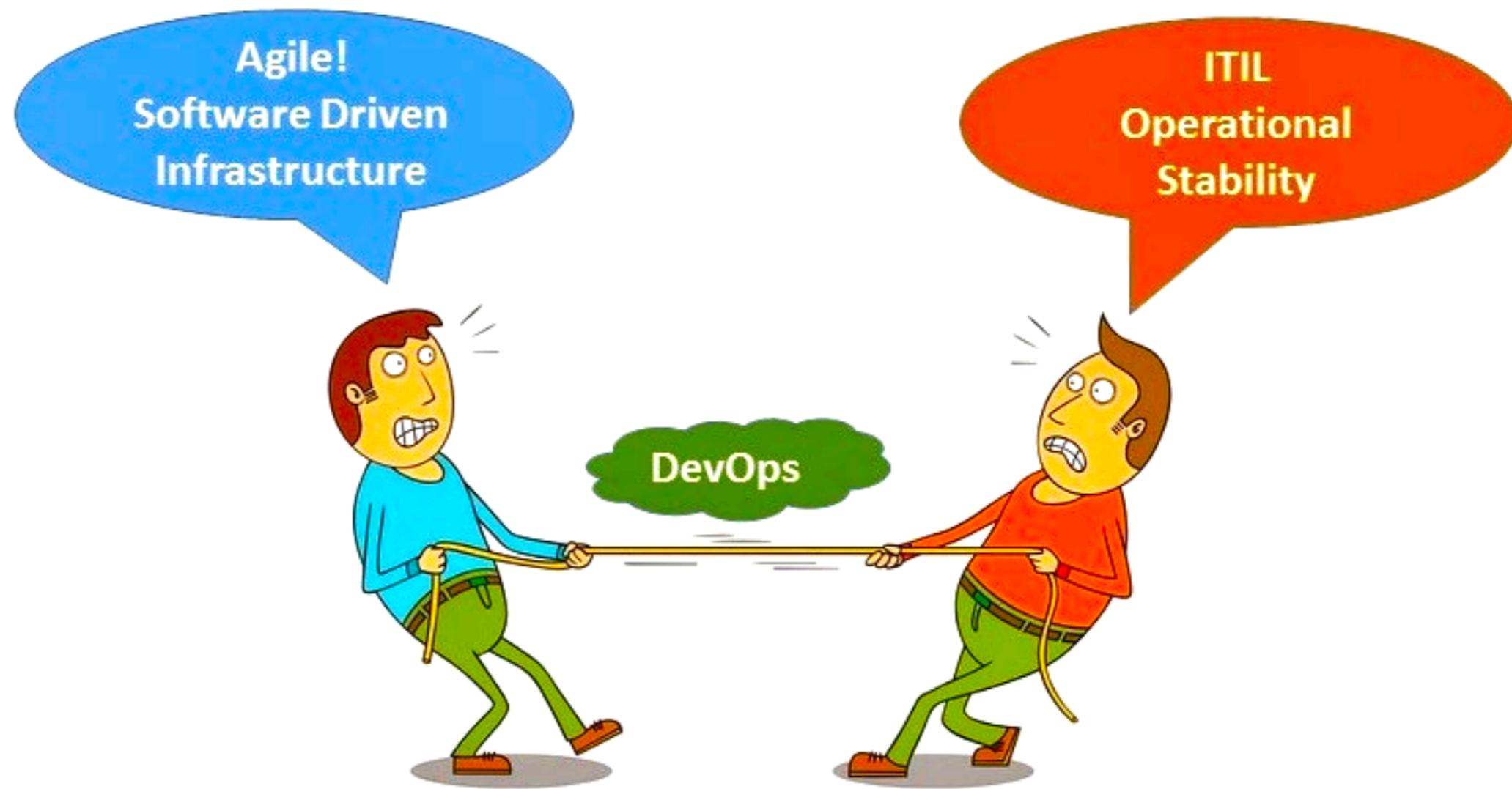


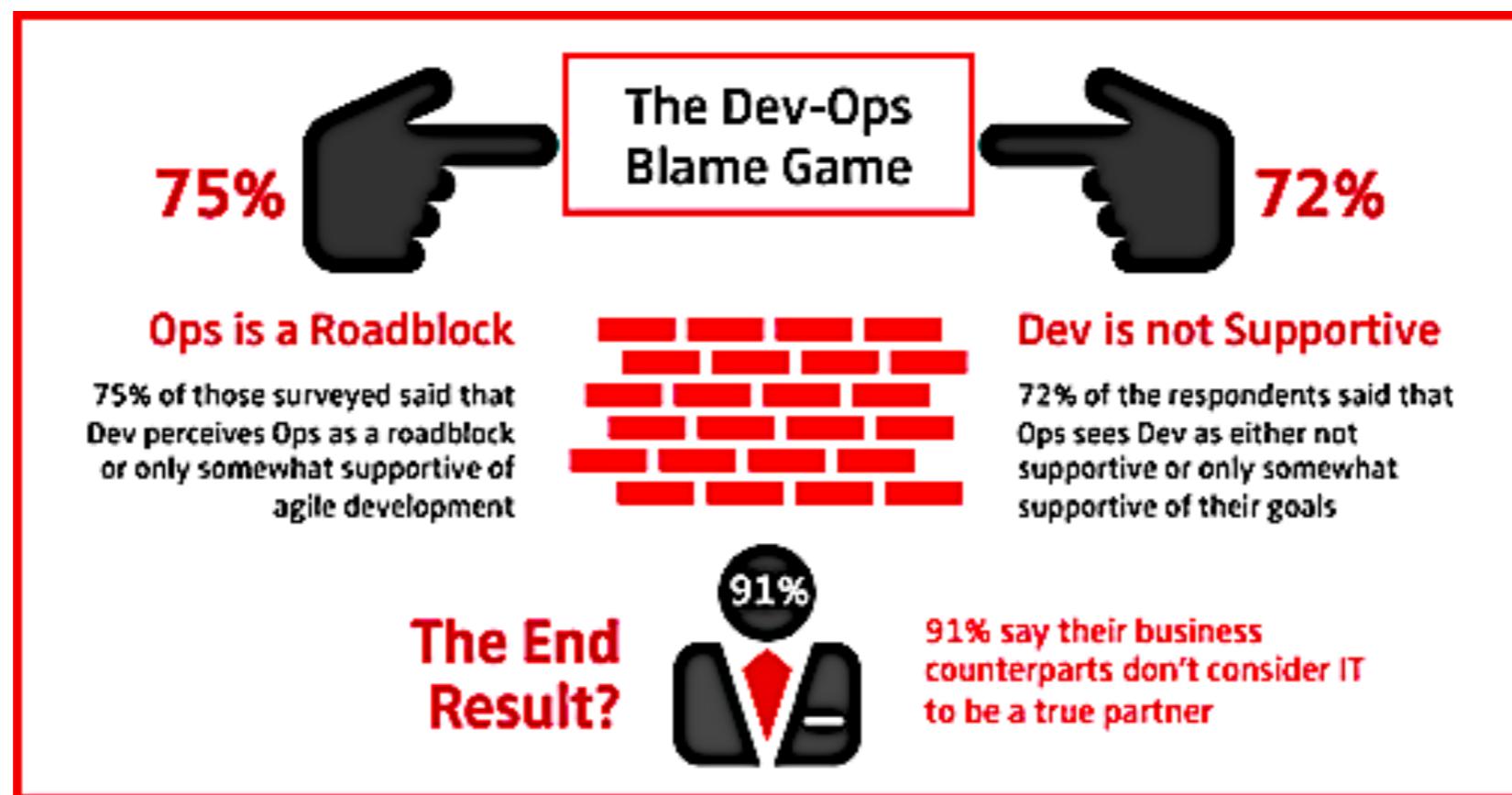
# Iterative/Agile development

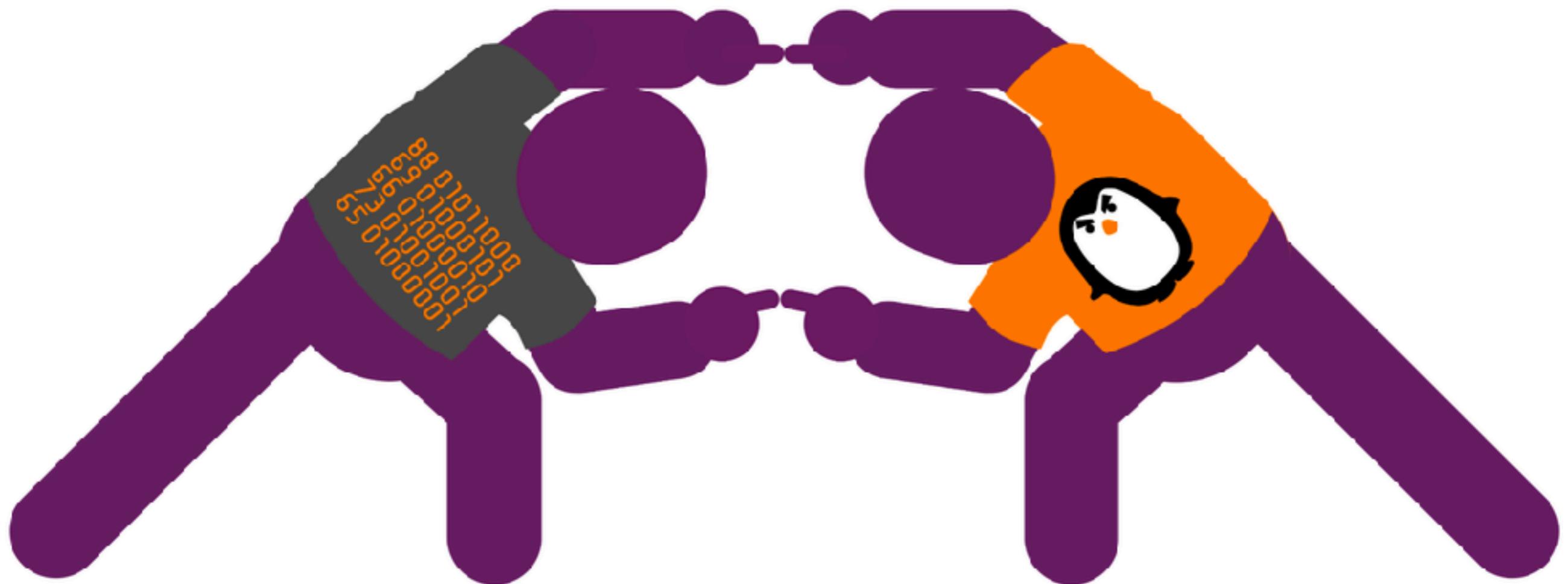


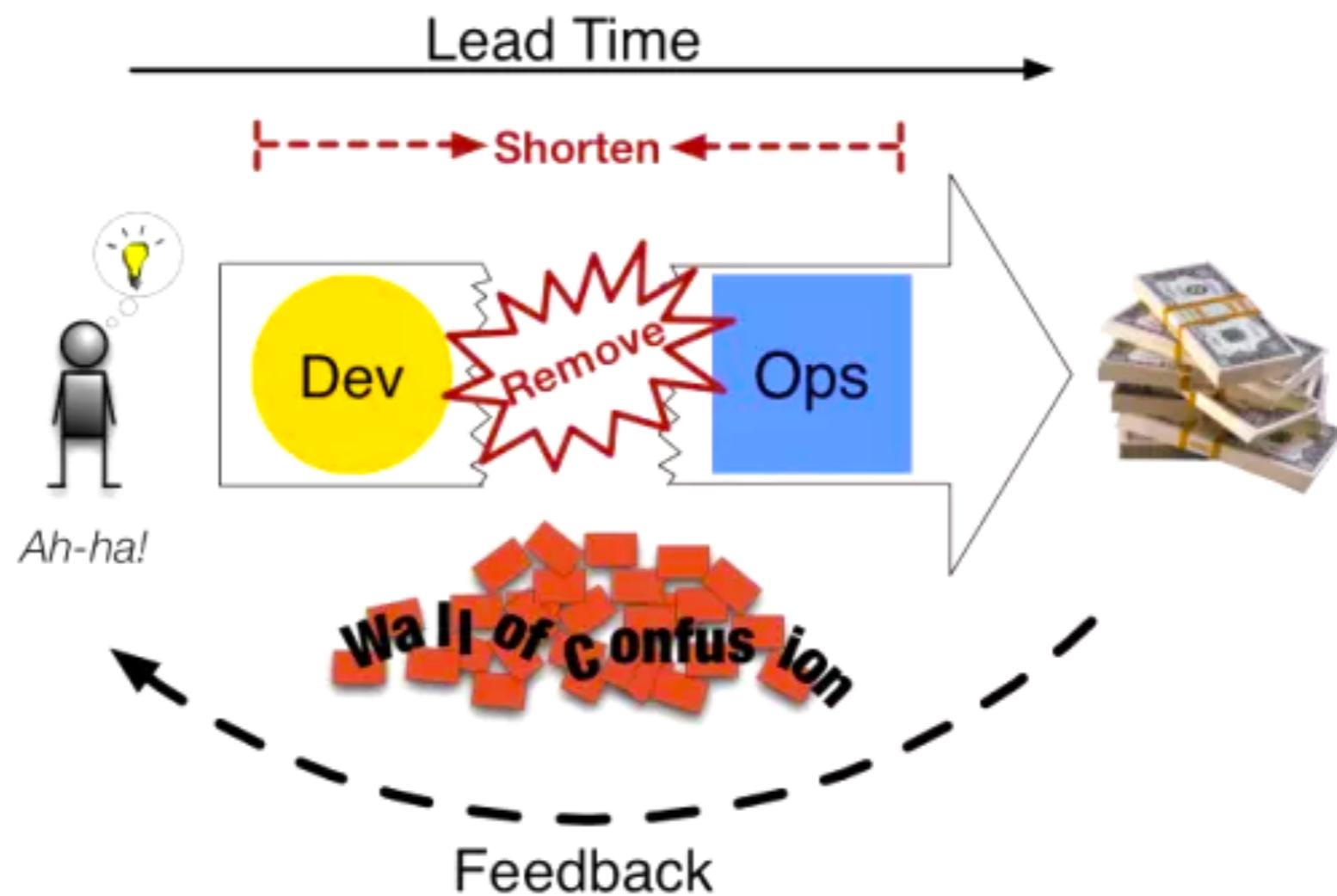
# Rise of DevOps

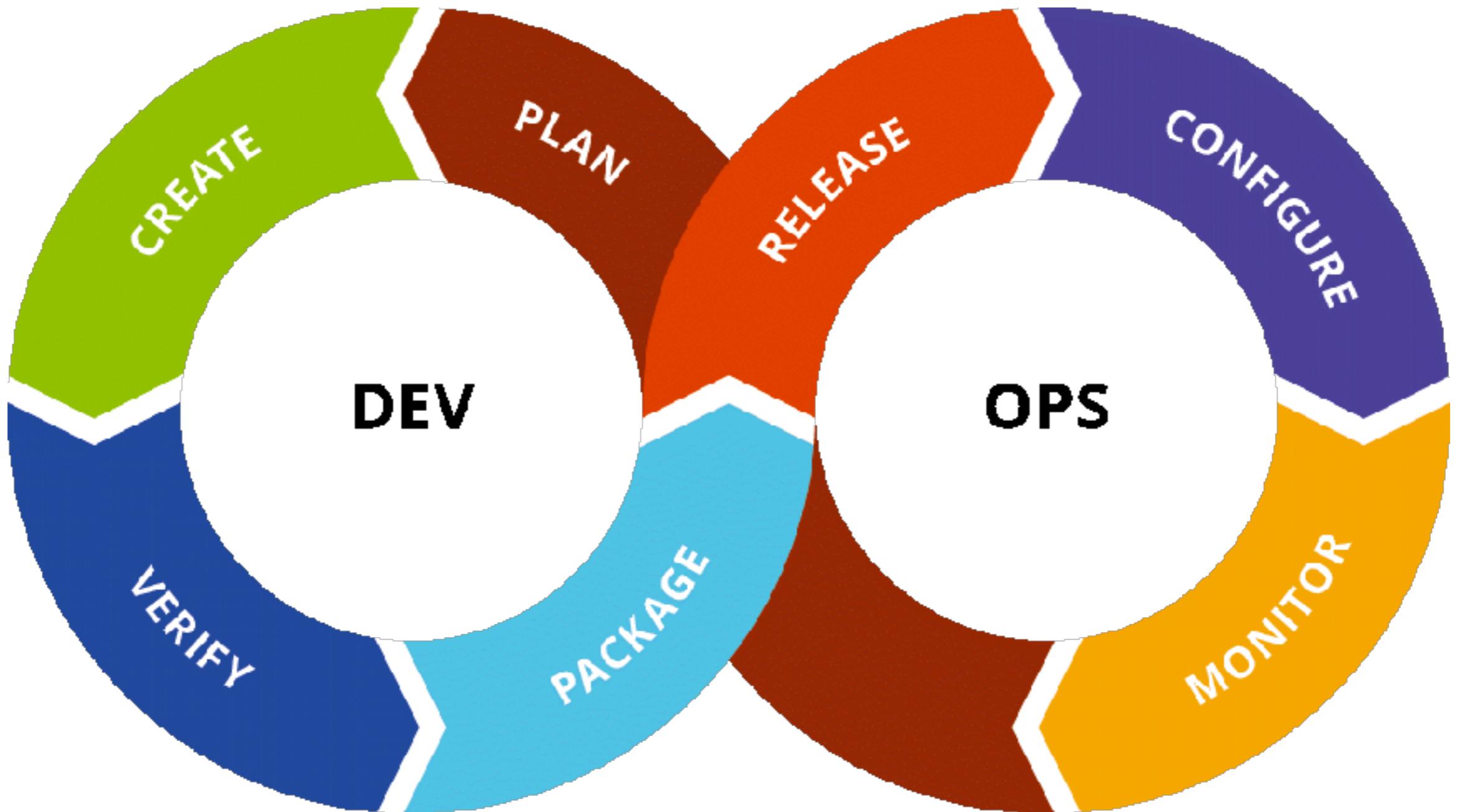












**DEV**

**OPS**

 **Application Performance**

Decrease latency by using APM Tools.

 **End User Analytics**

Monitor end user latency and check device performance

 **Quality Code**

Ensure deployments don't degrade performance

 **Code-Level Errors**

Lower MTTR by finding error root causes



 **Application Availability**

Make sure Uptime and SLAs are in order

 **Application Performance**

Solve problems by correlating infrastructure and application metrics

 **End User Complaints**

Fix problems before end users complain

 **Performance Analytics**

Use automatically generated baselines to focus troubleshooting



# DevOps ?

**"DevOps is**  
development  
and operations  
**collaboration"**

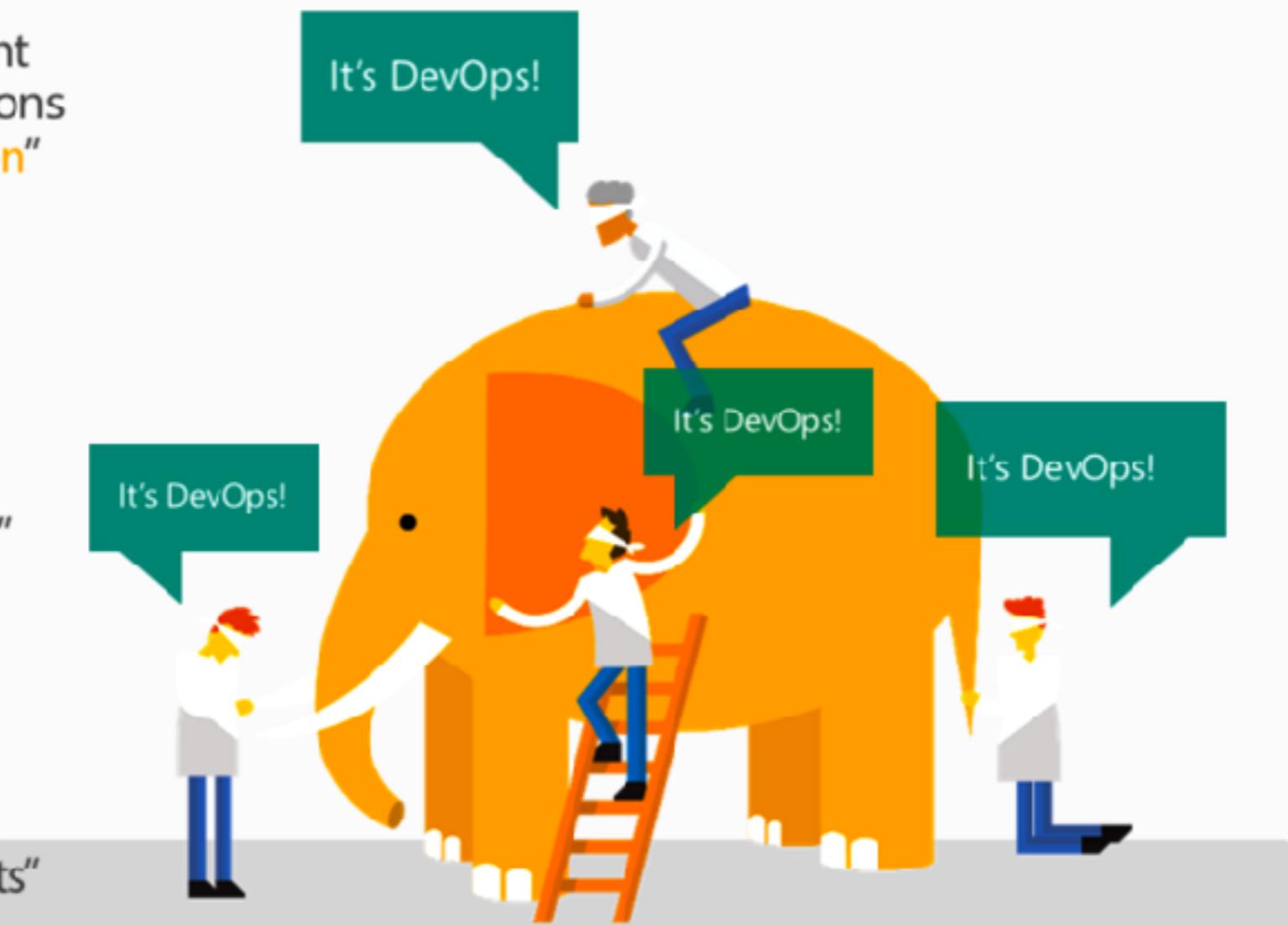
**"DevOps**  
is using  
**automation"**

**"DevOps**  
is **small**  
deployments"

**"DevOps is**  
treating your  
**infrastructure**  
**as code"**

**"DevOps**  
is feature  
**switches"**

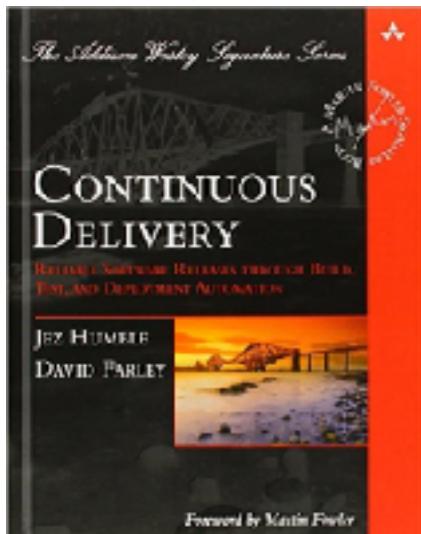
**"Kanban**  
for Ops?"



