Advanced Topics in Computer Science (CS3001)

Topic 2: Microservices

Dr Rumyana Neykova

Learning objectives

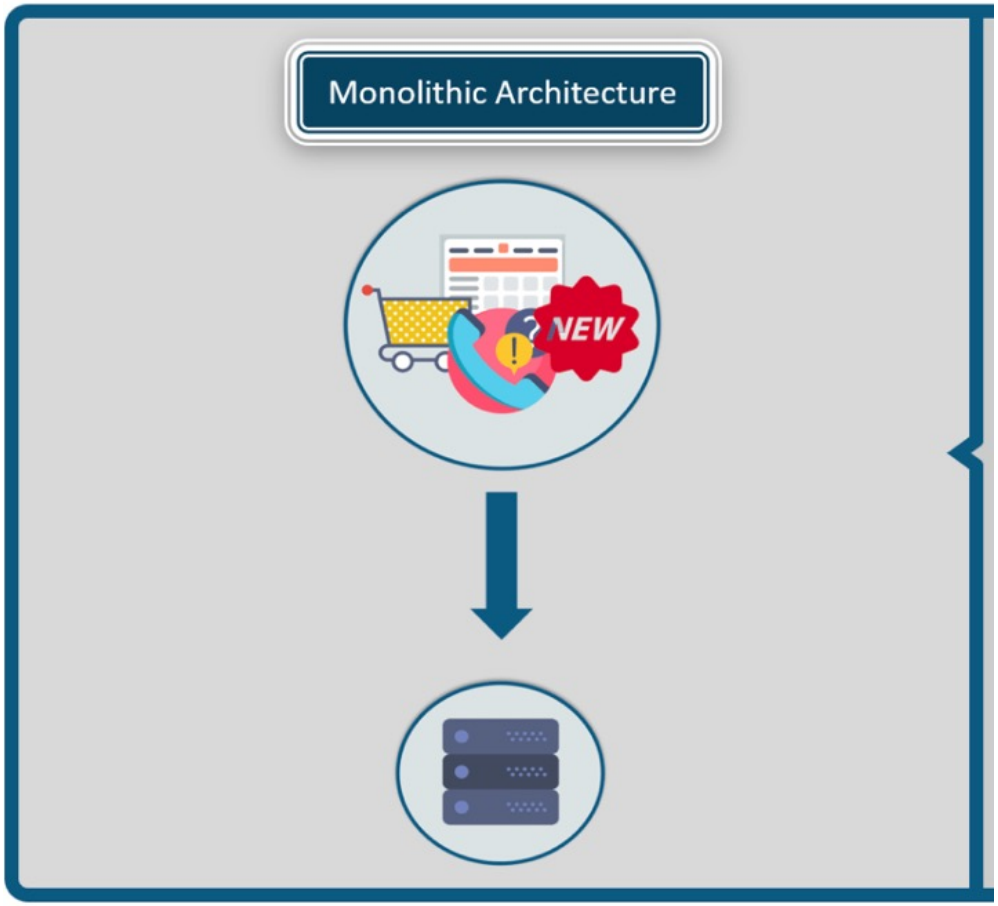
- Why should we care about microservices?
  - Company success stories
- What is a microservice architecture
  - What is a software architecture
  - Which are the characteristics of a microservice architecture
- What are the main advantages/disadvantages
- How to build a microservice architecture

Microservices: Why, What, How

Who uses microservices?