# DevOps ?

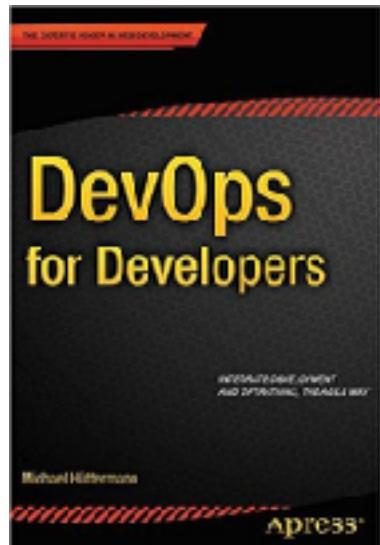
**“A movement of people who care about developing and operating reliable, secure, high performance systems at scale.”**

*- Jez Humble -*



# DevOps ?

“A mix of patterns intended to **improve collaboration** between development and operations. DevOps addresses **shared goals and incentives** as well as **shared processes and tools.**”

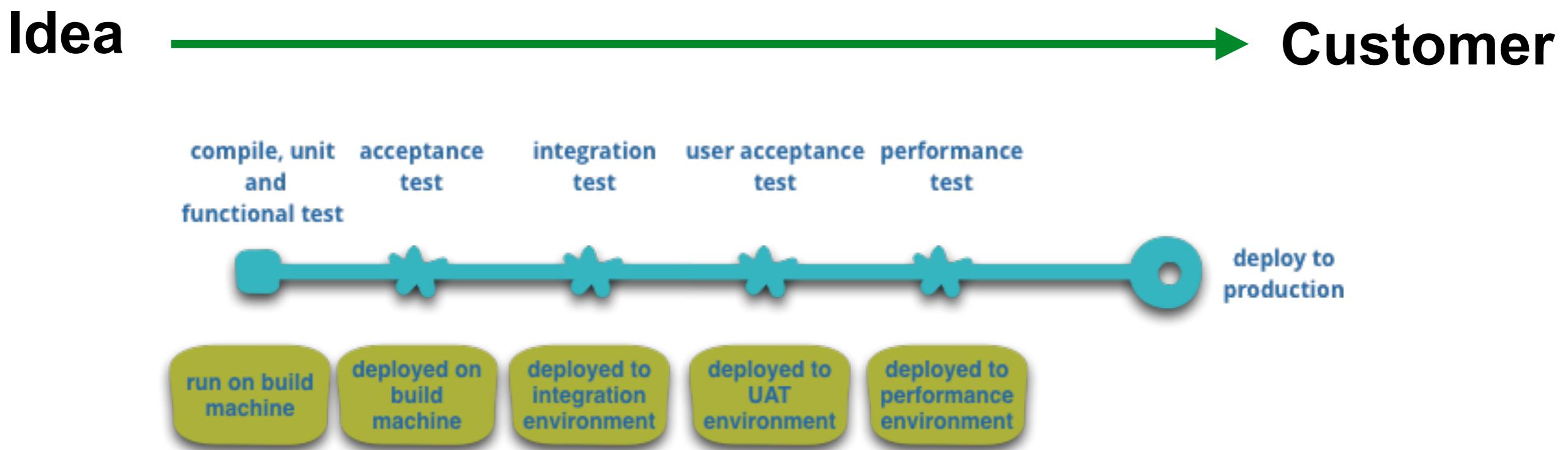


*- Michael Huttermann -*

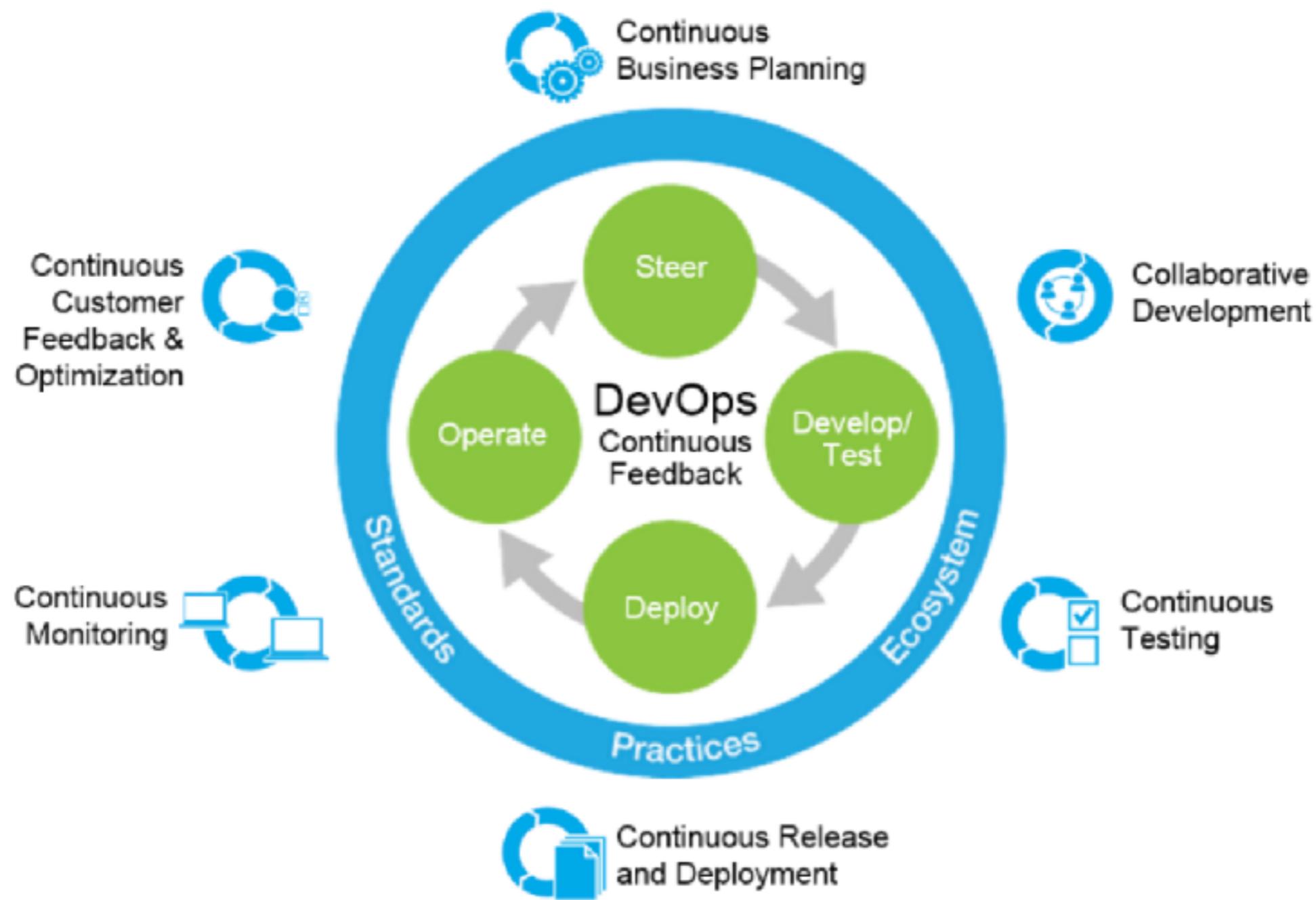


# Goal of DevOps

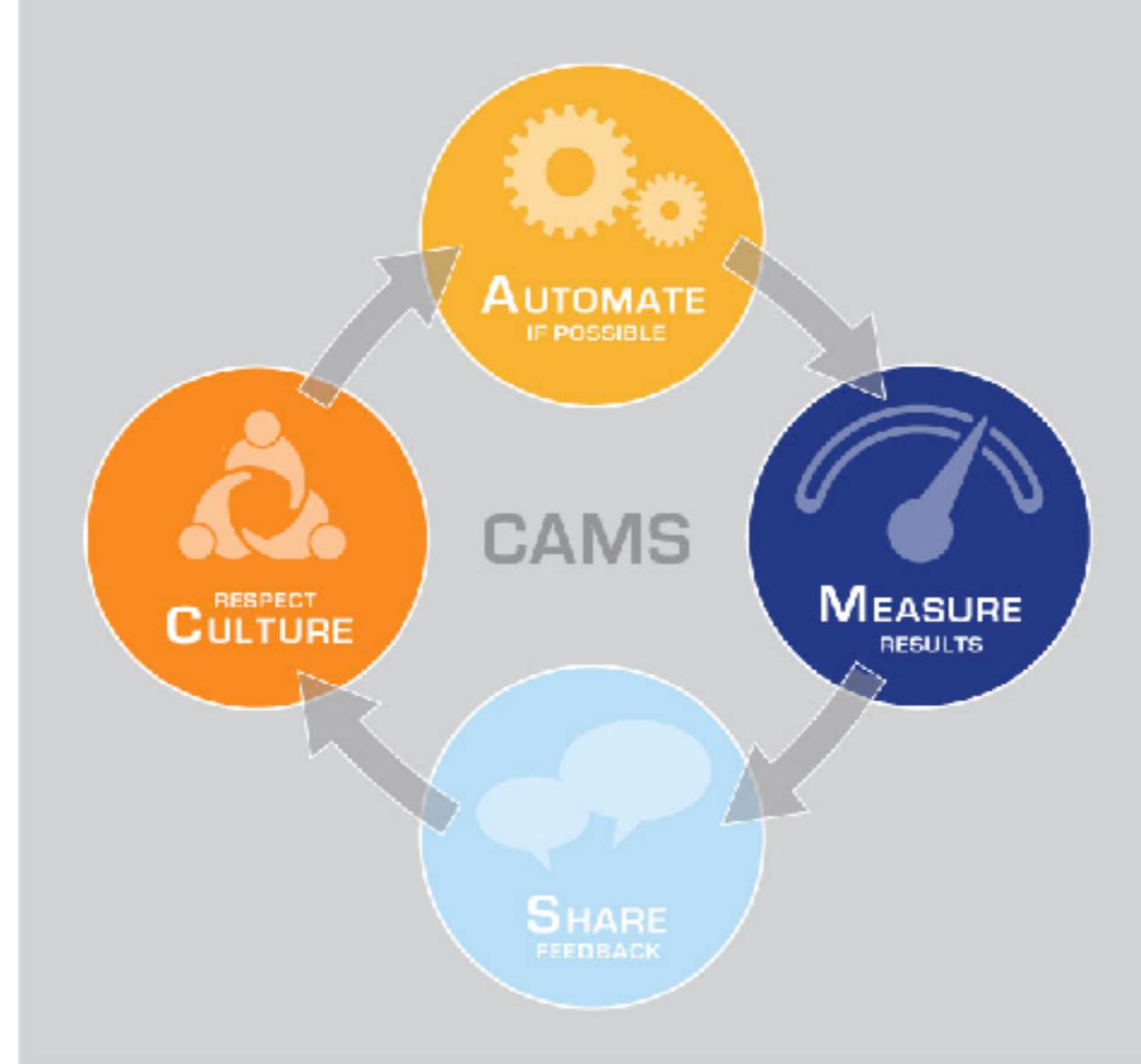
“Improve the delivery of value for Customer and Business”



# DevOps Life Cycle



# DevOps Principles



# DevOps Principles

**Culture => People, Process, Tools**

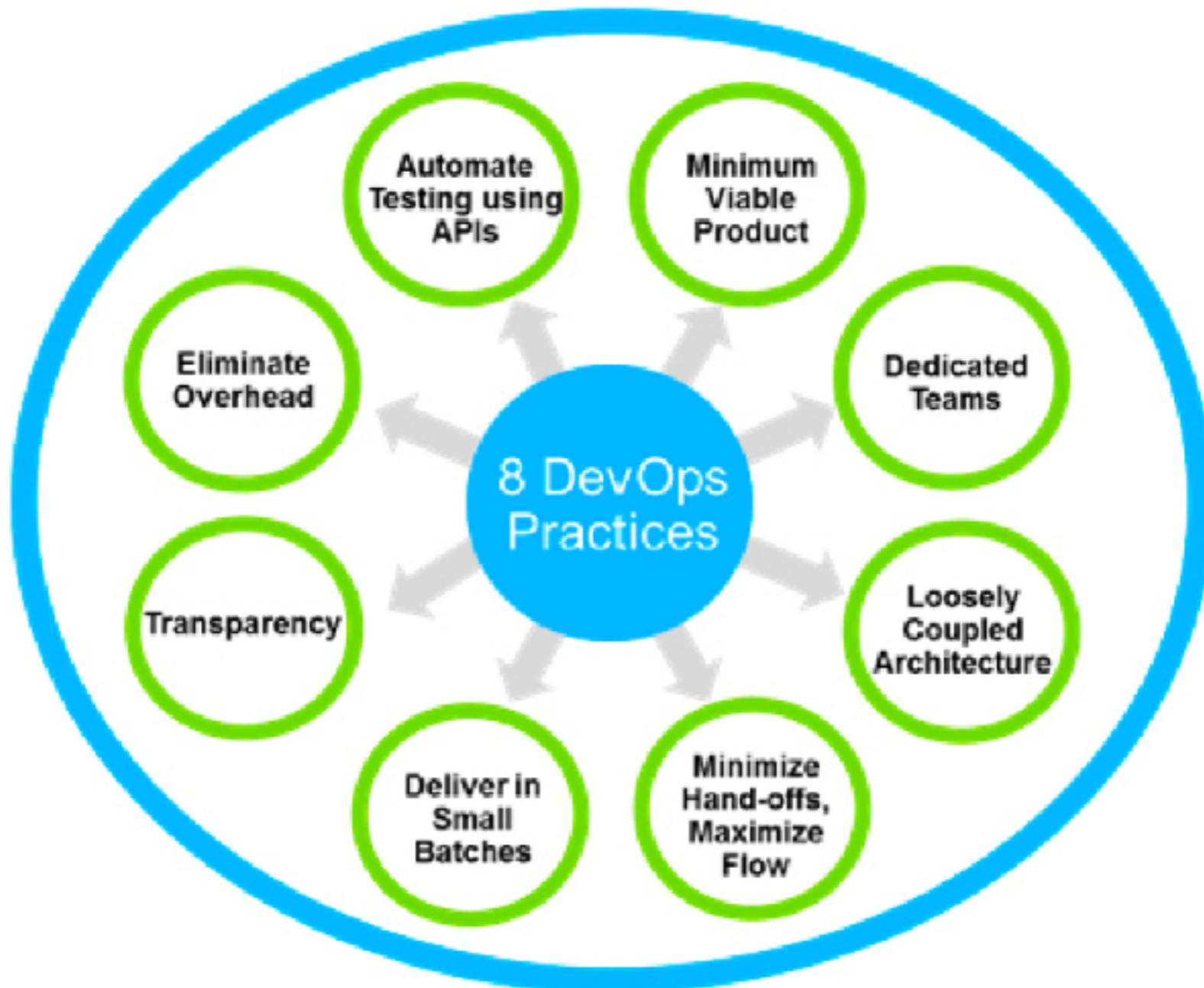
**Automation => Infrastructure as Code**