How did we use to write software

RECAP

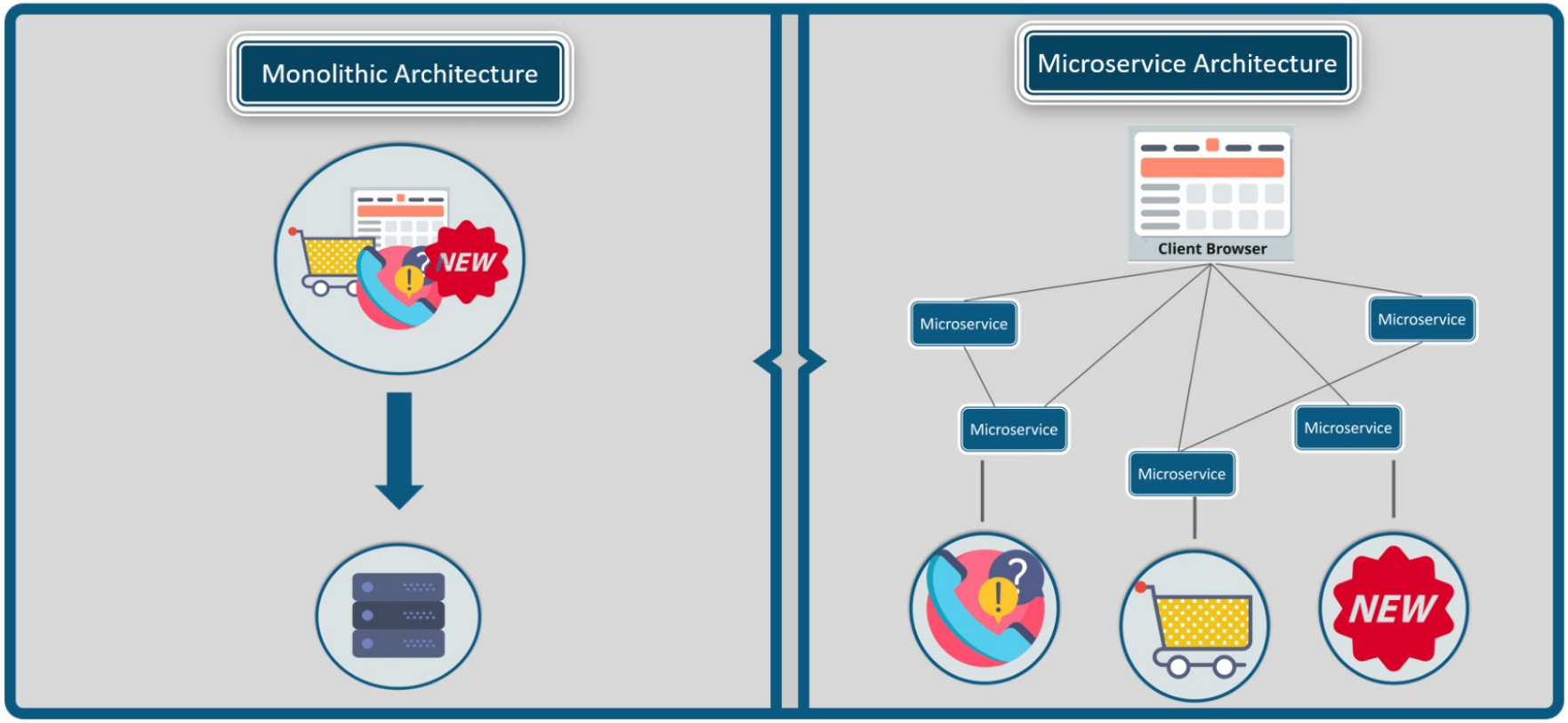


- Many components but….
- One database
- A single deployment unit
- A single language/technology

Do you see any problems with that?

How did we use to write software: Monolith vs Microservices

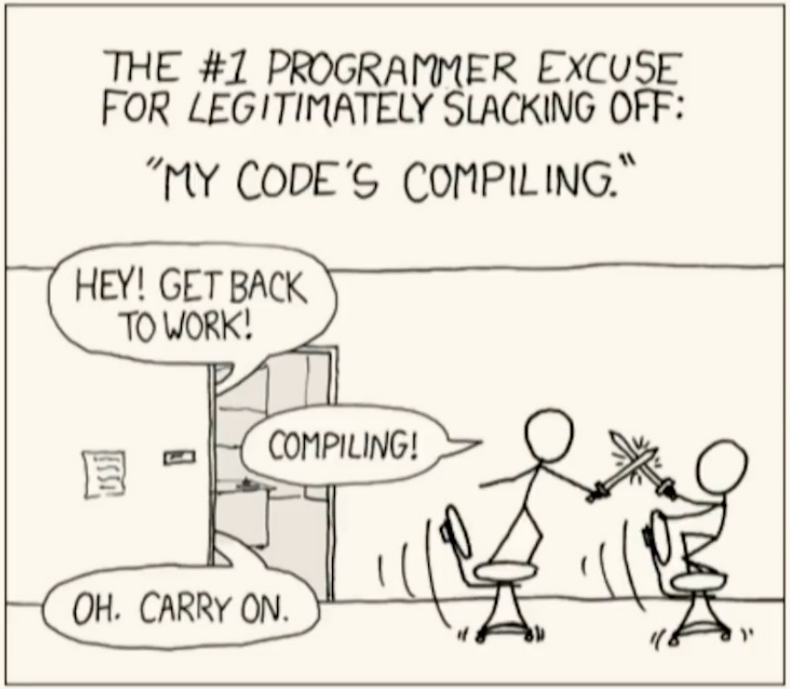
RECAP



Source:https://dzone.com/articles/microservice-architecture-learn-build-and-deploy-a

The Monolith Architecture

RECAP



15h
time per week spent in deploy lock

1.6h
95th percentile time from merge to deploy

200
commits/day deployed to monolith repo

Airbnb's Journey To A Microservice Architecture: https://www.youtube.com/watch?v=Z3_V5ooxfxs&t=60s

Monolith

How to improve?

Microservice architecture to the rescue

What are microservices?

The term "Microservice Architecture" has sprung up over the last few years to describe a particular way of designing software applications as suites of independently deployable services. While there is no precise definition of this architectural style, there are certain common characteristics around organization around business capability, automated deployment, intelligence in the endpoints, and decentralized control of languages and data.

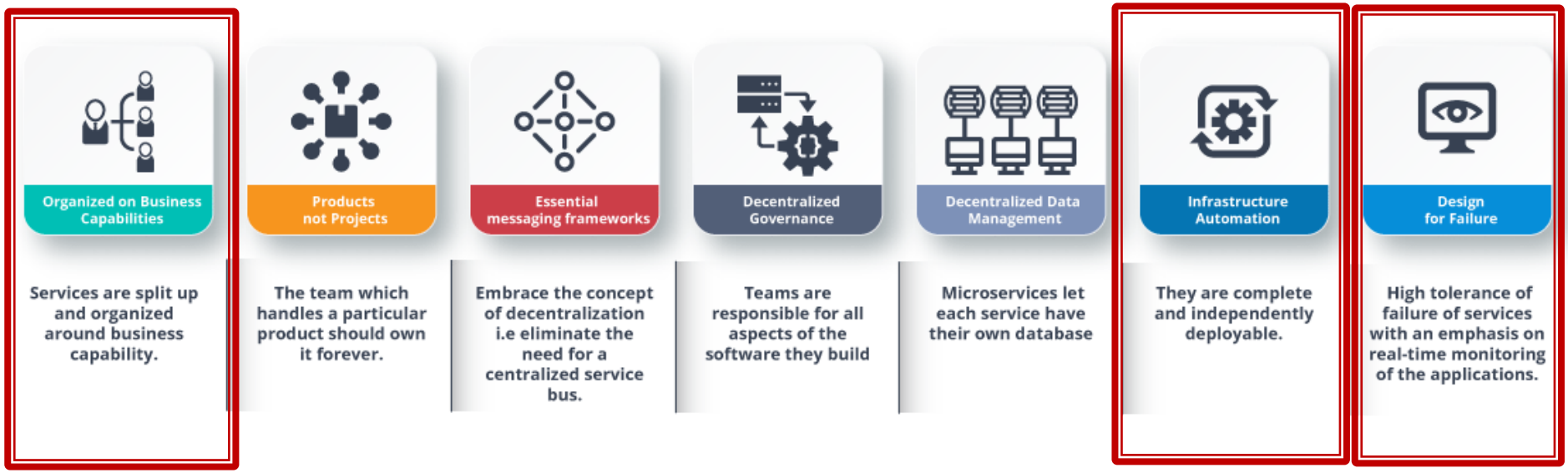
The following is an except from an article that originally appeared on Martin Fowler's website.