**Measurement => Measure everything**

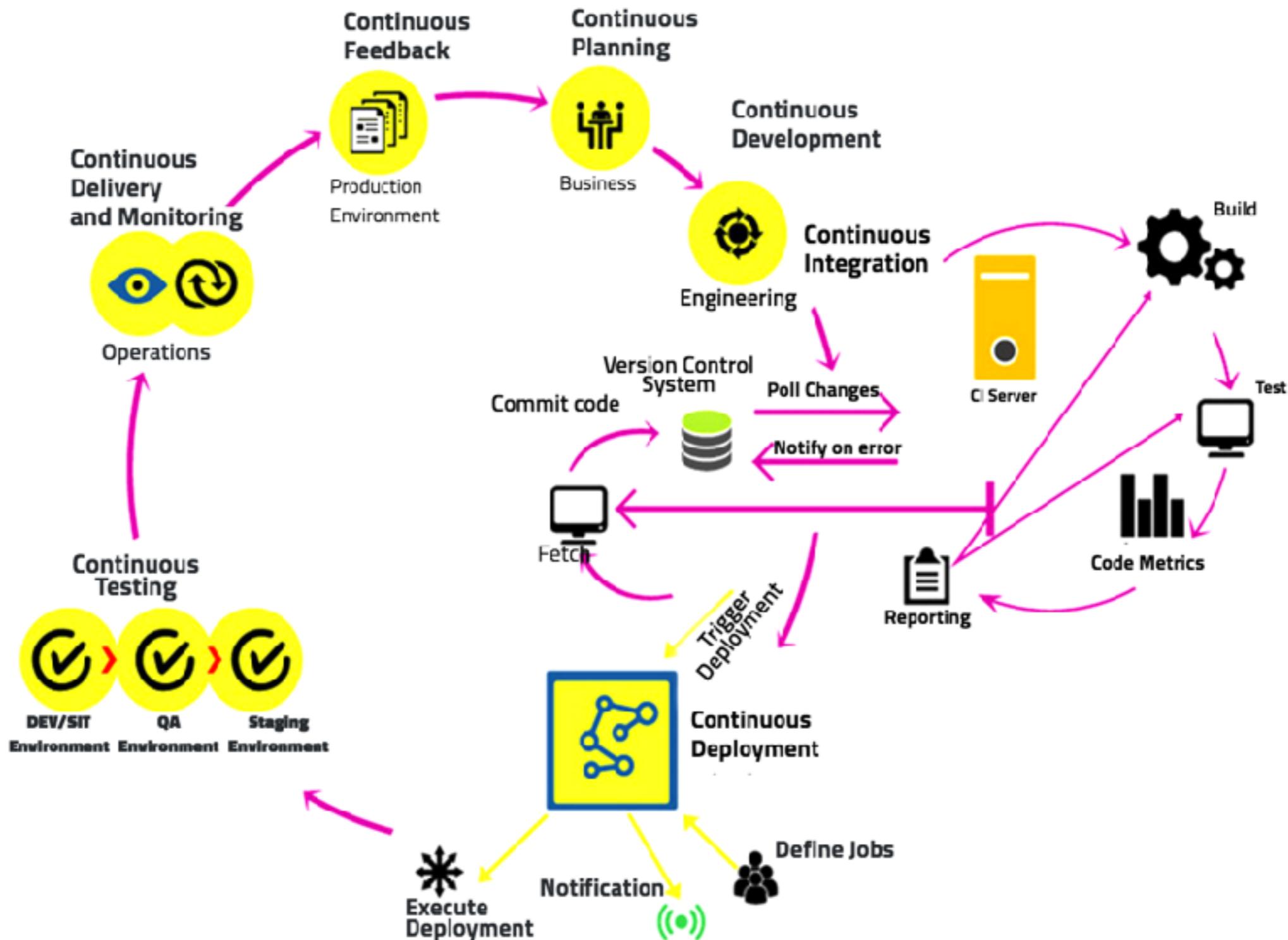
**Sharing => Collaboration/Feedback**



# DevOps Practices



# DevOps Practices



# **DevOps**

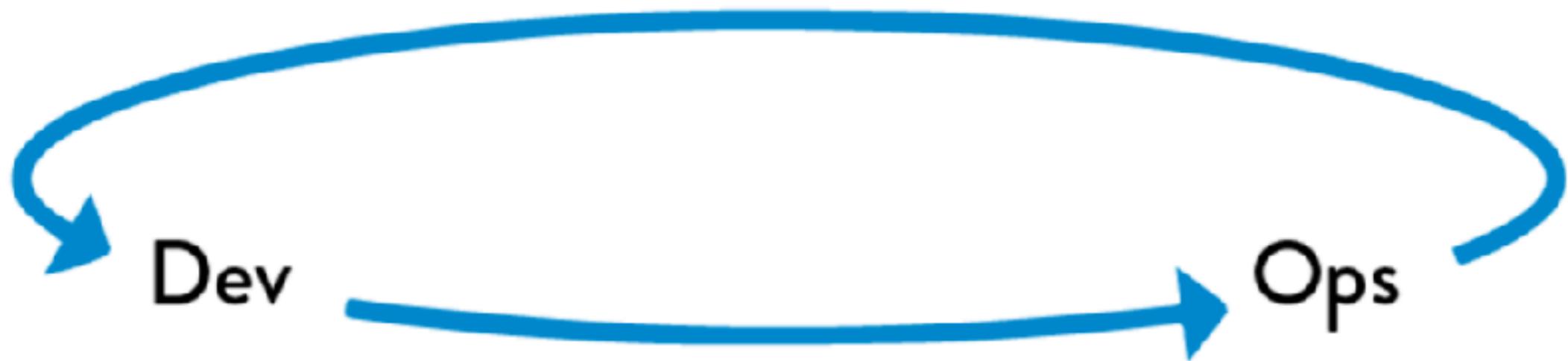
# **3 ways principle**



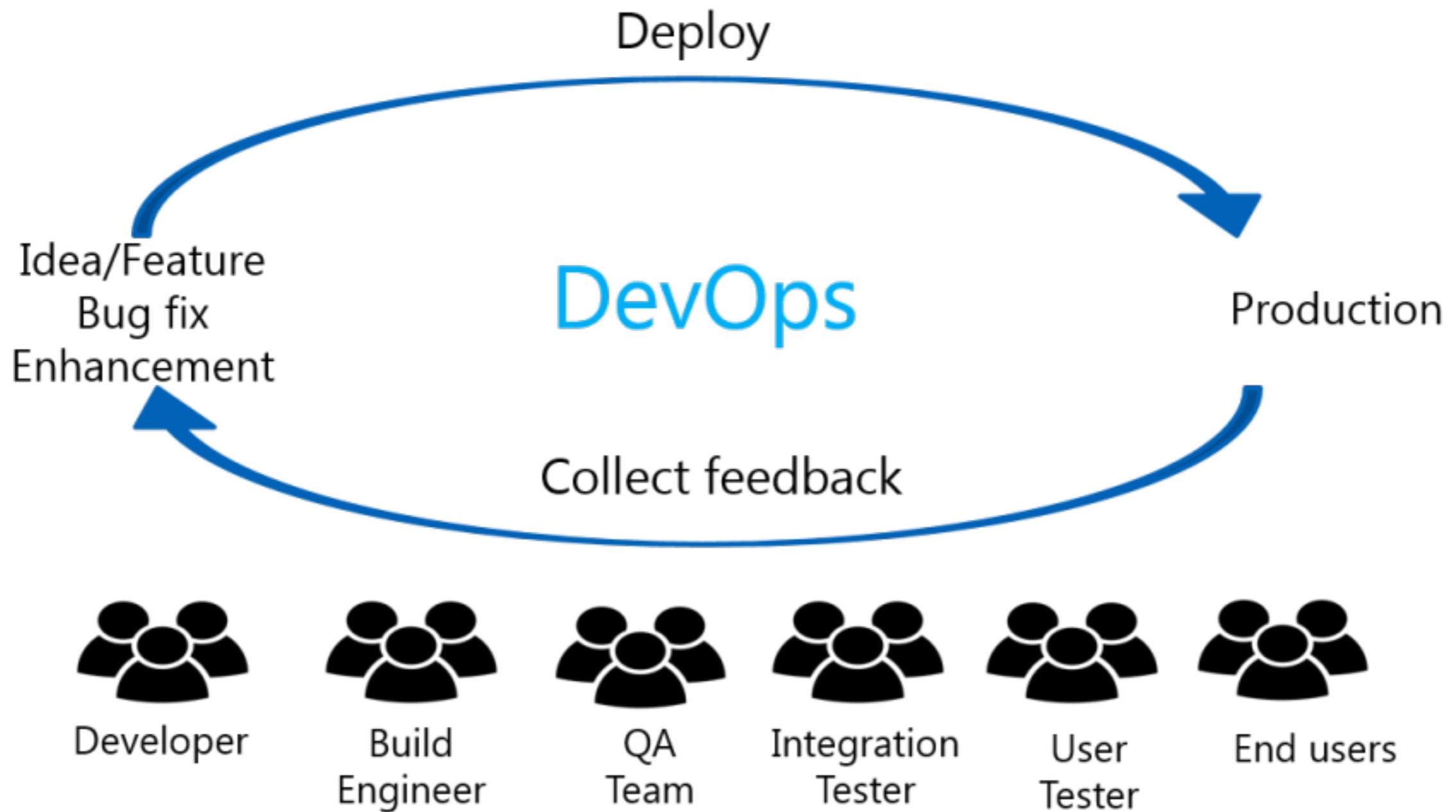
# Flow principle



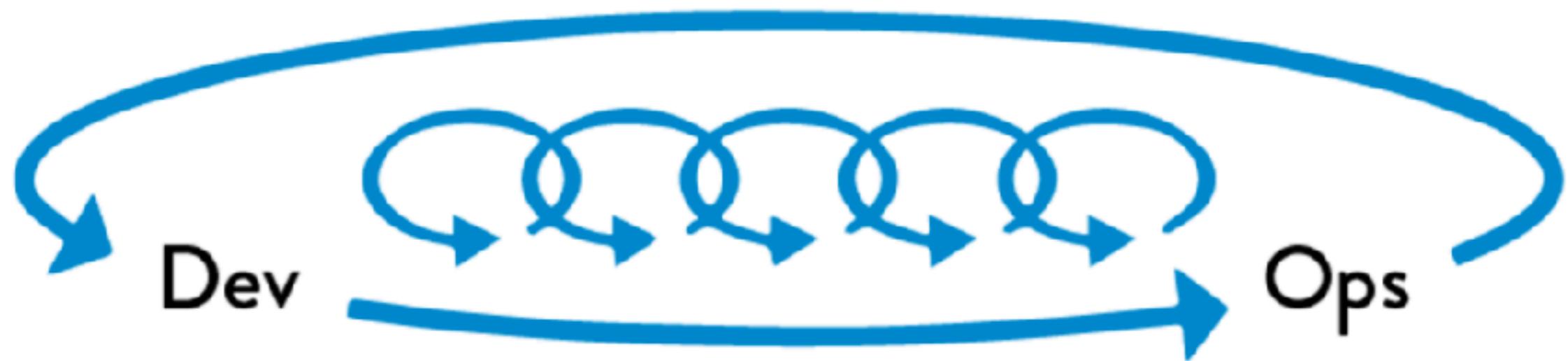
# Feedback principle



# Feedback principle

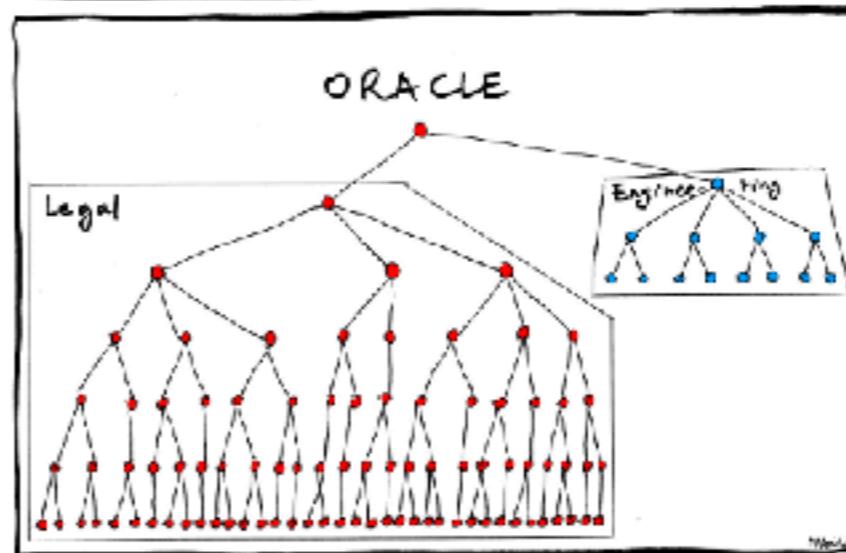
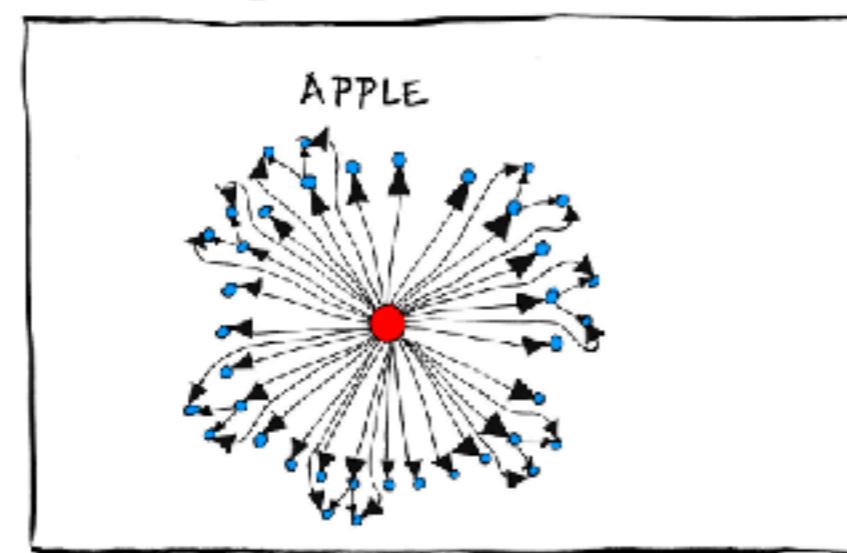
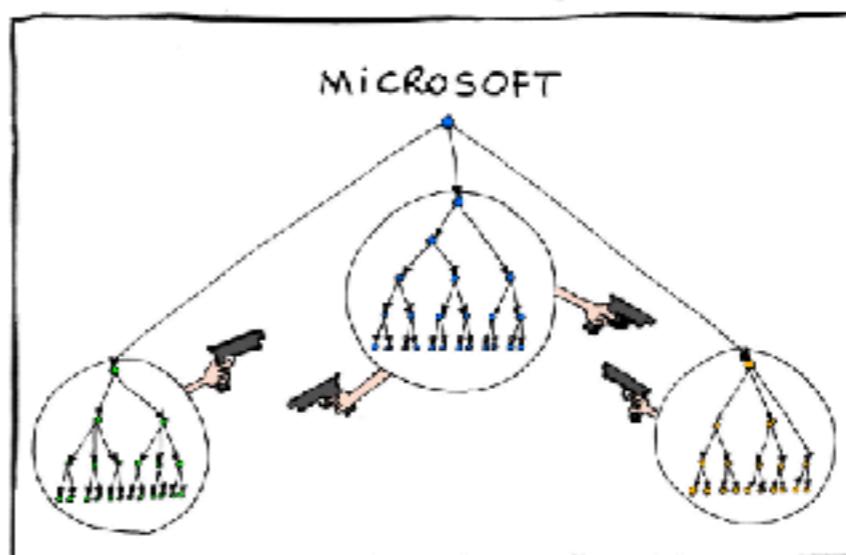
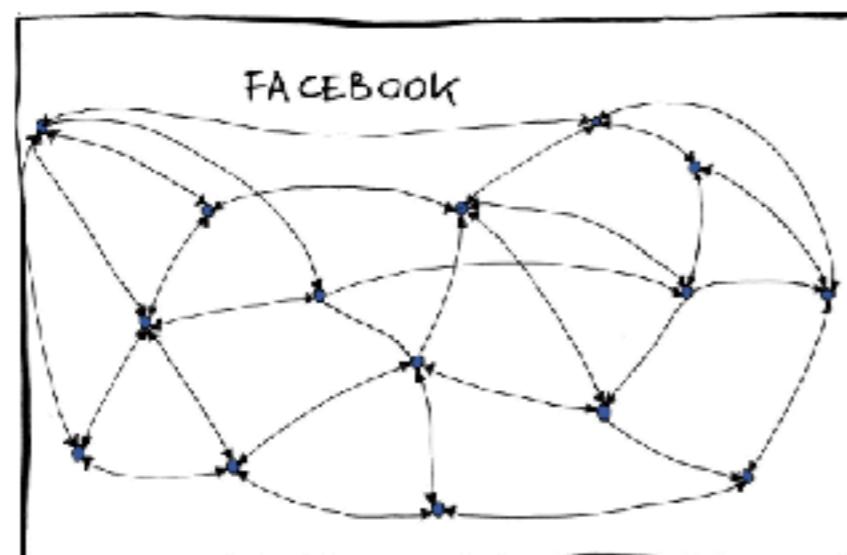
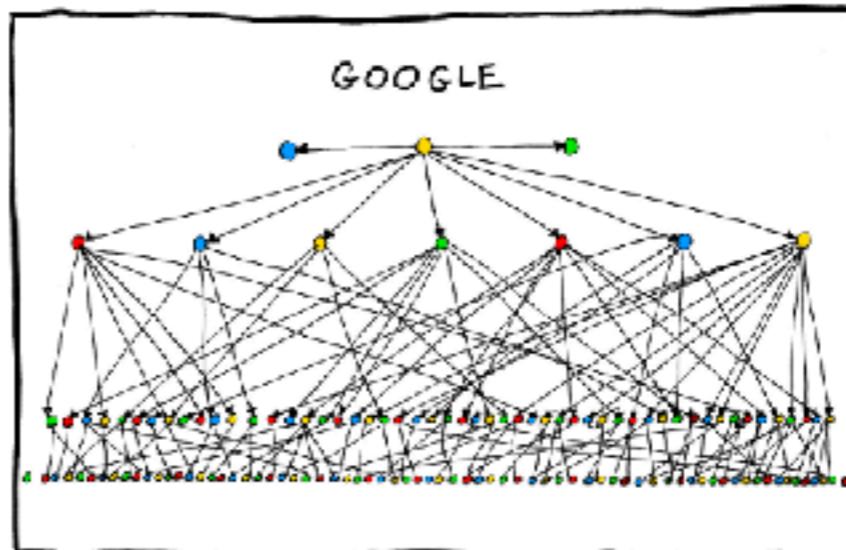
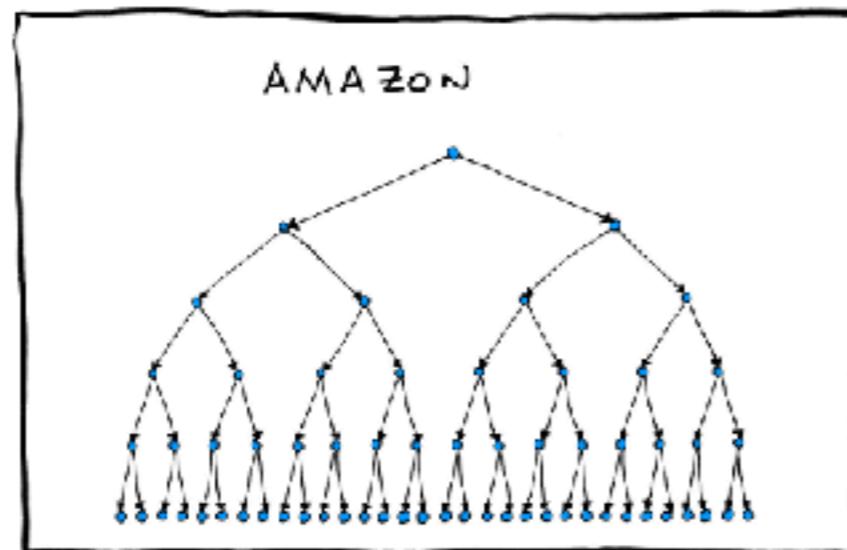


# Continuous learning principle



# All about Organization structure and culture





# **People -> Process -> Tool**



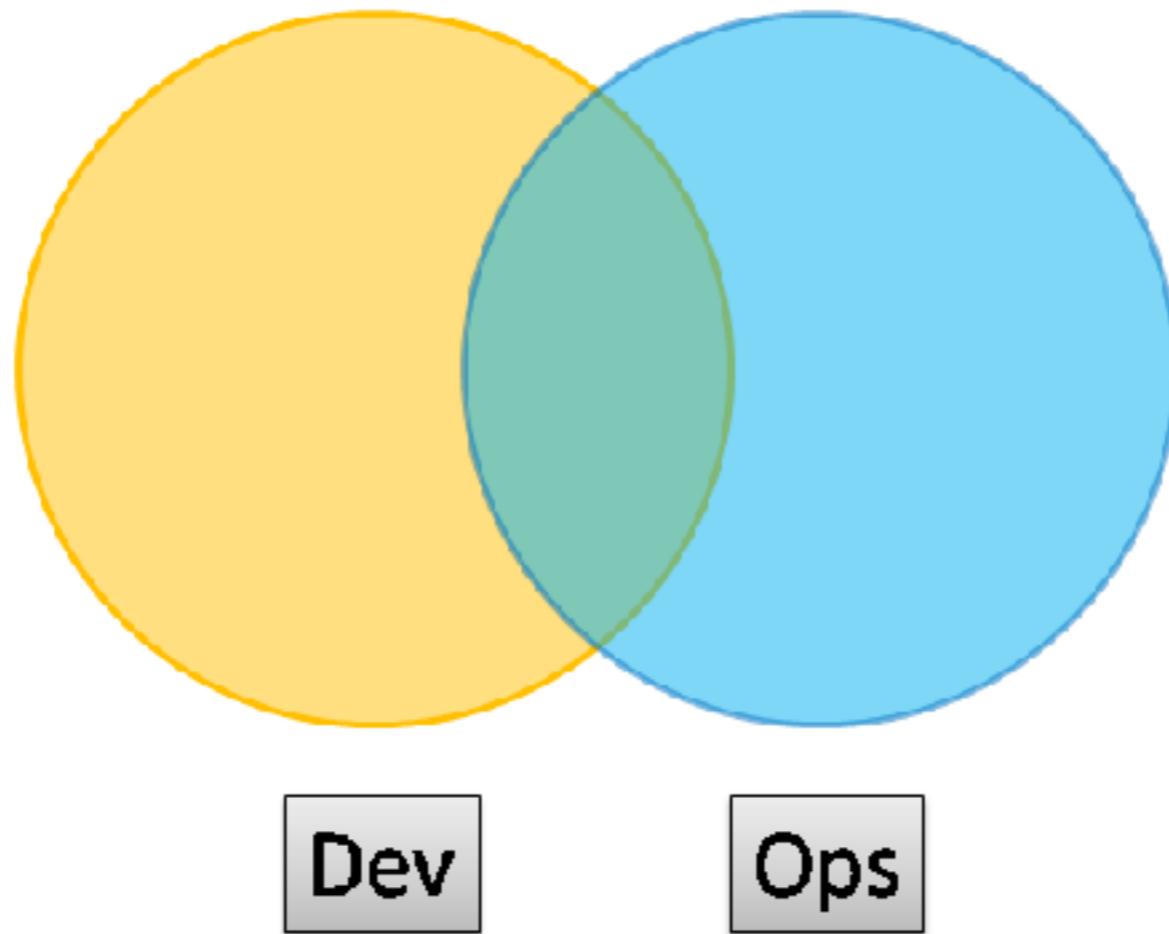
# Autonomous



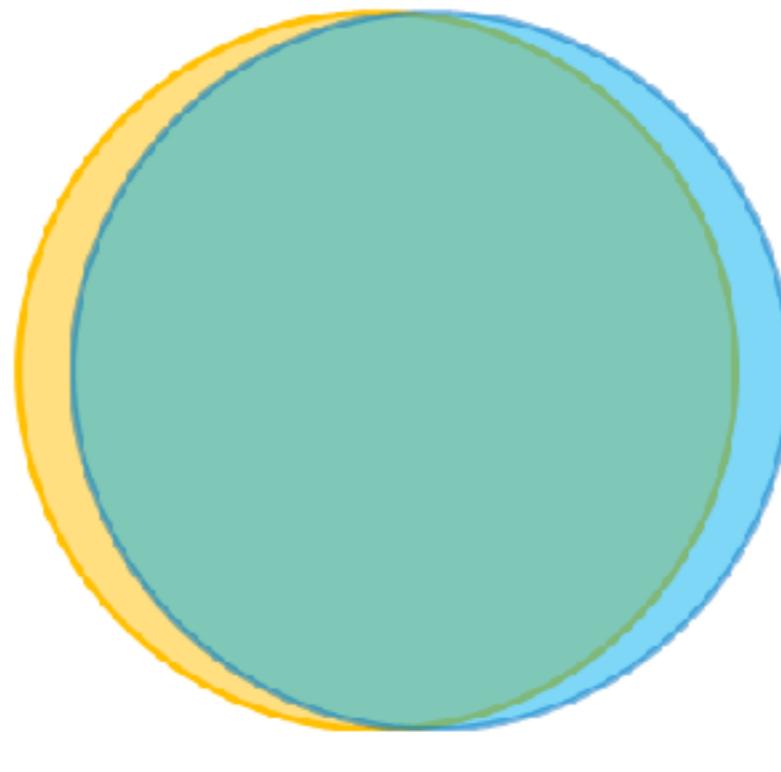
# DevOps Topologies



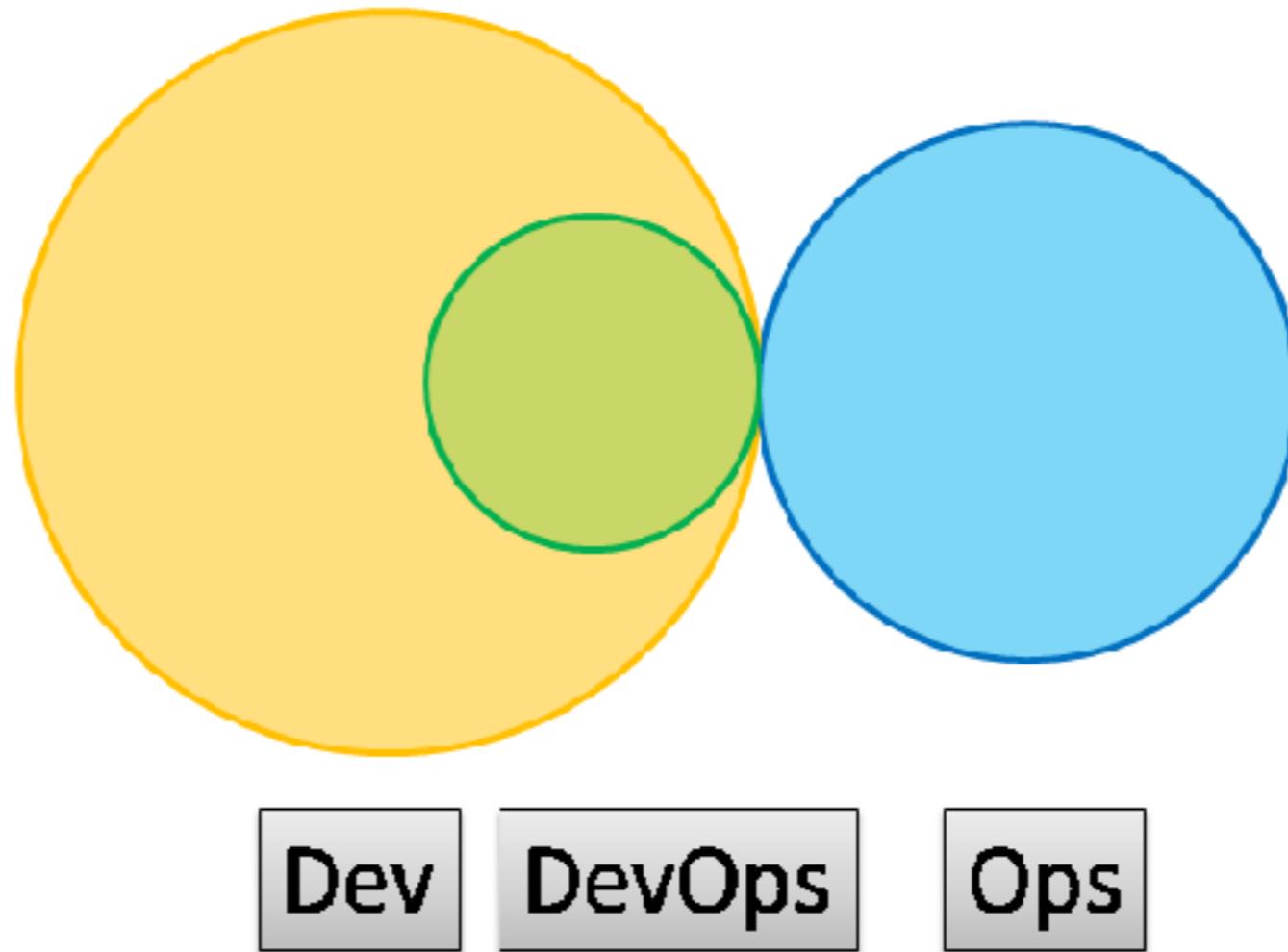
# Type 1 – Smooth Collaboration



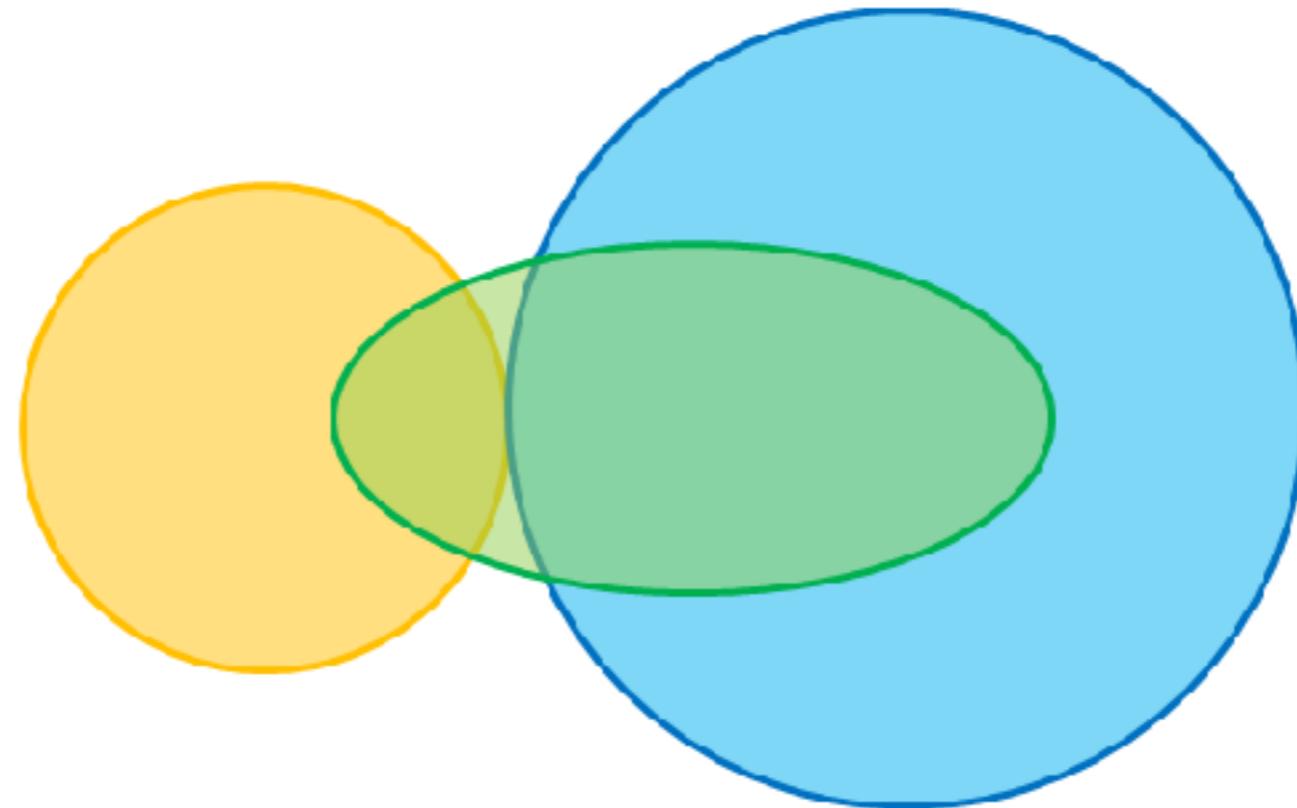
# Type 2 – Fully Embedded



# Type 3 – Infrastructure-as-a-Service



# Type 4 – DevOps-as-a-Service



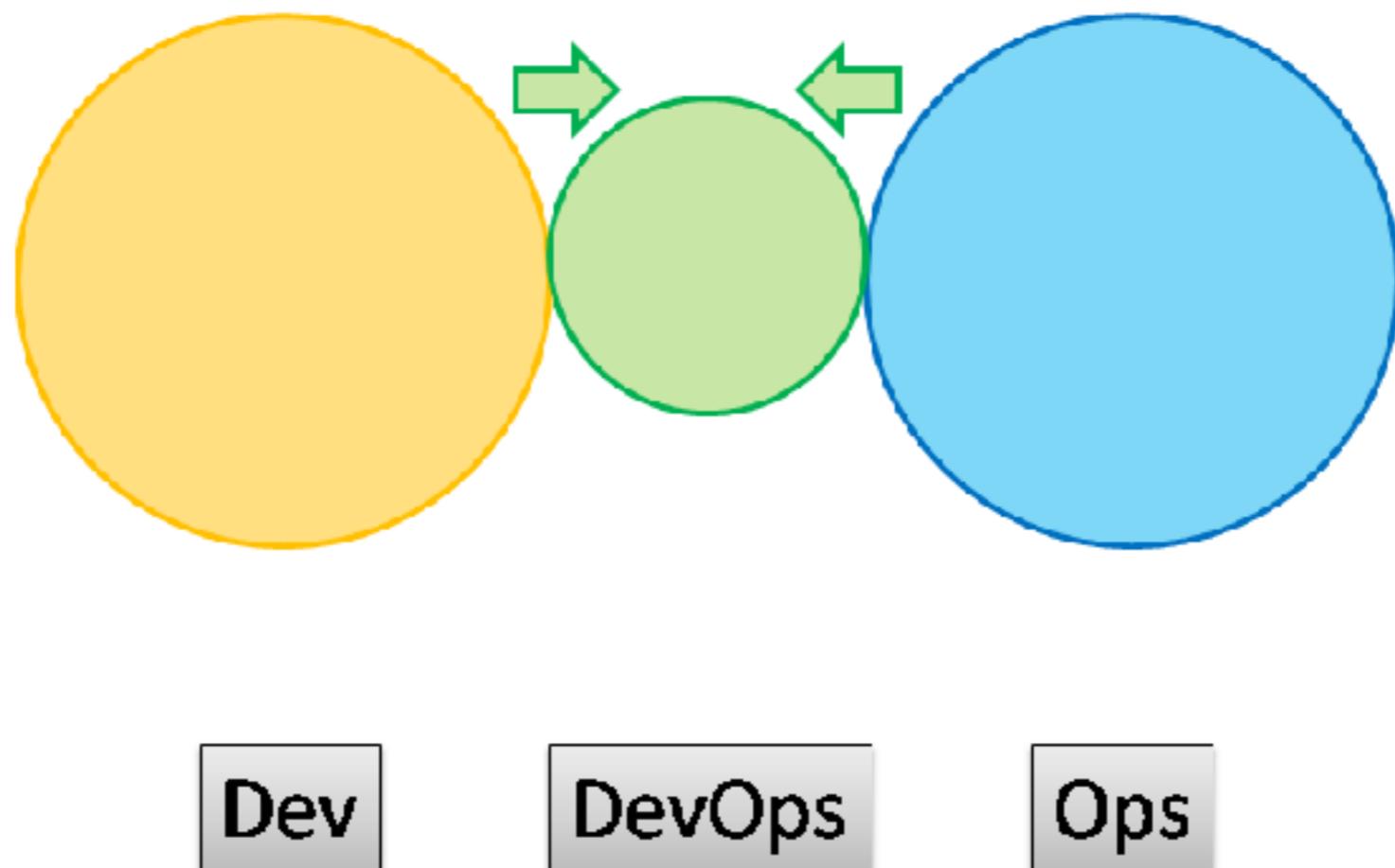
Dev

DevOps

Ops



# Type 5 – Temporary DevOps Team



# DevOps Tools

**PERIODIC TABLE OF DEVOPS TOOLS (V2)**

EMBED DOWNLOAD ADD

The Periodic Table of DevOps Tools is a grid-based visualization of various software tools used in the DevOps ecosystem. The table is organized into four main columns based on tool type:

- OpenSource**: Tools 1 through 20.
- SCM**: Tools 21 through 30.
- Build**: Tools 31 through 40.
- Database Mgmt**: Tools 41 through 50.

Each cell contains the tool name, its primary function, and its license type (Os, Fr, Fm, Pd, En). The table also includes a legend for tool categories and a color-coded grid for each row.

Category	Tool	Function	License
OpenSource	Gh	Github	Os
SCM	Gt	Git	Os
Build	Dm	Drone	En
Database Mgmt	Bb	Bitbucket	Fm
SCM	Lb	Liquibase	Os
CI			
Deployment			
Config / Provisioning			
Containerization			
Release Mgmt			
Collaboration			
Security			
BI / Monitoring			
Logging			
Ch	Chef	Ansible	En
Pu	Puppet	Salt	En
An	Ansible	Docker	Os
Sl	Salt	Azure	Os
Dk	Docker	Az	Os
Az	Azure		
Gc	Google Cloud Platform		
Rs	Rackspace		
Op	OpenStack		
Ds	Swarm		
Hr	Heroku		
Os	OpenShift		

**XebiaLabs**  
Deliver Faster

Follow @xebialabs

This is a smaller periodic table specifically for XebiaLabs, showing a subset of tools from the main table. It includes tools from categories like CI, Deployment, Configuration, Containerization, and Monitoring.

Tool	Function	License
Xlr	XL Release	Fr
Ur	UrbanCode Release	Fr
Bm	RMC Release Process	Fr
Hp	HP Cedar	Fn
Au	Atomic	Fn
Pi	Plutora Release	Fn
Sr	Serenity Release	Fn
Tfs	Team Foundation	Pd
Tr	Telco	Fm
Jr	Jira	Pd
Rf	HipChat	Fm
Sl	Slack	Fm
Fd	Flowdock	Fm
Pv	Putnall Tracker	Fm
Sn	ServiceNow	Fn
Ki	Kibana	Os
Nr	New Relic	Fm
Ni	Nagios	Os
Zb	Zabbix	Os
Dd	Datadog	En
Ei	Elasticsearch	Os
St	StackStorm	En
Sp	Splunk	En
Le	Logentries	Fm
Sl	Sumsologic	Fm
Ls	Logstash	Os
Gr	Graylog	Os
Sn	Smart	Os
Tr	Tripwire	Os
Ff	Fortify	En

<https://xebialabs.com/periodic-table-of-devops-tools/>



Microservices

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

# No DevOps Team

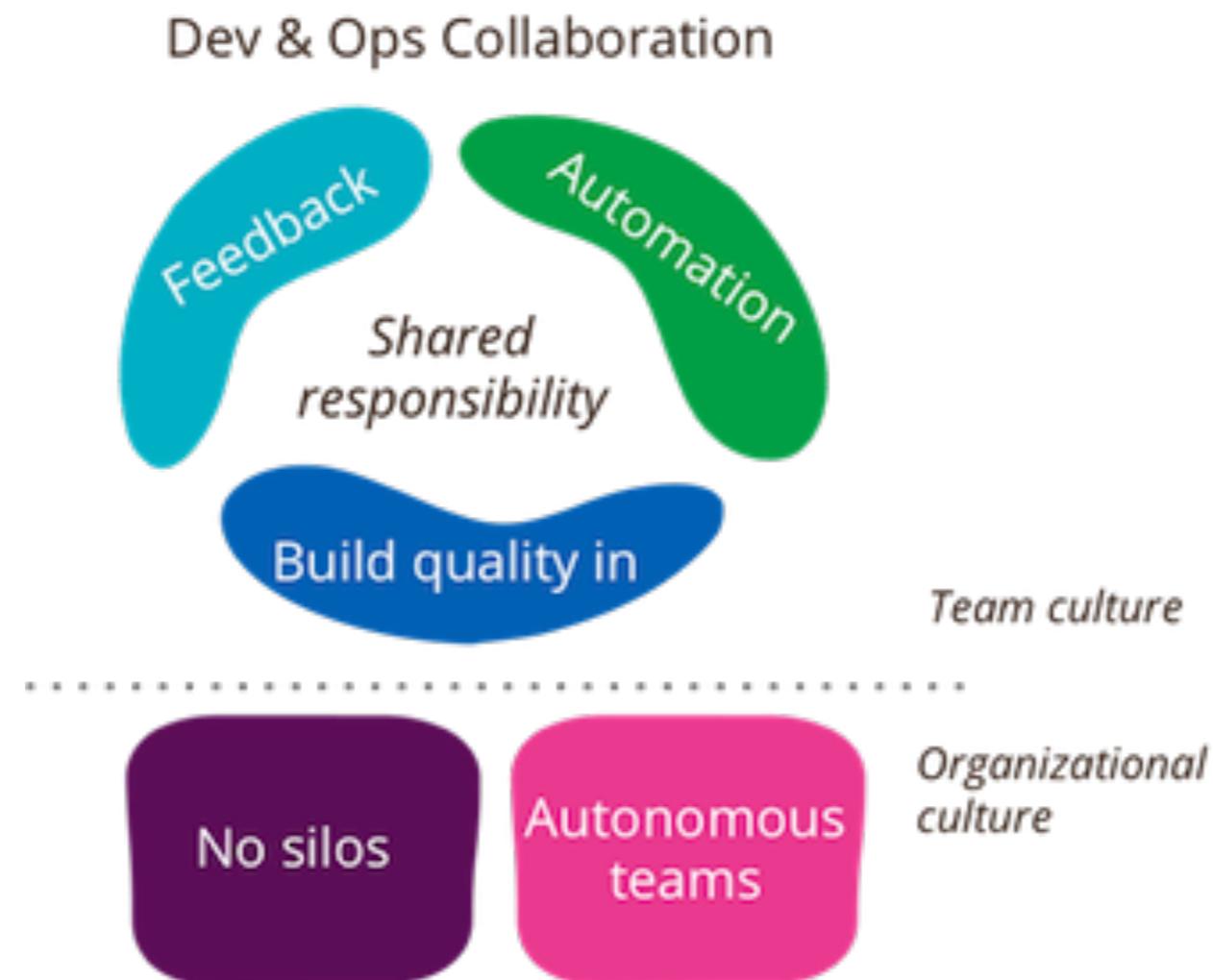
Problem department !!