What are microservices - the short version

Microservices = Independent Deployment Units

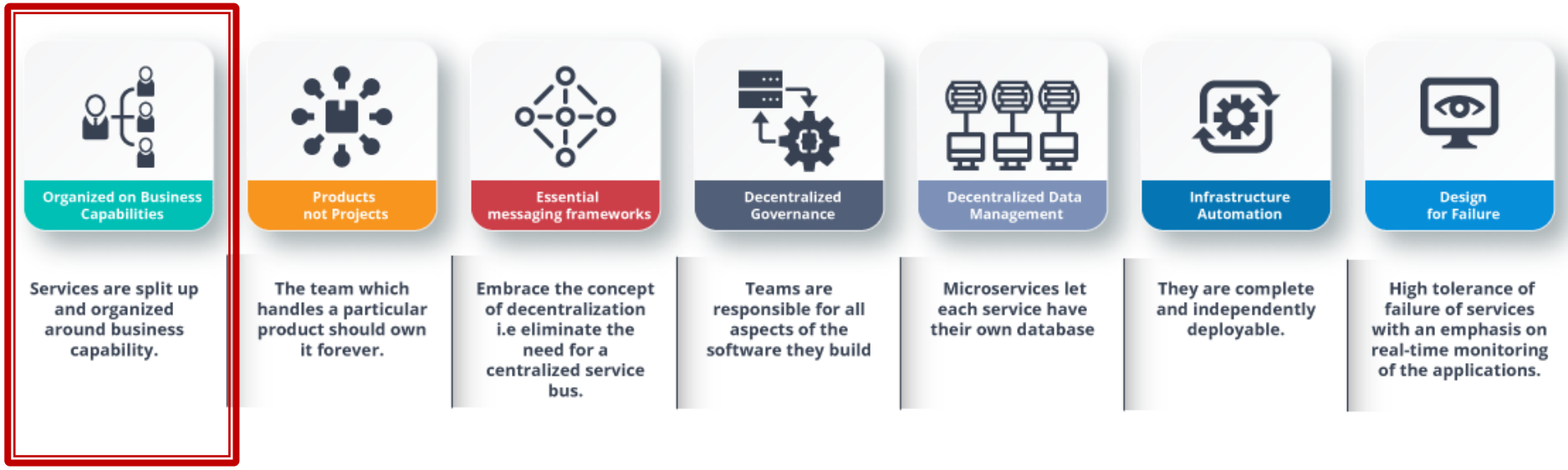
What are the features of Microservices?

- Organised on Business Capabilities
- Infrastructure Automation (build, test, deploy)
- Design for failure (resilience to failure)



What are the features of Microservices?