**DevOps != Tools**  
**Tools enable DevOps**



# Team and Organization culture



<https://martinfowler.com/bliki/DevOpsCulture.html>





# DevOps success ?



# How do i know something is wrong ?

Missed deadline

Site is always down

Unhappy customers

Long waits for small changes or fixes

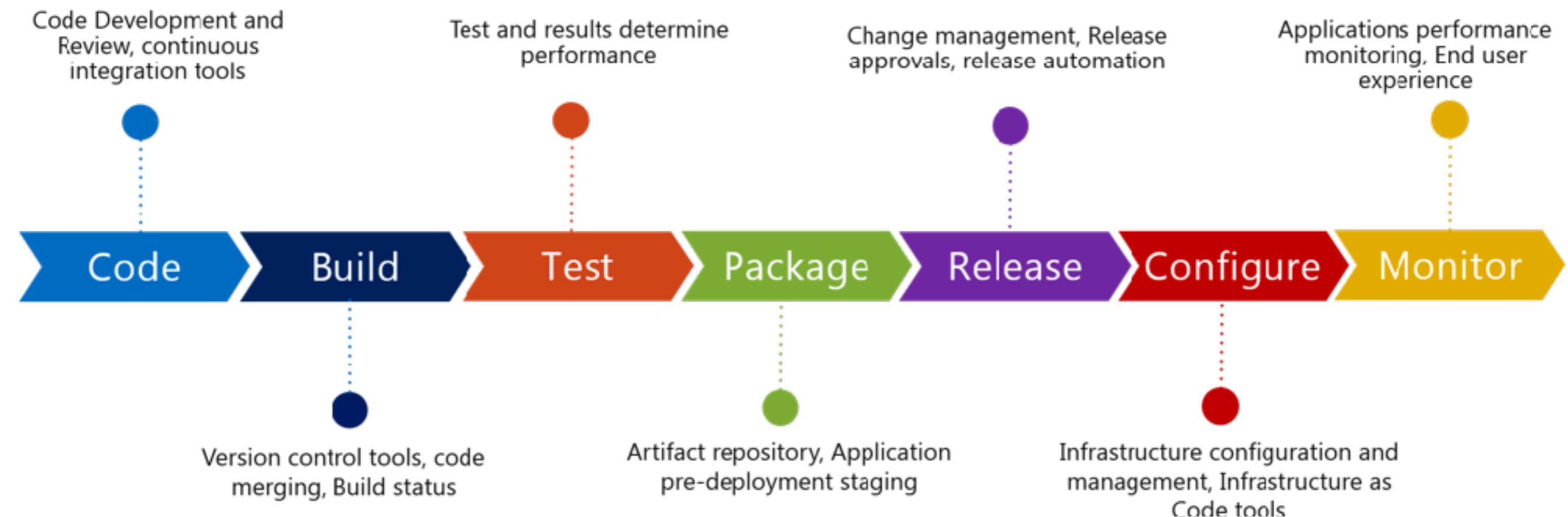
Changes cost too much



# What can i measure ?



# DevOps Process & Tools



# What can i measure ?

Mean Time to Recover/Repair (MTTR)

Mean Time to Detection (MTTD)

Change Lead Time

Change Failure Rate

Deployment or Change Frequency

Deployment Time

Percentage of successful deployments



# What can i measure ?

Application Usage and Traffic

Application Performance

Automated Test Pass (%)

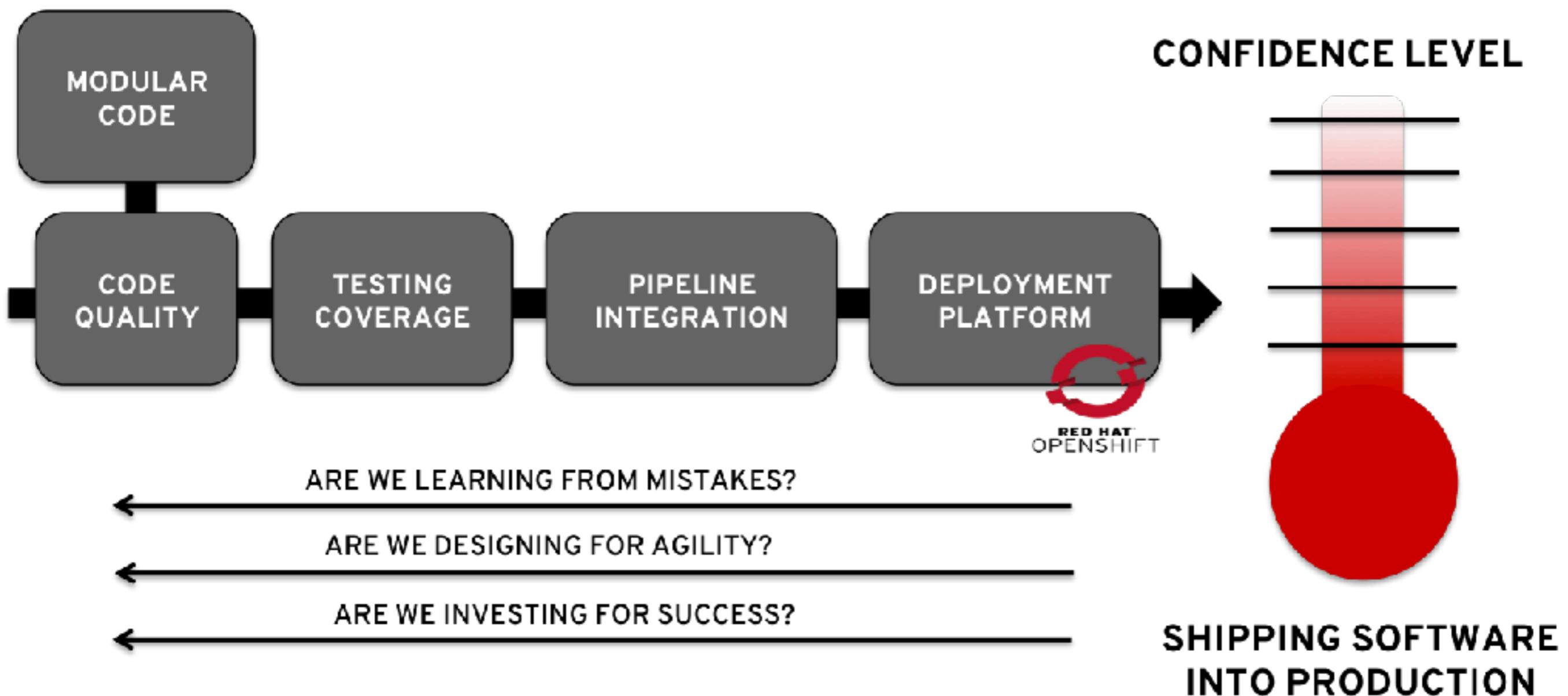
Defect Rate

Failed Deployments

Availability



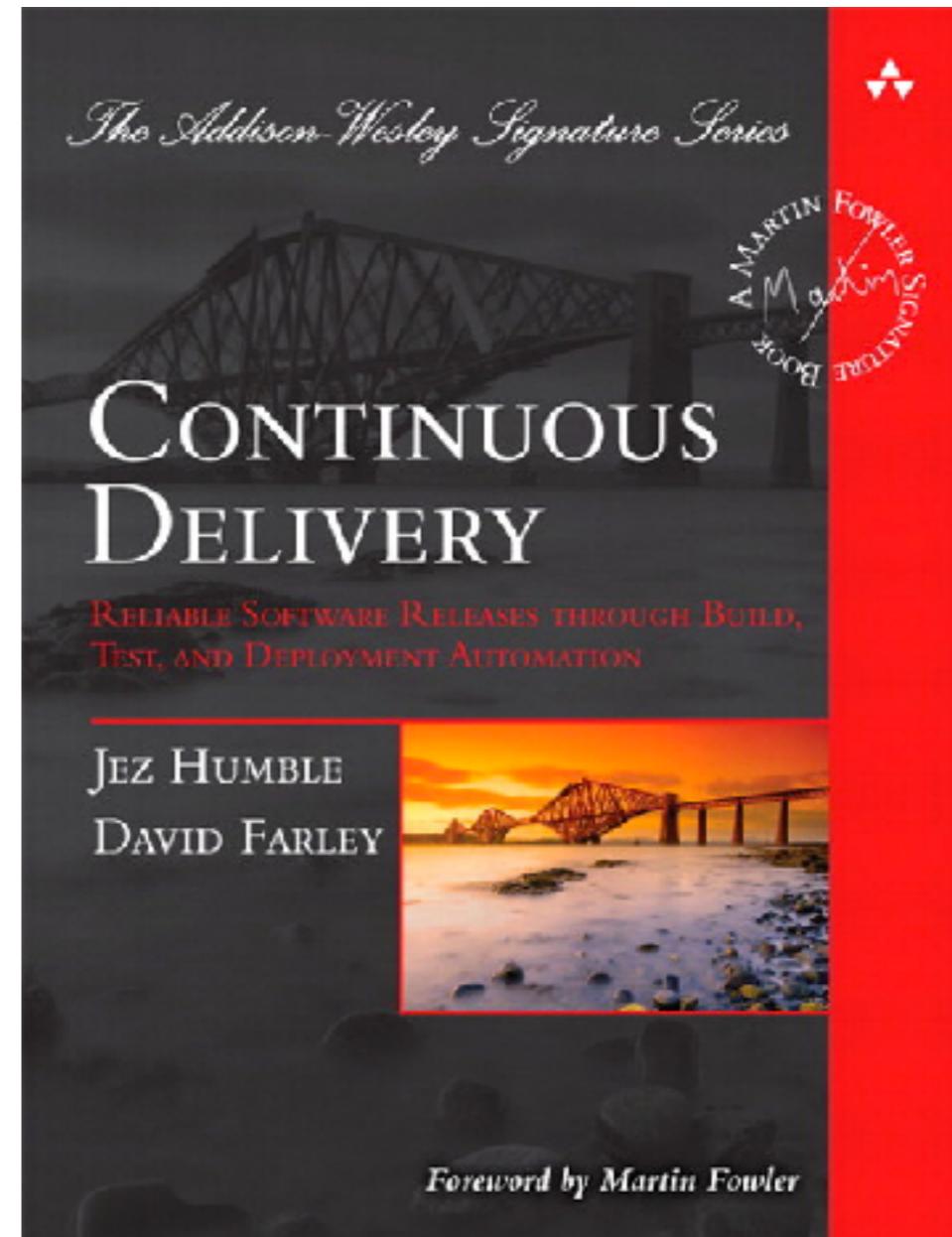
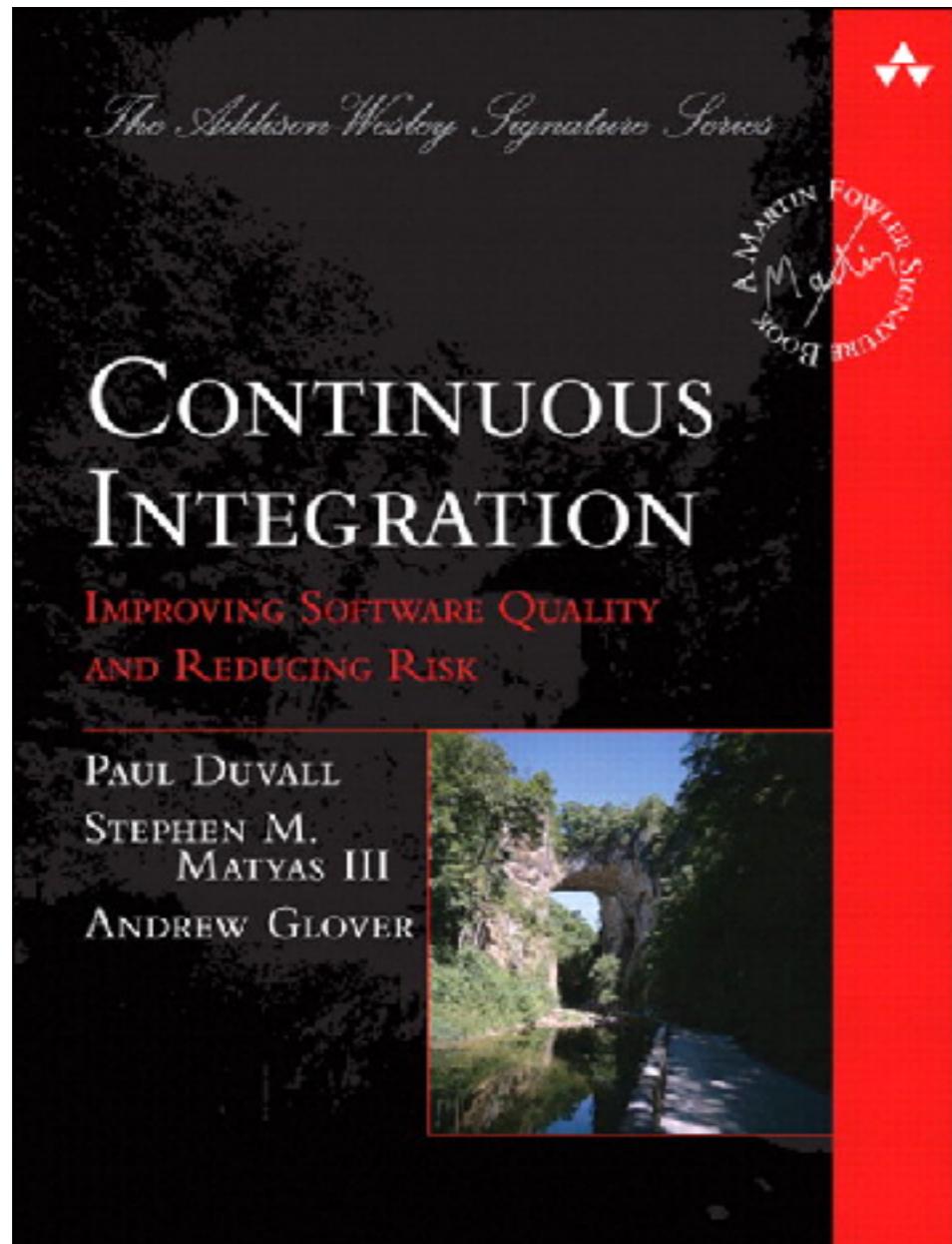
# What can i measure ?



# **Start with Continuous Integration Continuous Delivery**



# Improve quality and reduce risk



Microservices

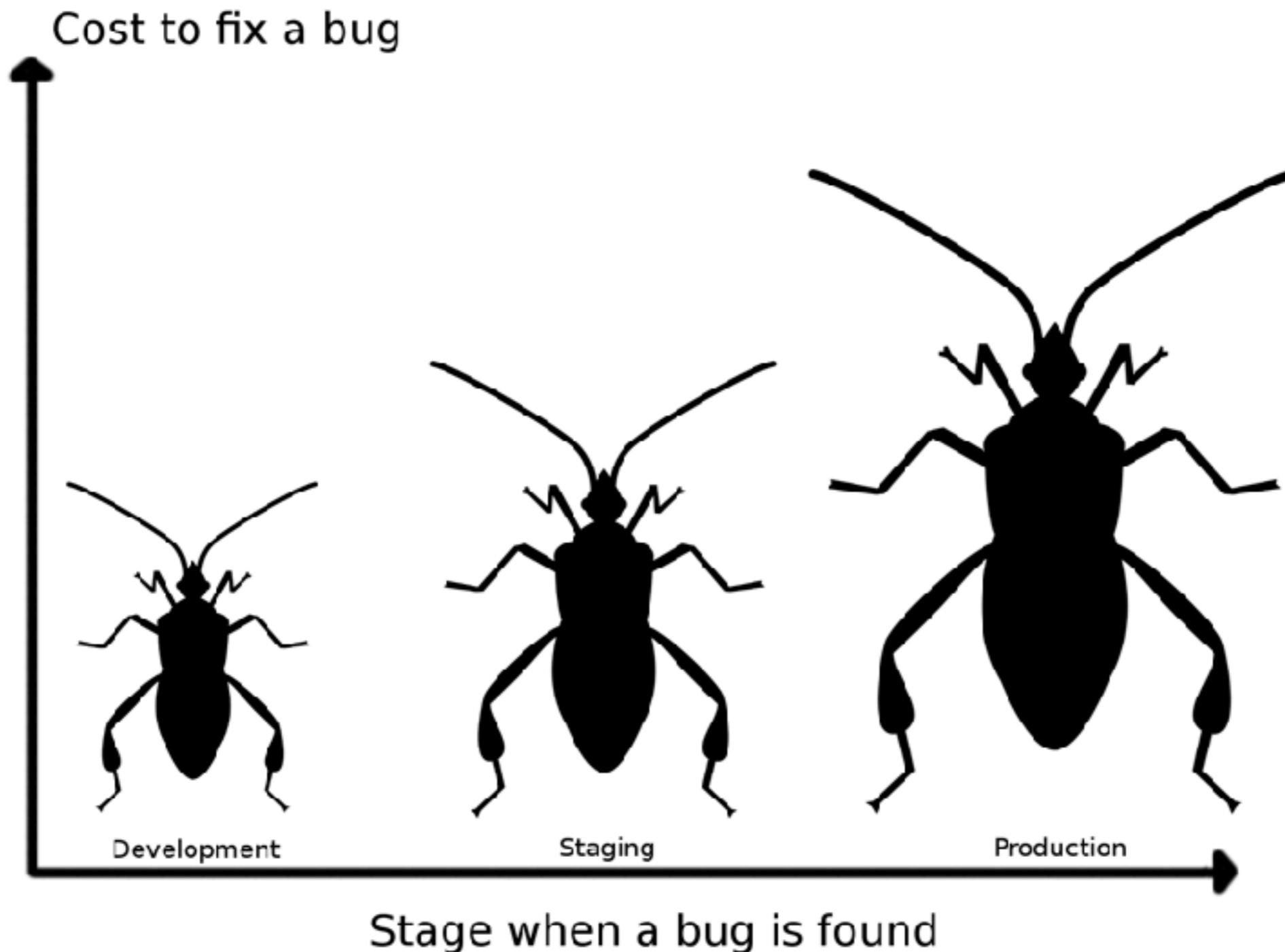
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

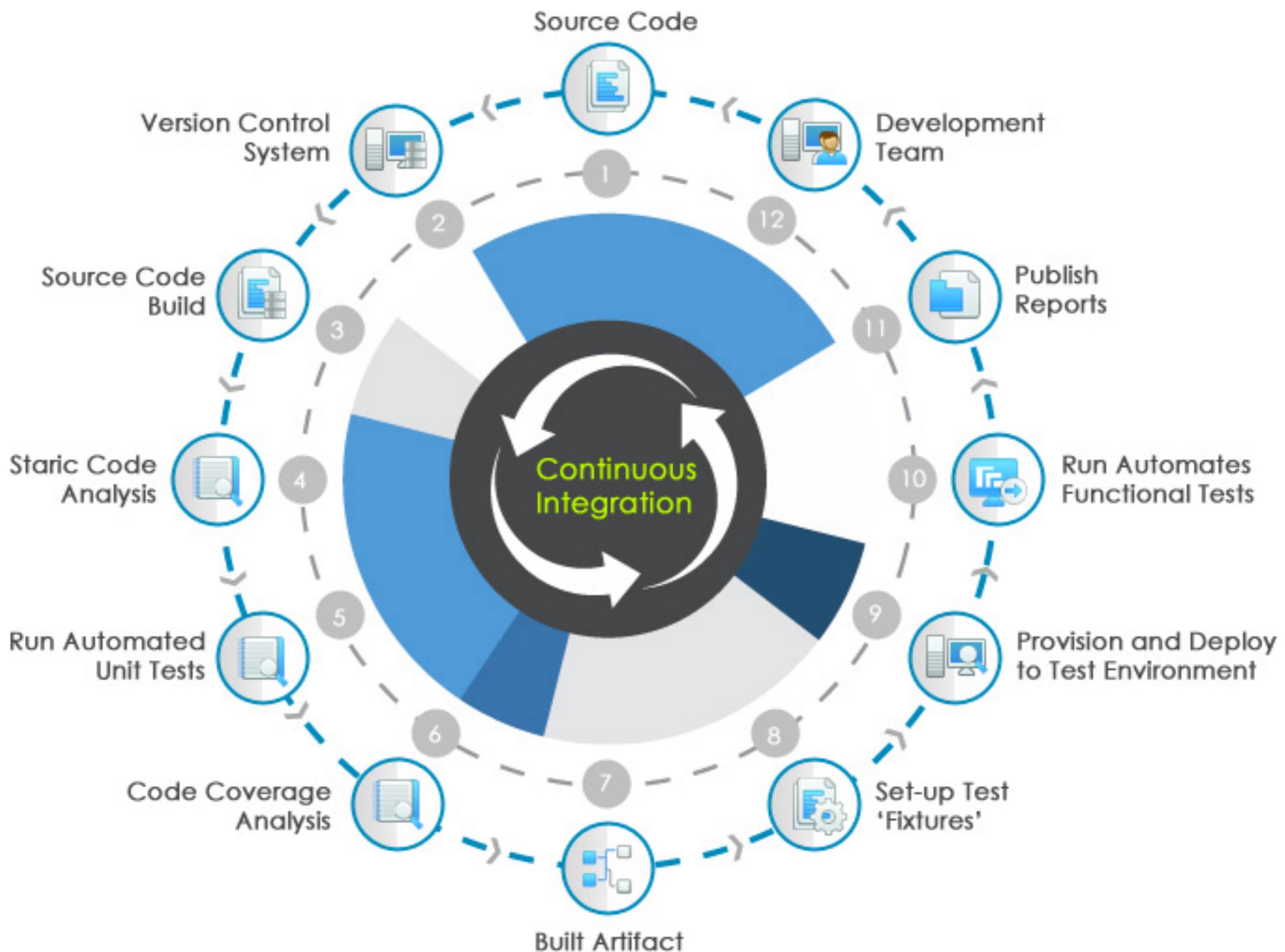
# The cost of integration

1. Merging the code
2. Duplicate changes
3. Test again again !!
4. Fixing bugs
5. Impact on stability



# The cost of integration







Jenkins

Bamboo



TeamCity

> go<sup>TM</sup>



Hudson





Jenkins

Bamboo

CI is about what people do  
not about what tools they use



Hudson



# Continuous Integration

Discipline to integrate frequently



# Continuous Integration

Strive to make **small change**

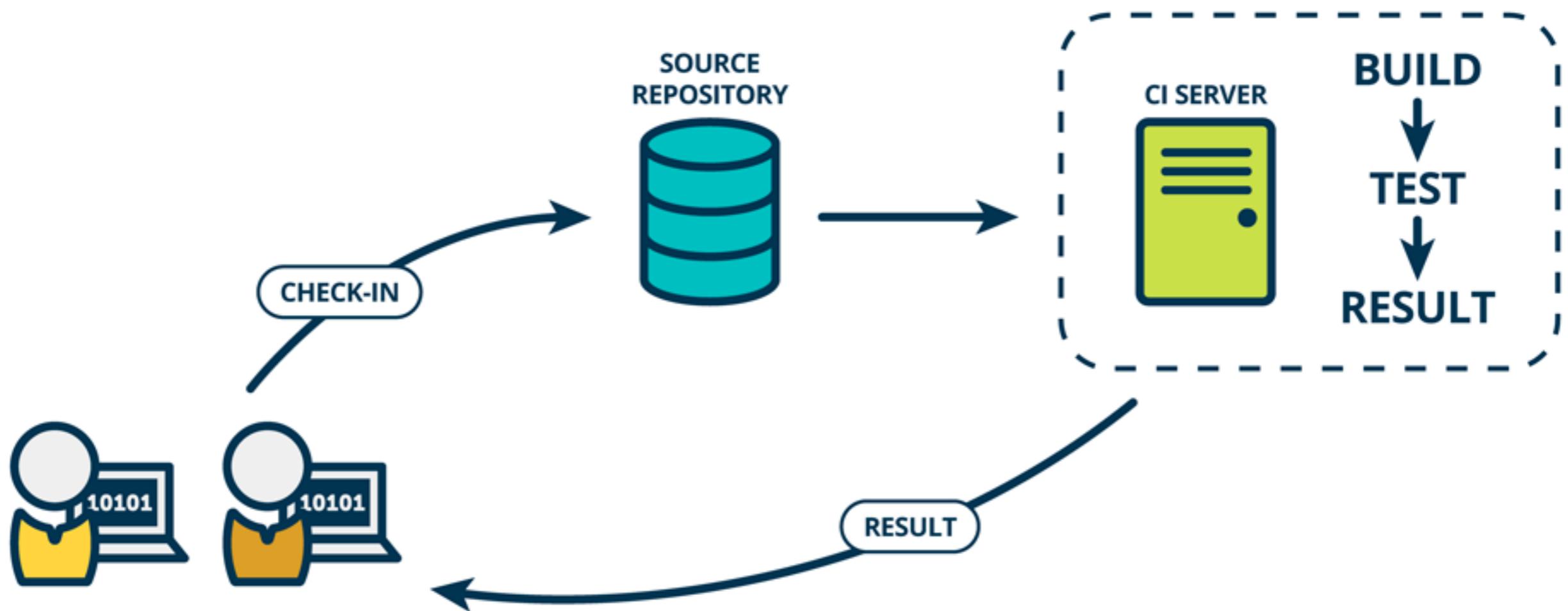


# Continuous Integration

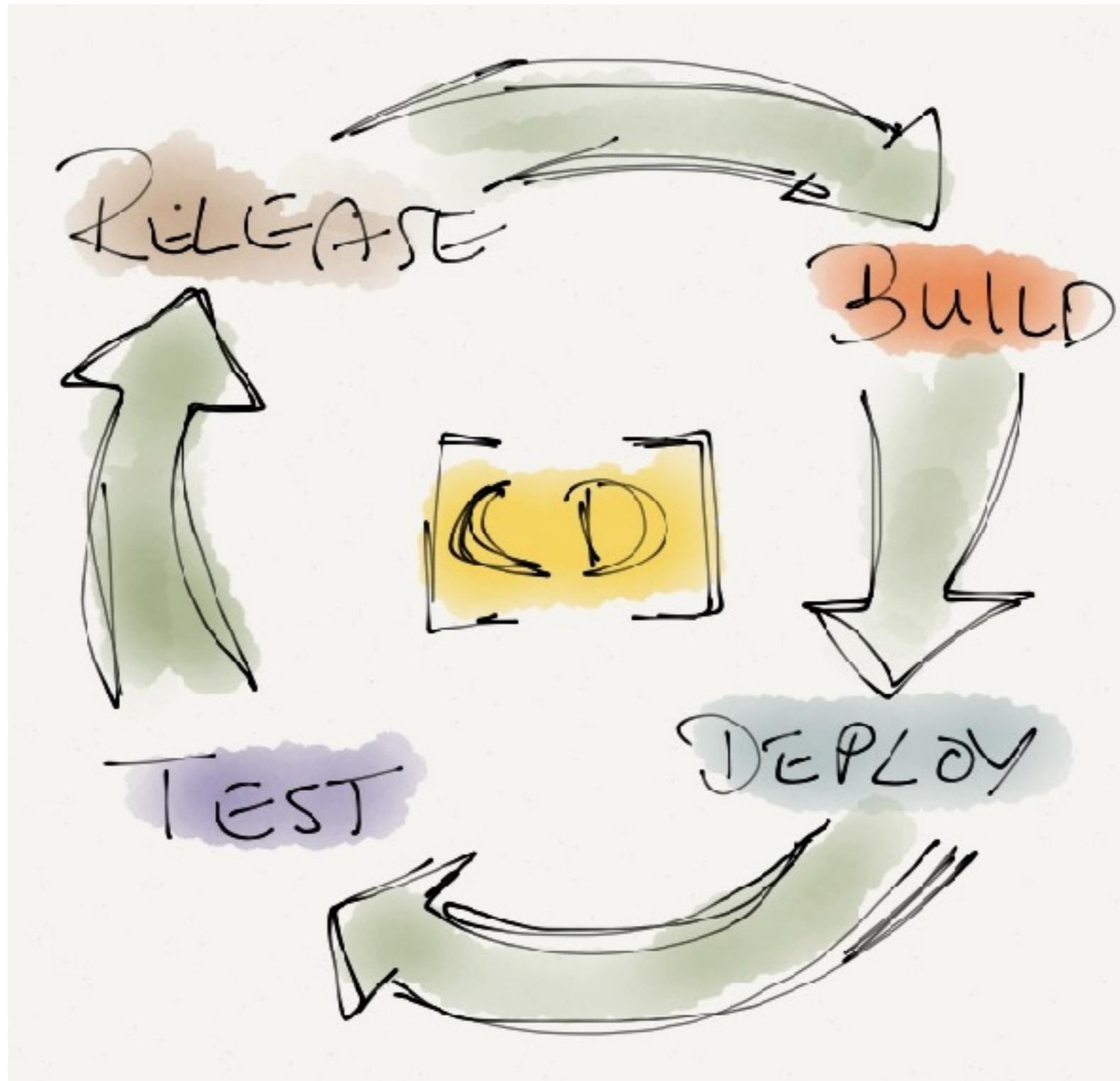
Strive for **fast feedback**



# Continuous Integration



# CD ?



# CD ?

## CONTINUOUS DELIVERY



## CONTINUOUS DEPLOYMENT



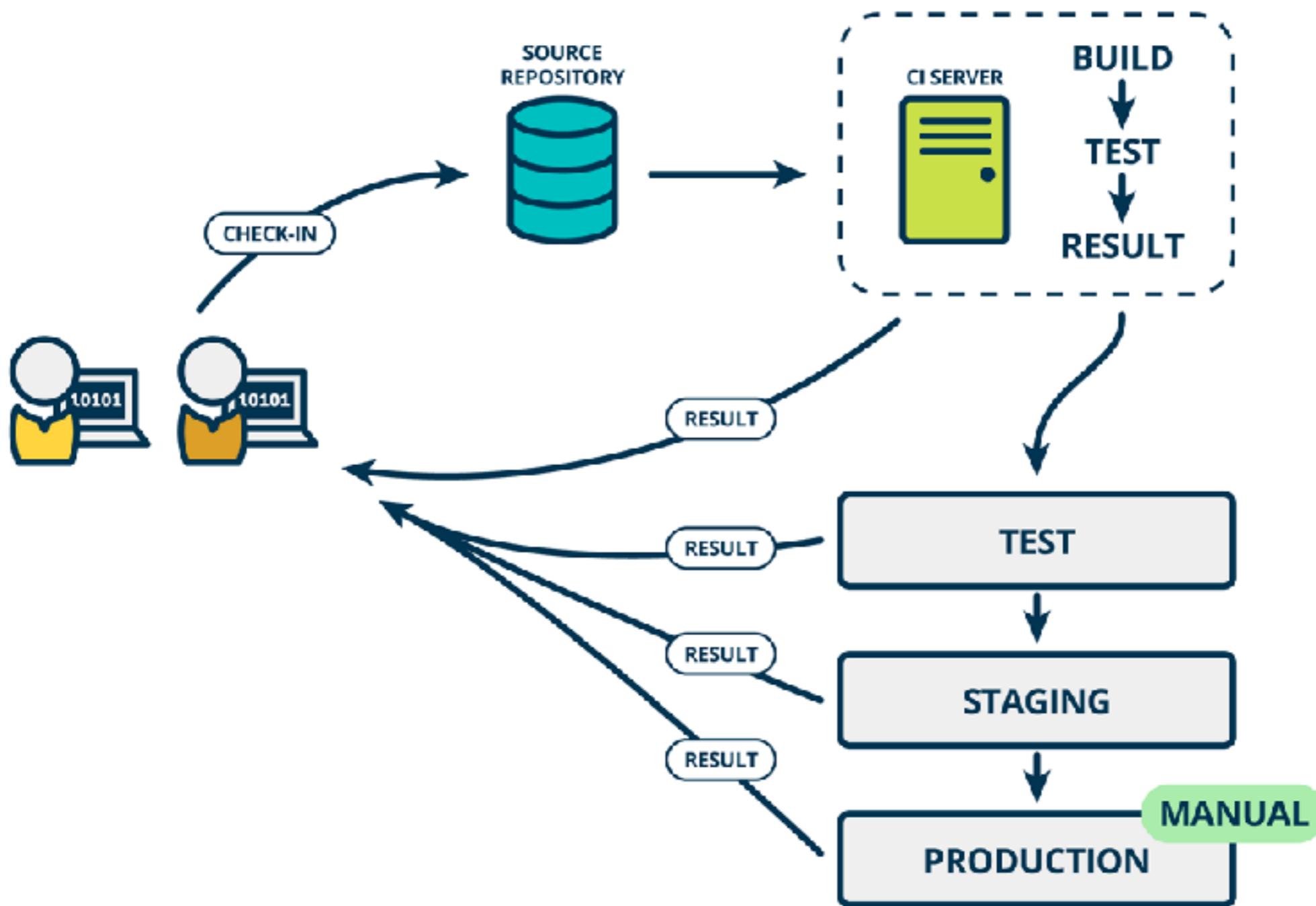
<http://blog.crisp.se/2013/02/05/yassalsundman/continuous-delivery-vs-continuous-deployment>



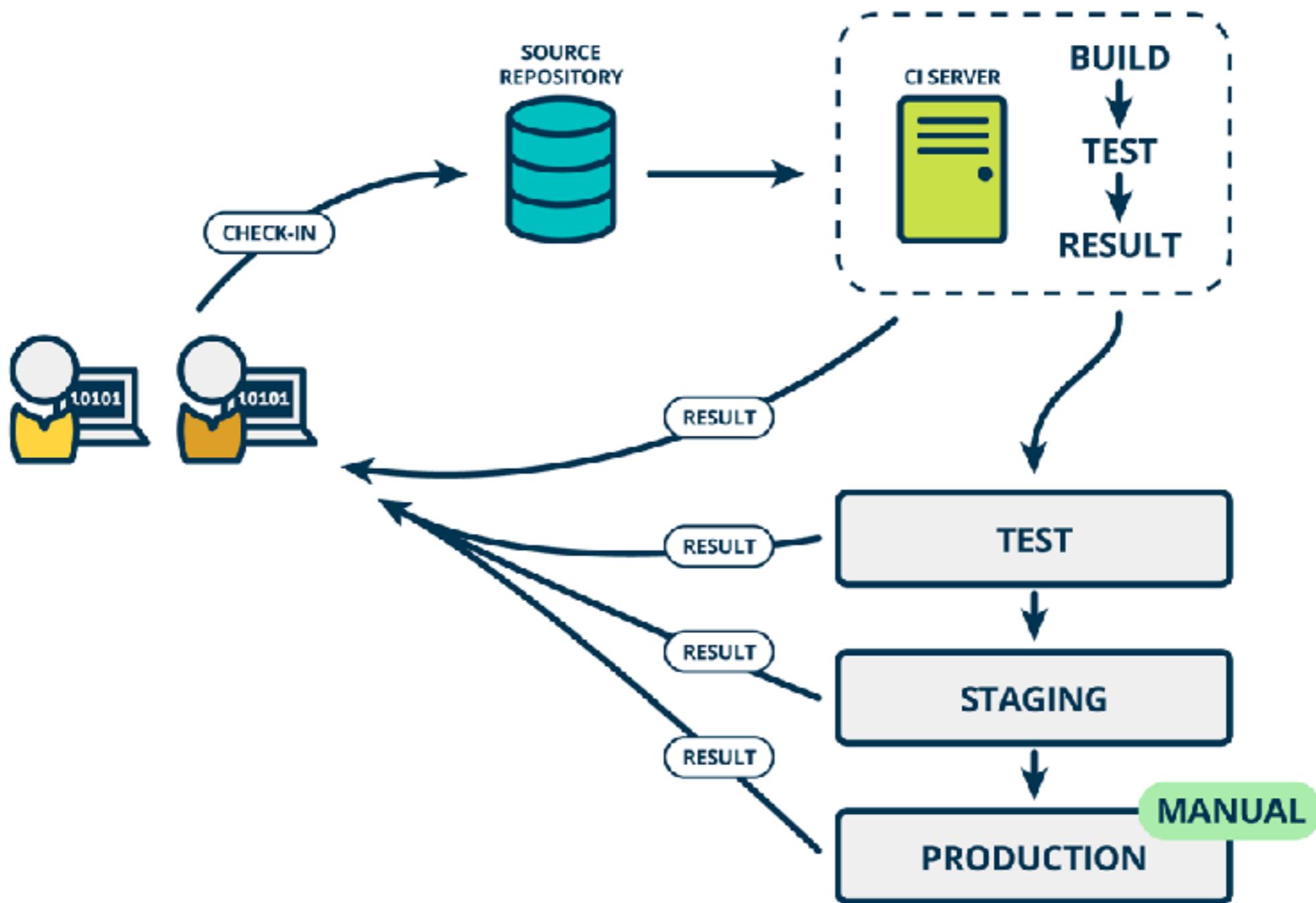
Microservices

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

# Continuous Delivery



# Rise of DevOps



# **Continuous Integration**

**is a Software development practices**



# Practice 1

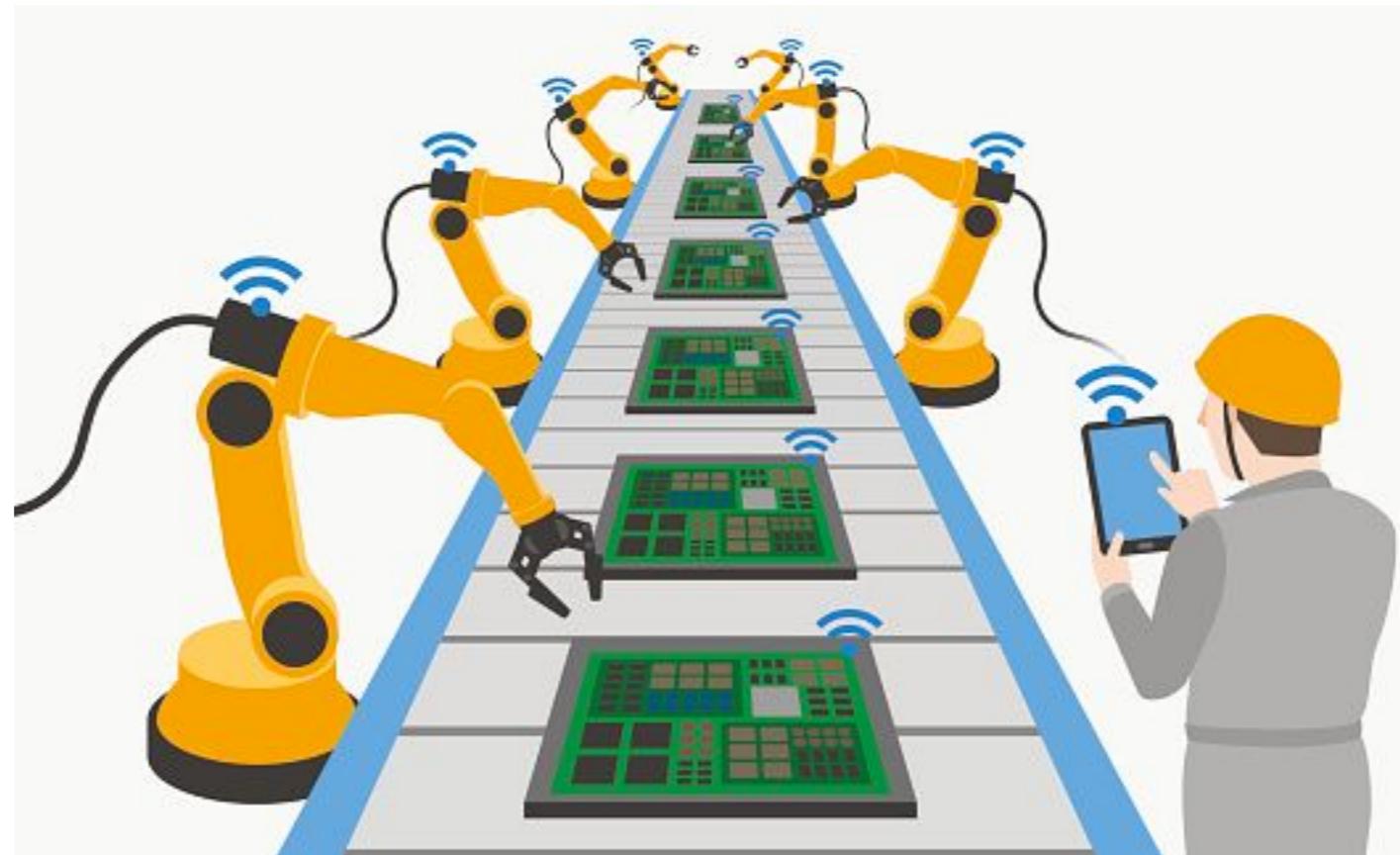
Maintain a single source repository

In general, you should store in source control  
everything you need to build anything



# Practice 2

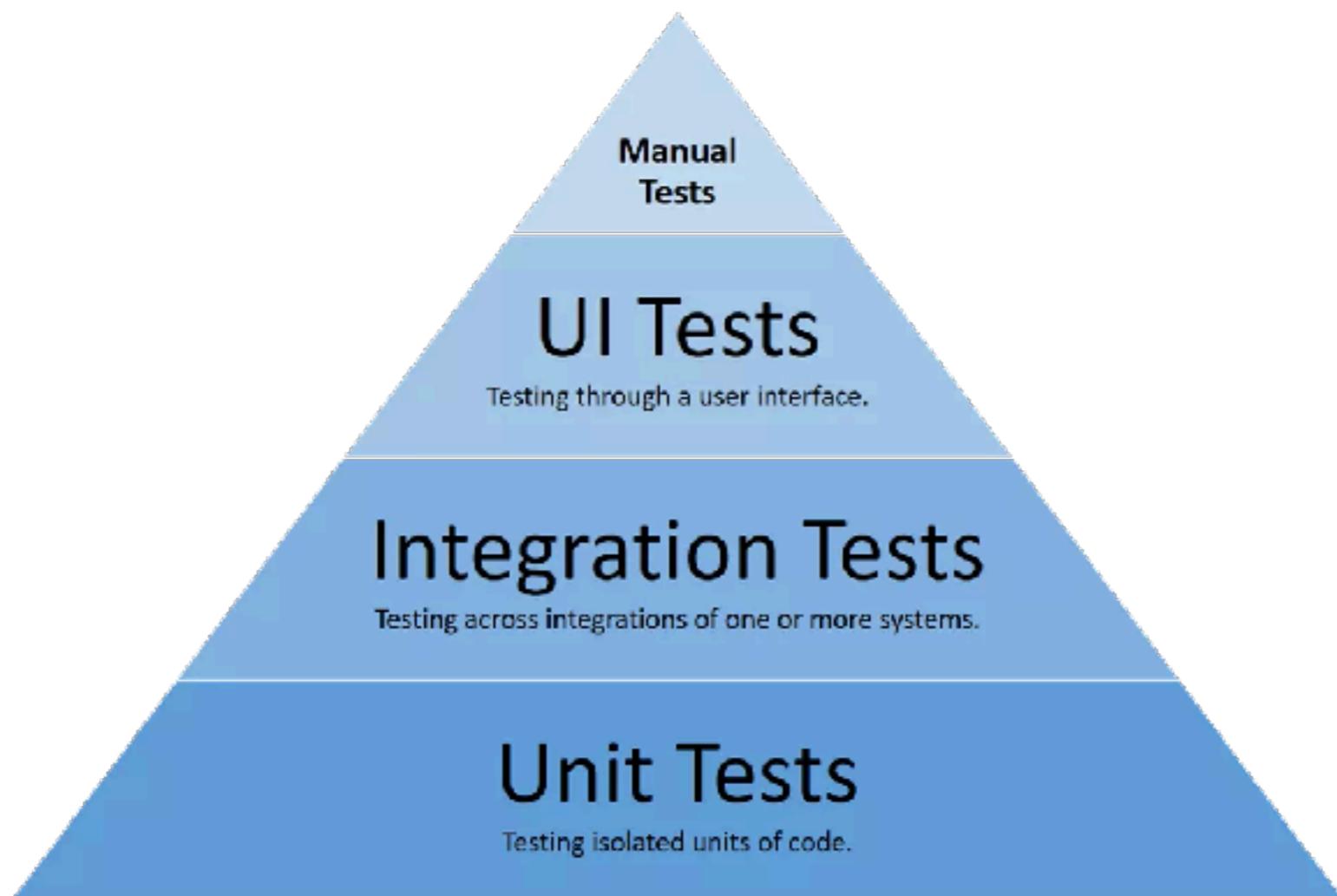
Automated the build  
Automated environment for builds