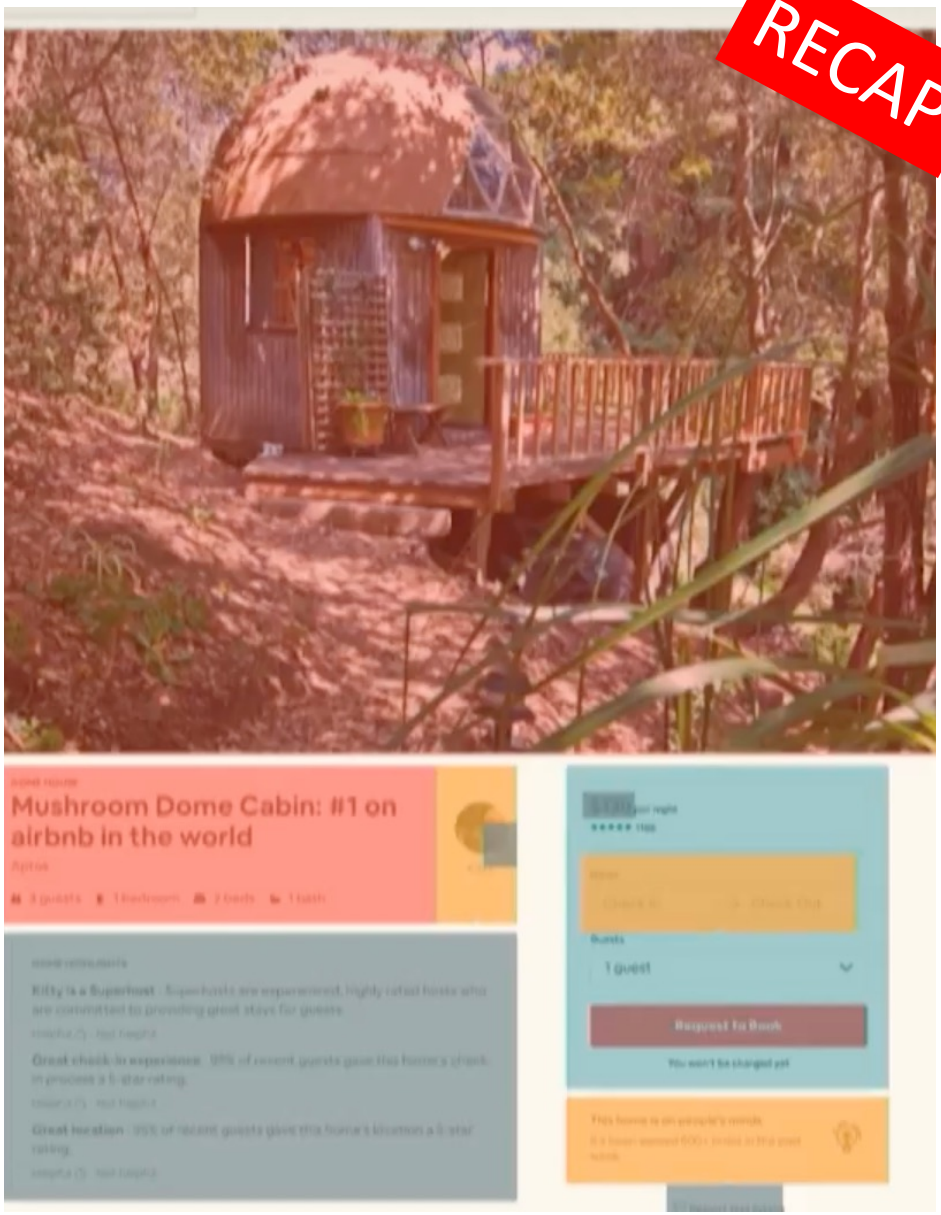
- **Organised on Business Capabilities**
- Infrastructure Automation (build, test, deploy)
- Design for failure (resilience to failure)



From Monolith to Microservices

Split into three services (products):