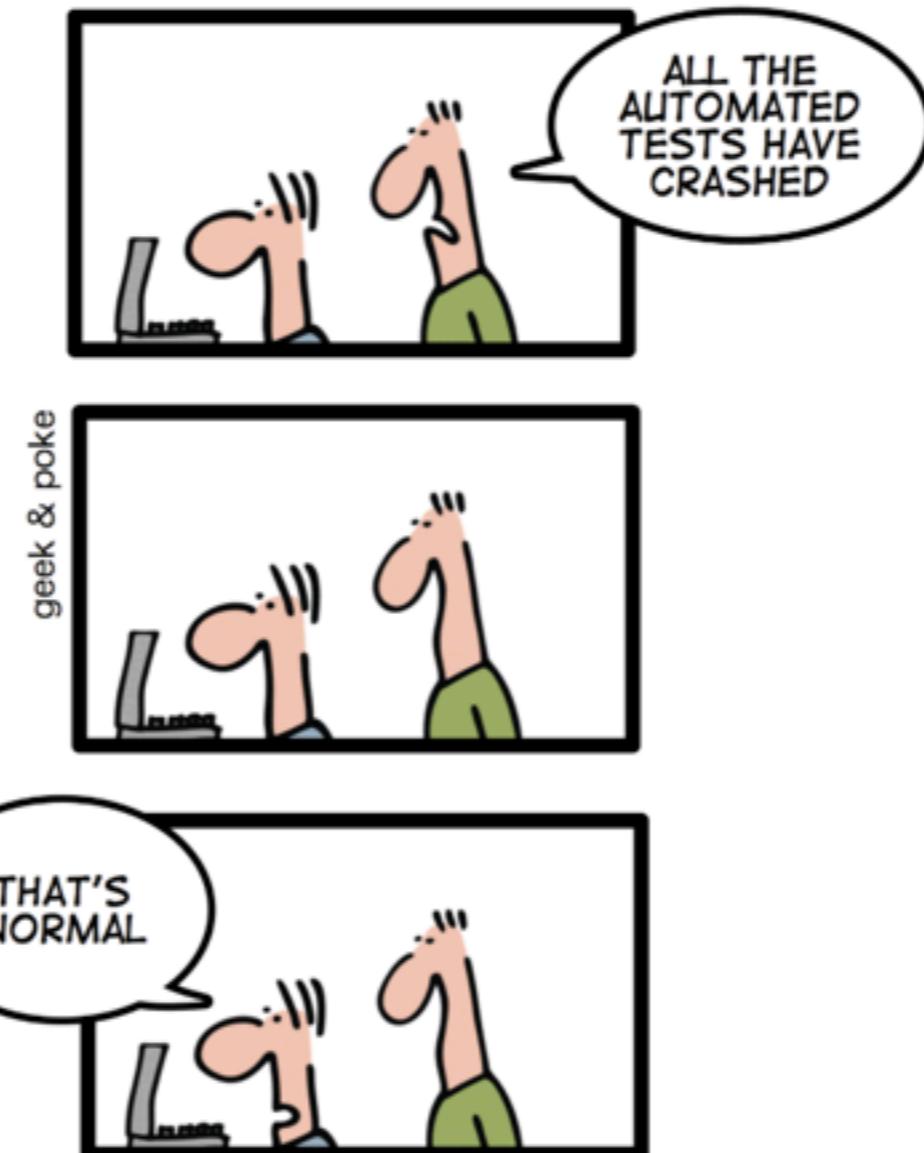
# Practice 3

Make your build **self-testing**

Build process => compile, linking and **testing**



*TODAY: CONTINUOUS INTEGRATION  
GIVES YOU THE COMFORTING  
FEELING TO KNOW THAT  
EVERYTHING IS NORMAL*

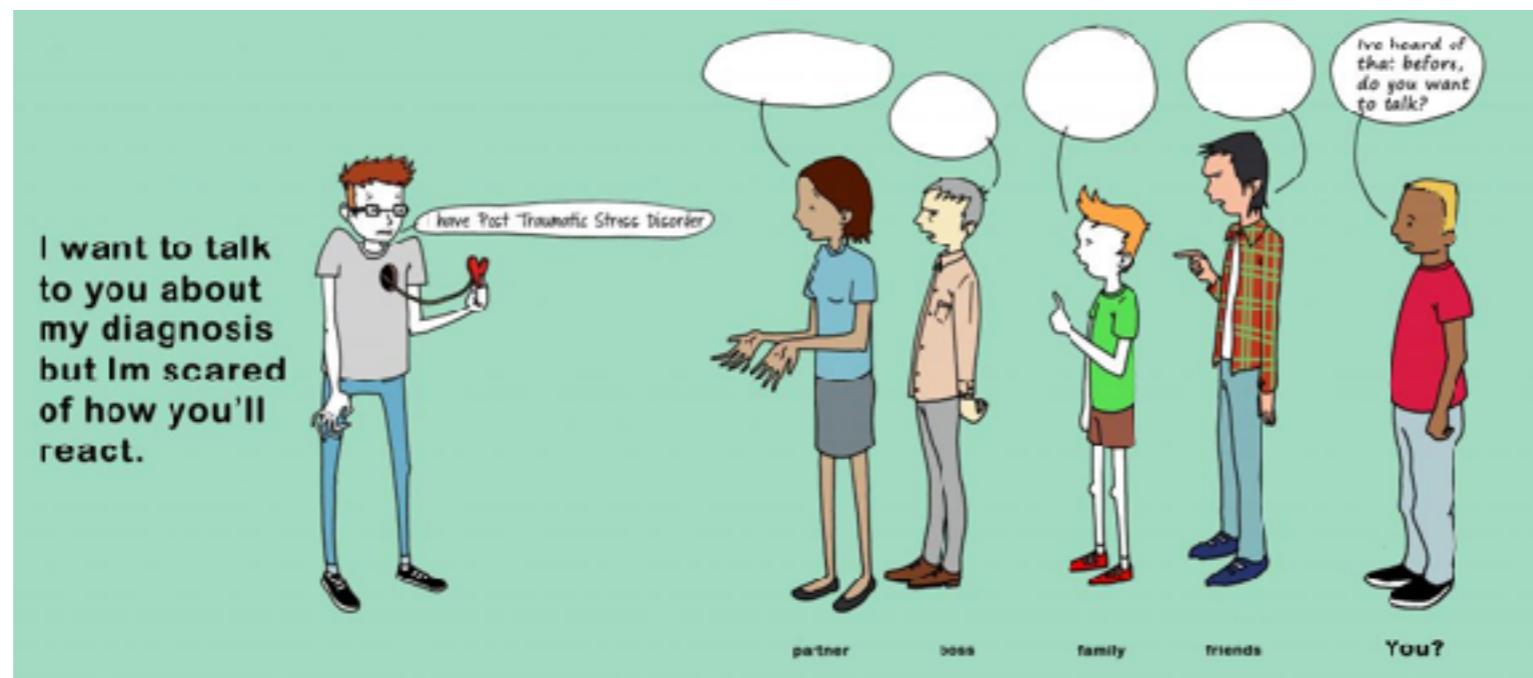


# Practice 4

**Everyone commits to the mainline everyday**

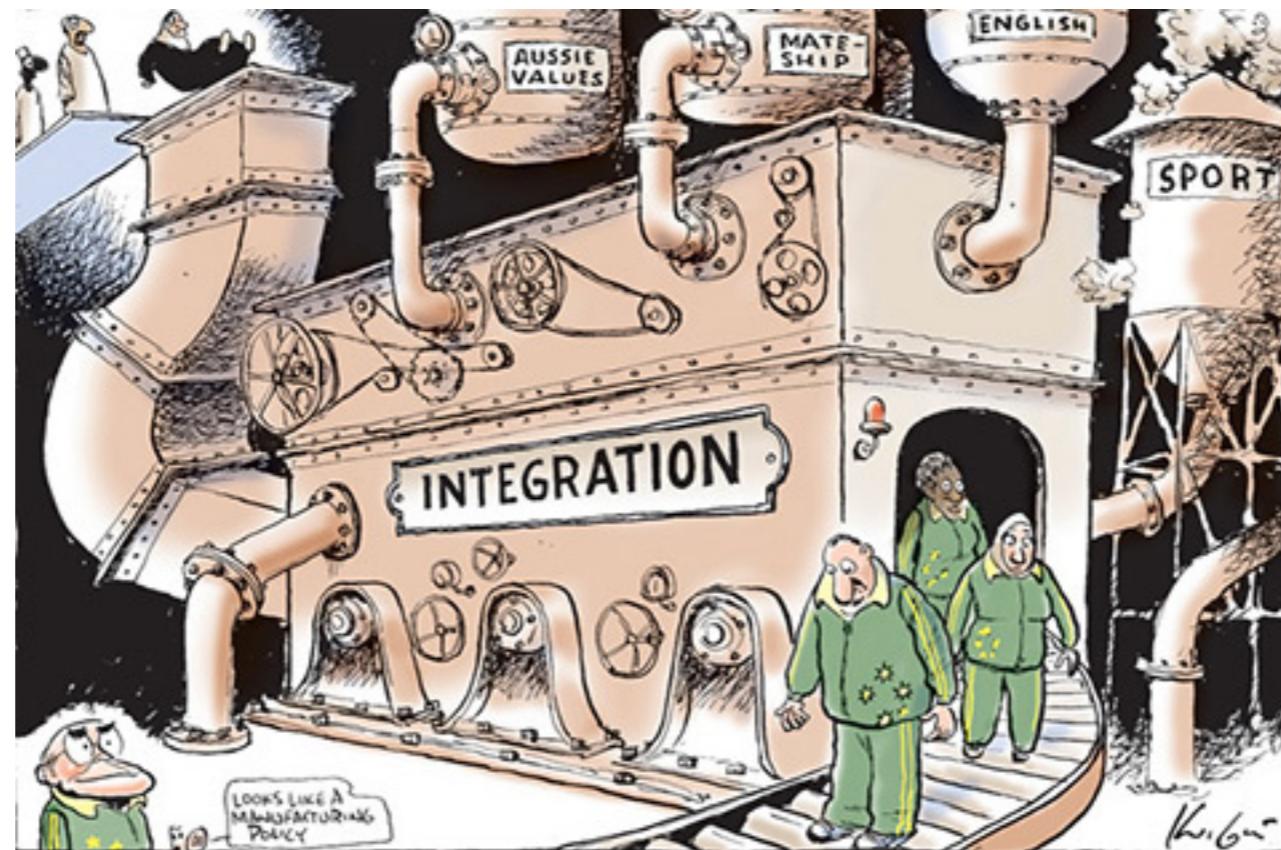
**Integration is about communication**

**Integration allows developers to tell other developers**



# Practice 5

Every commits should build the mainline on an  
**Integration machine**



# Nightly build is not enough for Continuous Integration



# Practice 6

**Fix broken builds immediately**

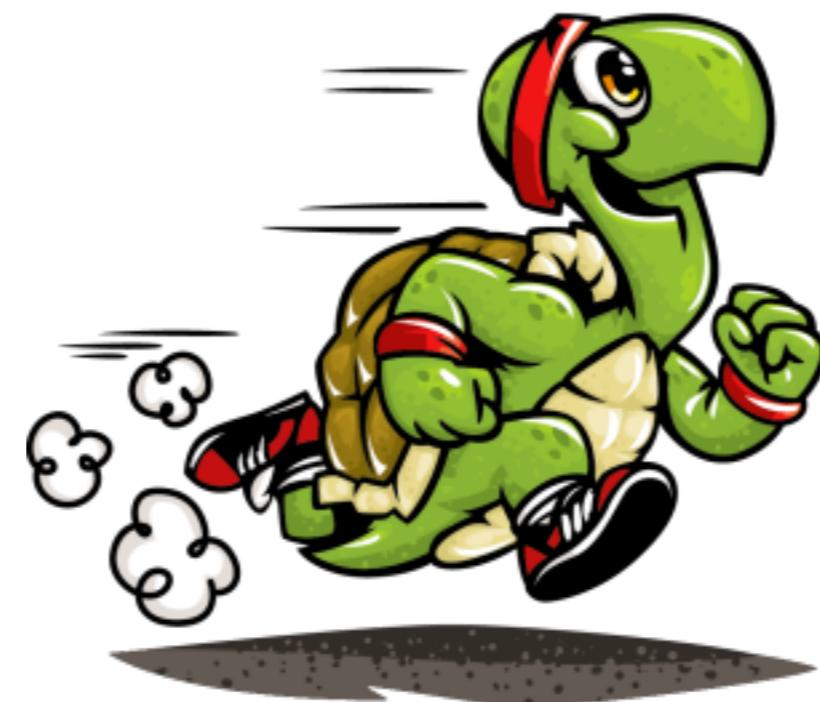
**“Nobody has a higher priority task than fixing the build”**



# Practice 7

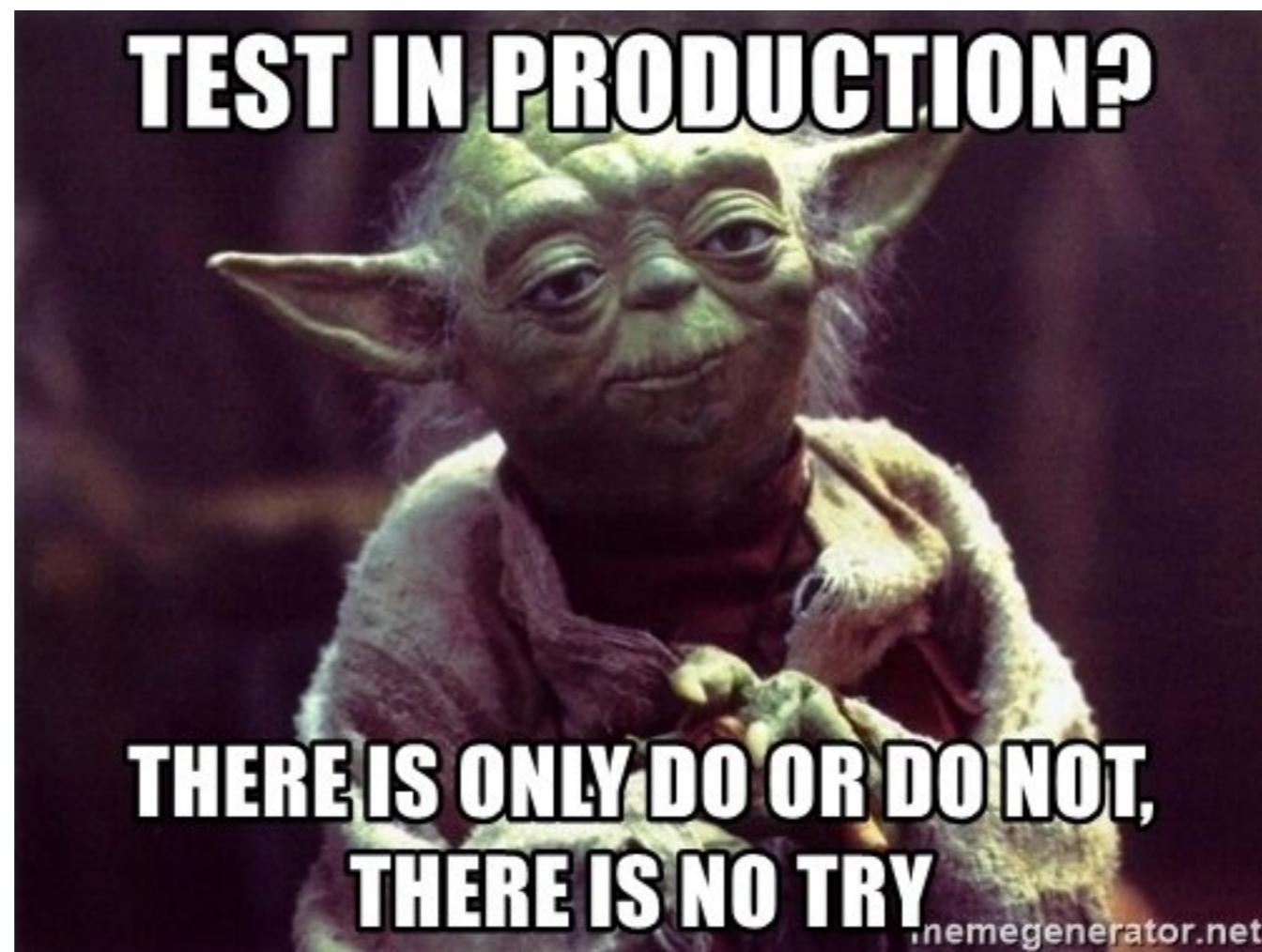
Keep the build **fast**

Continuous Integration is to provide rapid feedback



# Practice 8

Test in clone of the **Production** environment



# Practice 9

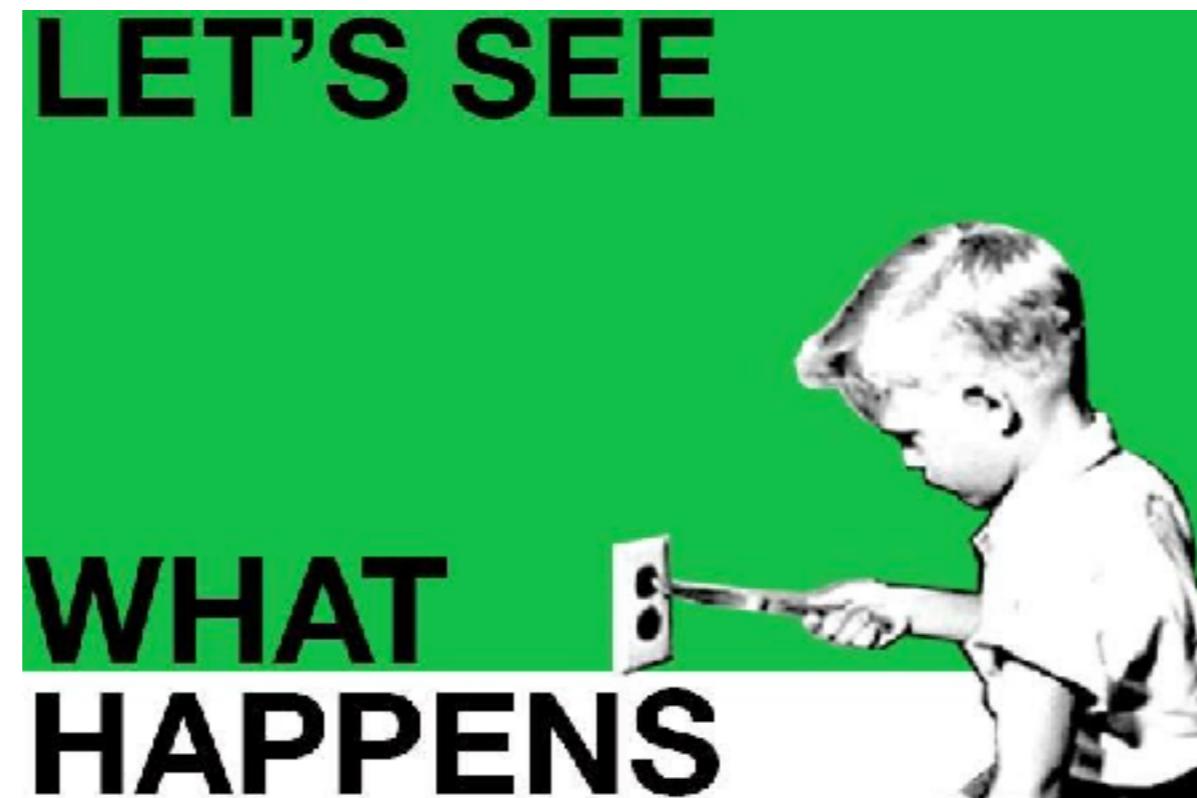
Make it easy for anyone to get  
the latest executable

Make sure well known place where people can find



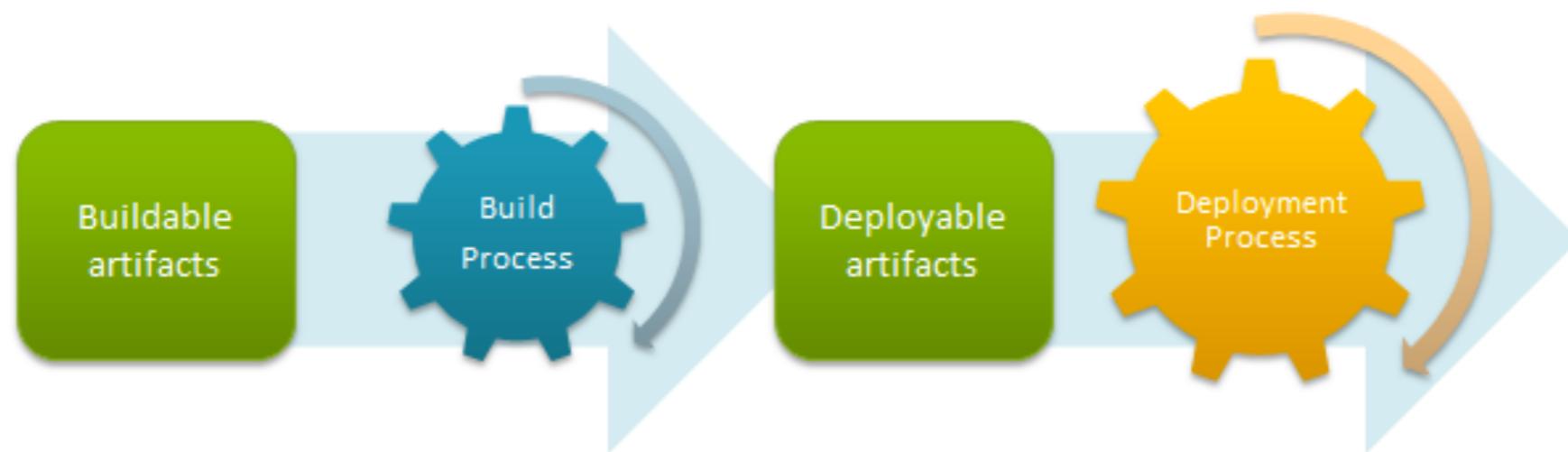
# Practice 10

**Everyone** can see what's happening  
**Easier** to see the state of the system and changes  
Show the good information



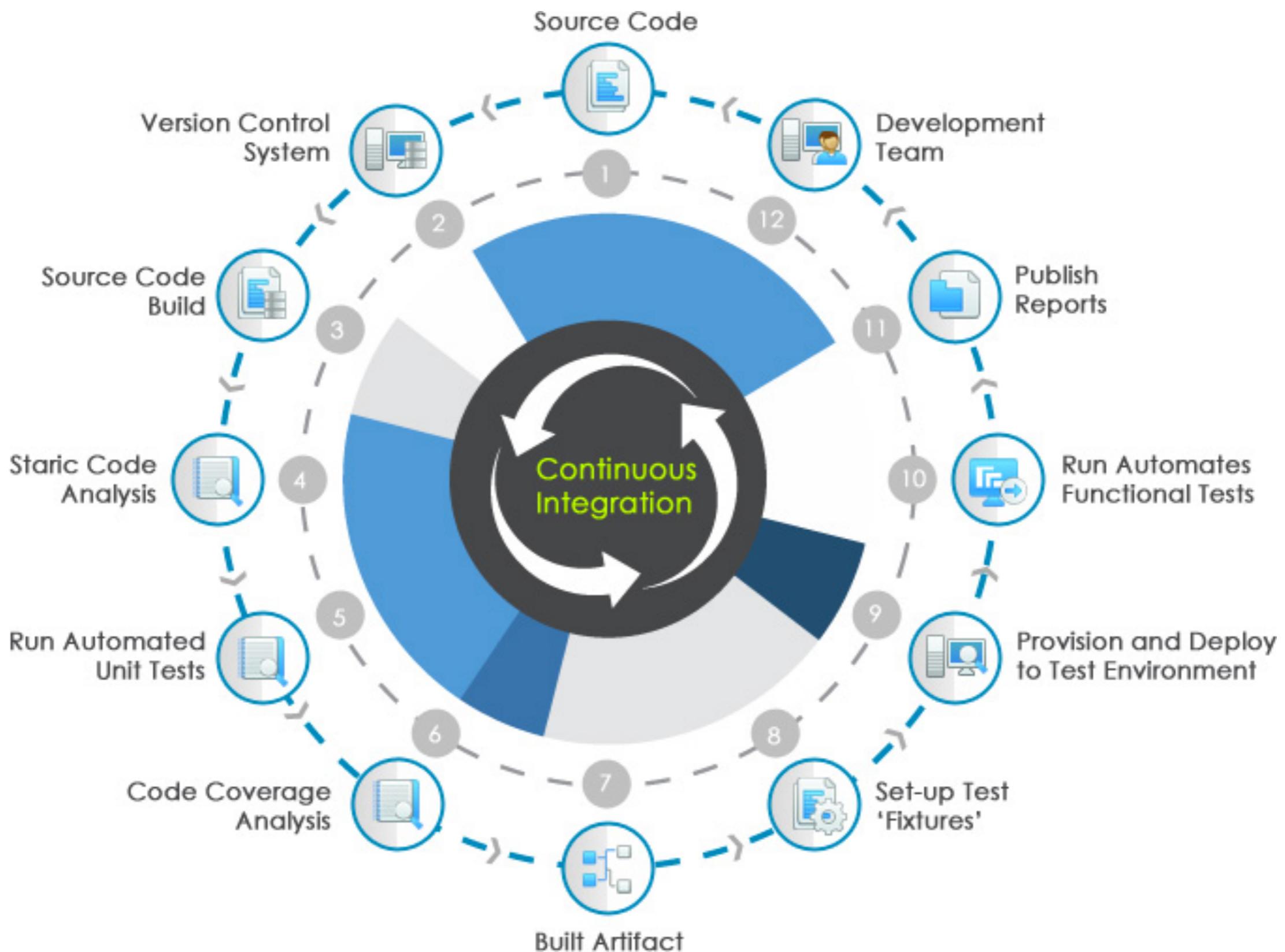
# Practice 11

## Automated deployment



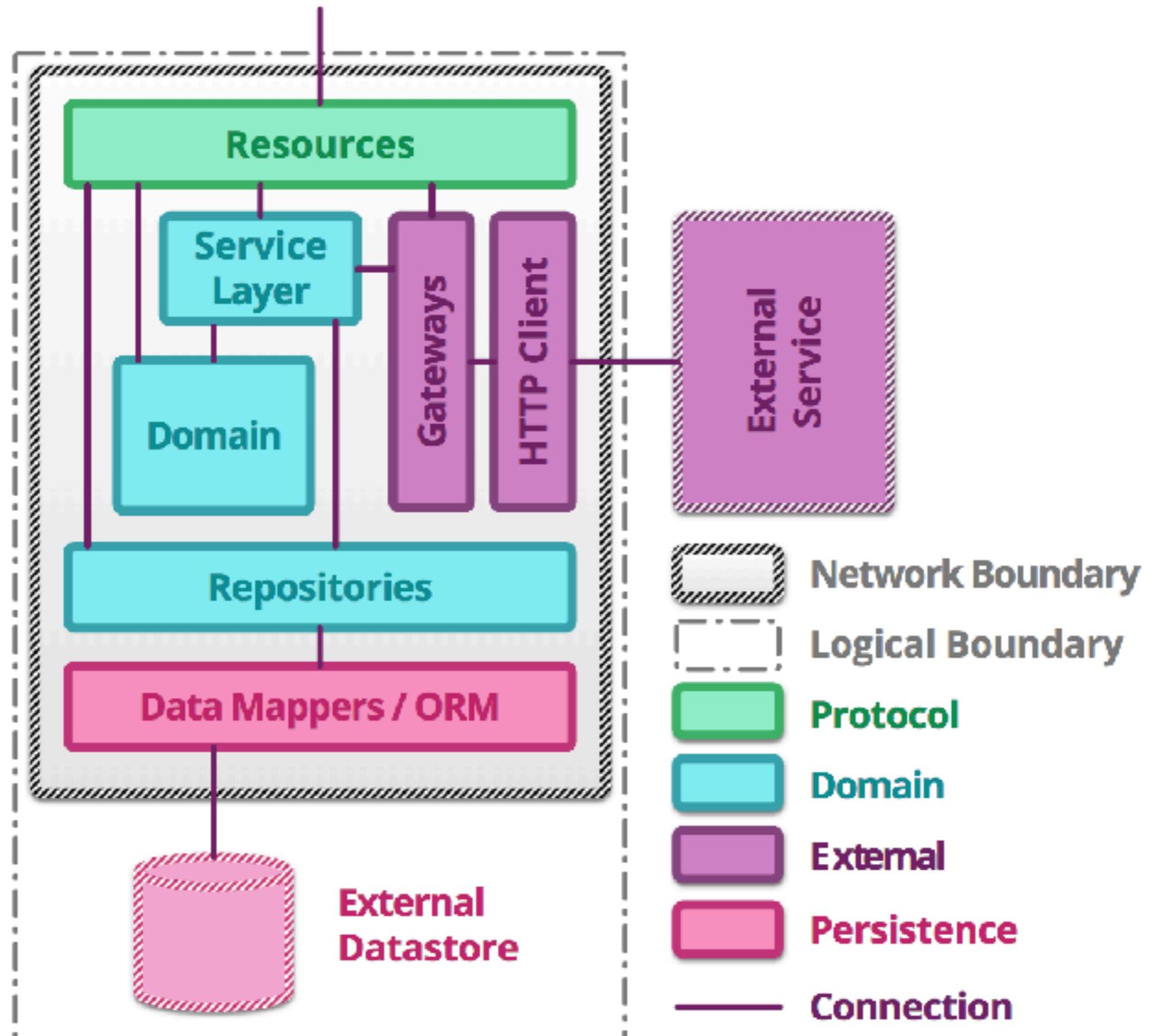
**“Behind every successful agile  
project, there is a  
Continuous Integration Server”**





# Let's workshop





# Development



# Testing



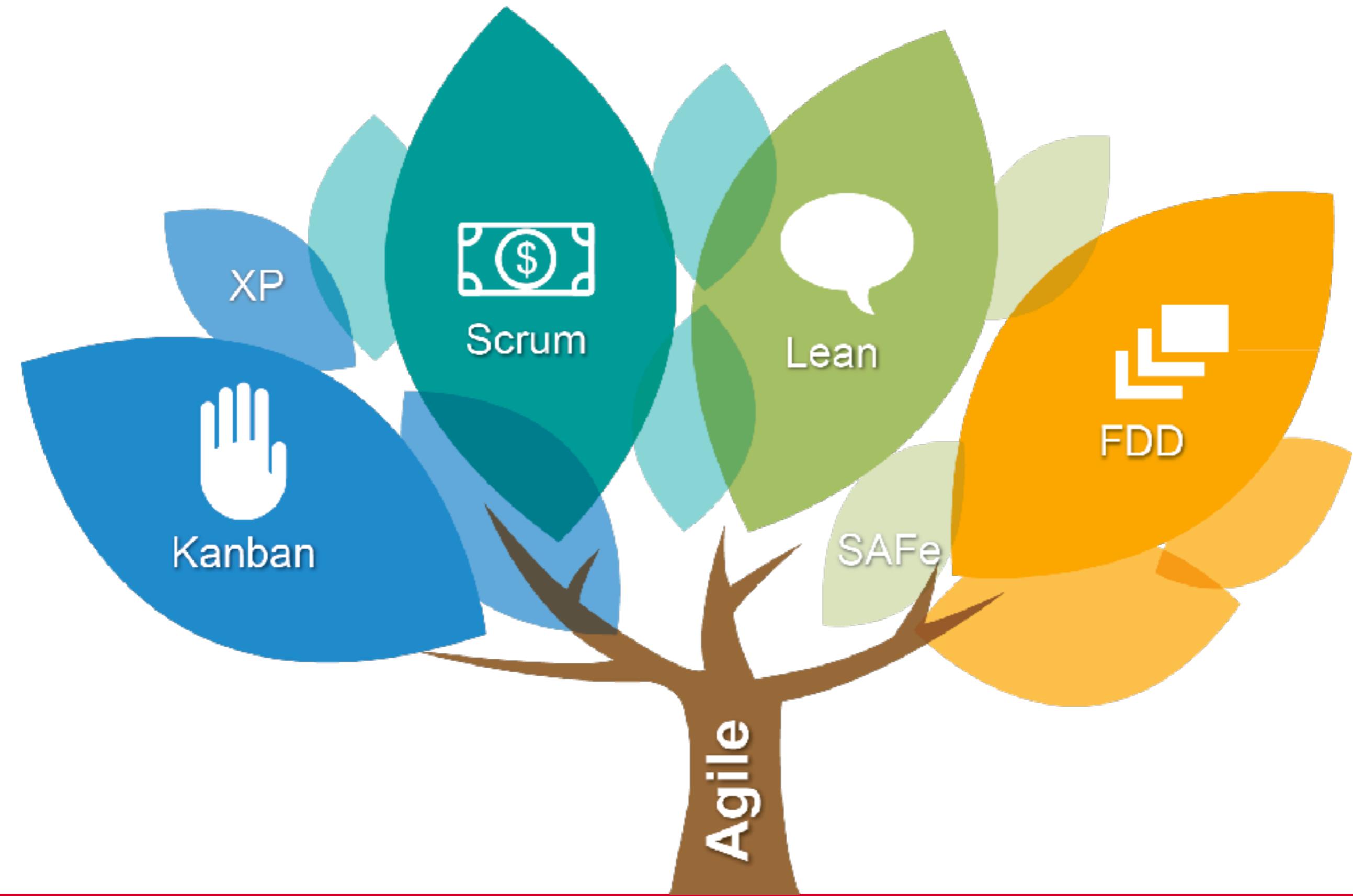
# Deployment

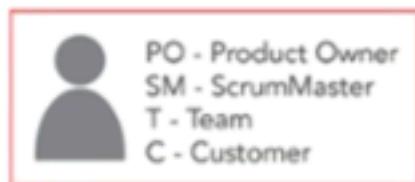


# Summary

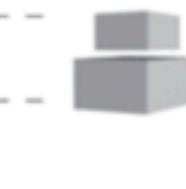
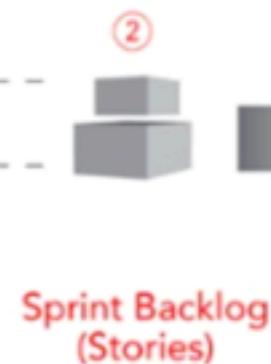








Product Backlog (Features)



Sprint Planning



Feedback Loop to PO



Feedback Loop to PO

Sprint 1-4 Weeks

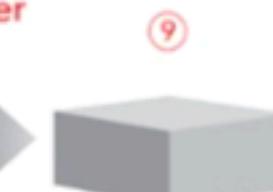
© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.



24 hrs



Product Owner



Microservices

215

# Agile manifestos

## THE AGILE MANIFESTO

We are uncovering better ways of developing software by doing it and helping others do it.

**CUSTOMER  
COLLABORATION**  
over contract negotiation

**RESPONDING TO  
CHANGE**  
over following a plan

**INDIVIDUALS AND  
INTERACTIONS**  
over processes and tools

**WORKING  
SOFTWARE**  
over full documentation



# Agile principles

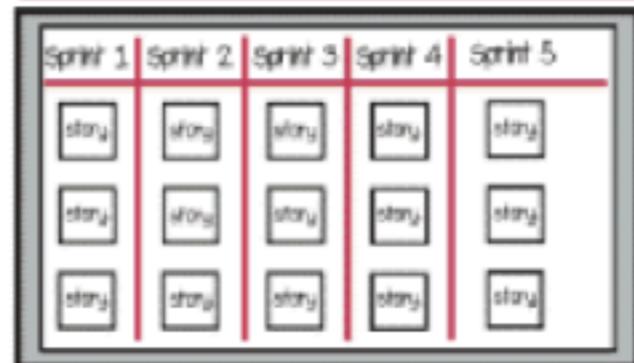
1 Satisfy the **customer**



Welcome **change**



Deliver **frequently**



4 Work **together**



5 Trust and **support**



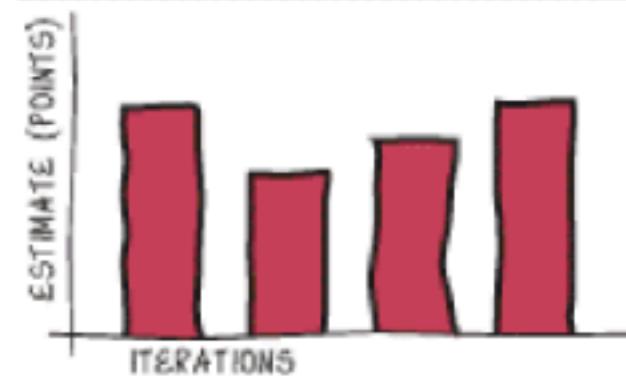
Face-to-face **conversation**



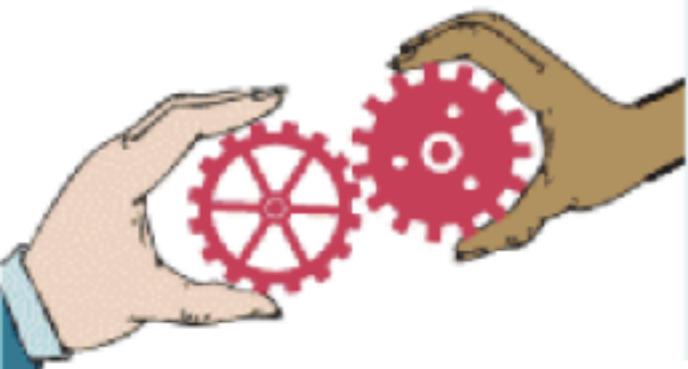
Working **software**



8 Sustainable **development**



9 Technical **Excellence**



10 Maintain **simplicity**



11 Self-organizing **teams**

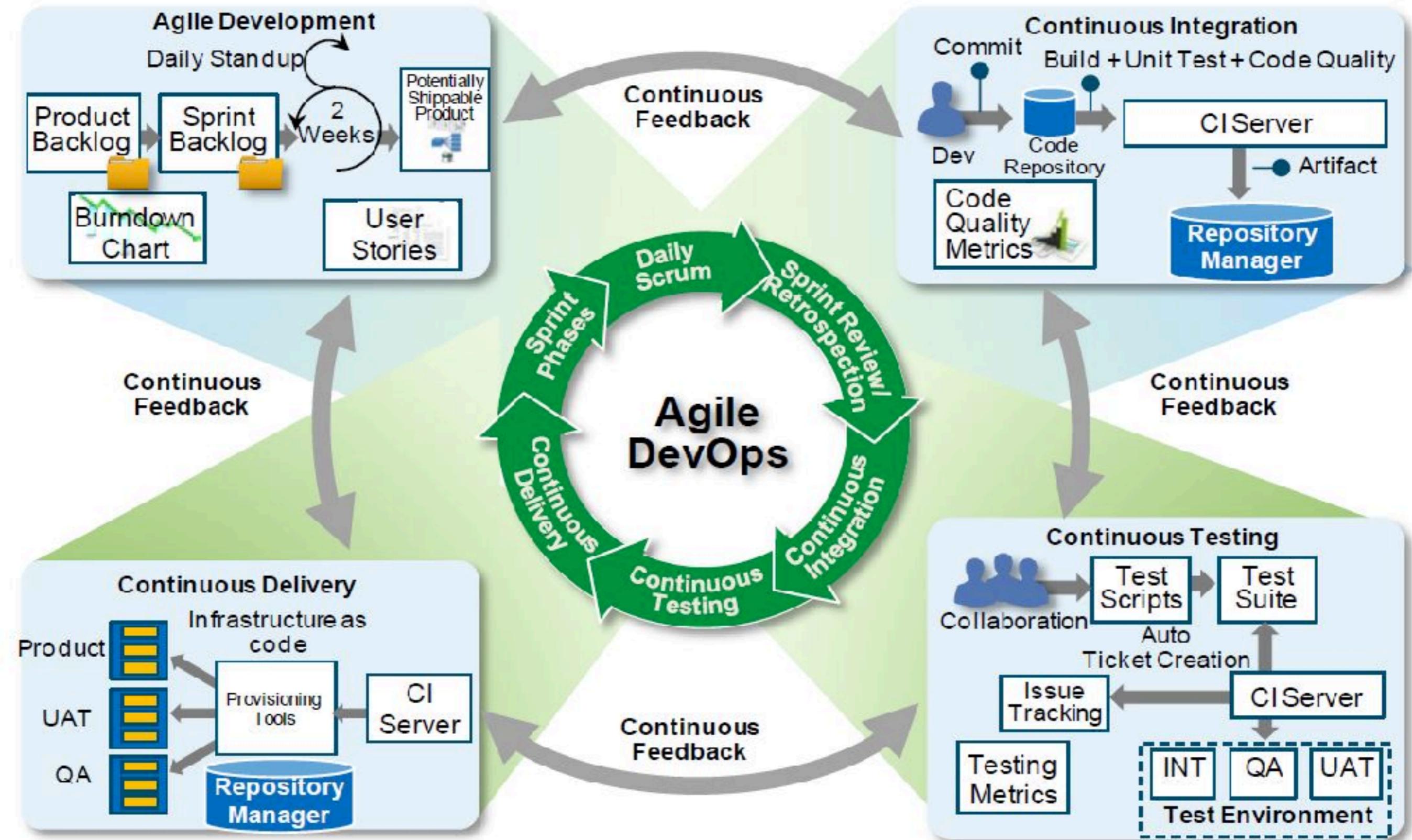


12 Reflect and **adjust**

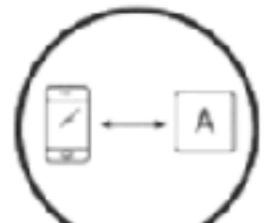


Origin by <https://www.knowledgetrain.co.uk>, modified by Jacky Shen

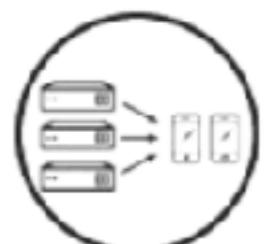




# Continuous Improvement



Monolith



N-Tier



Microservices

**Applications**



Datacenter



Hosted

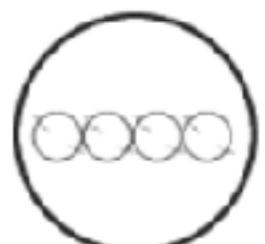


Hybrid

**Infrastructure**



Waterfall



Agile



DevOps

**Process**



# Improve Time to Value