- image rendering
- price calculation
- host information



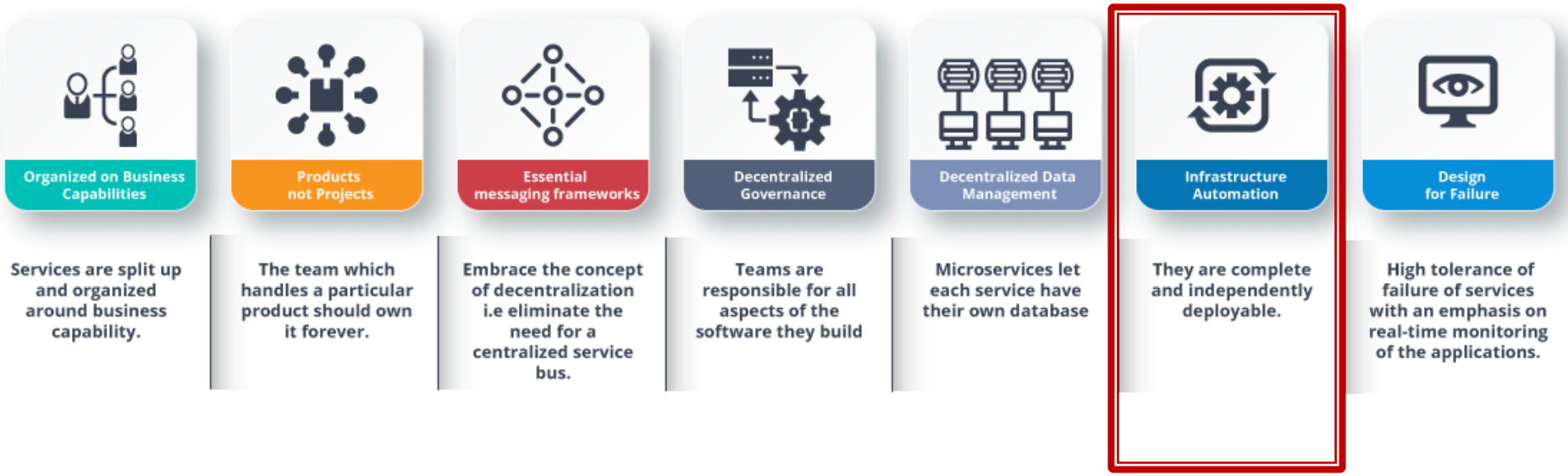
Source: https://www.youtube.com/watch?v=_dFJOSR-aFs

Organised Around Business Domains

The 2 Pizza rule = better productivity

What are the features of Microservices?

- Organised on Business Capabilities
- **Infrastructure Automation (build, test, deploy)**
- Design for failure (resilience to failure)



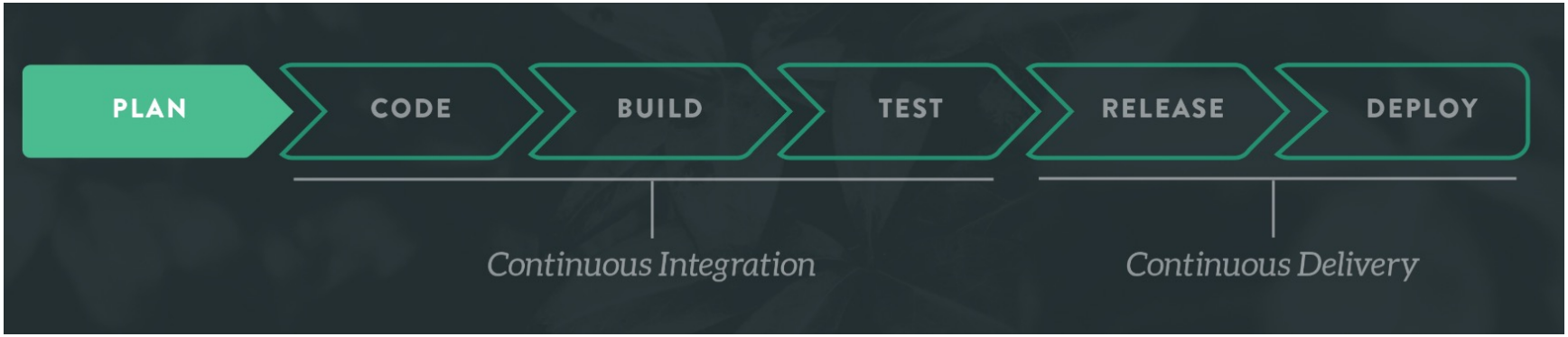
Infrastructure Automation

- Lots of microservices
- Lots of stages
- Deployment automation mandatory

Hold my orange juice while I deploy a Microservice!

Continuous Integration\ Continuous Delivery

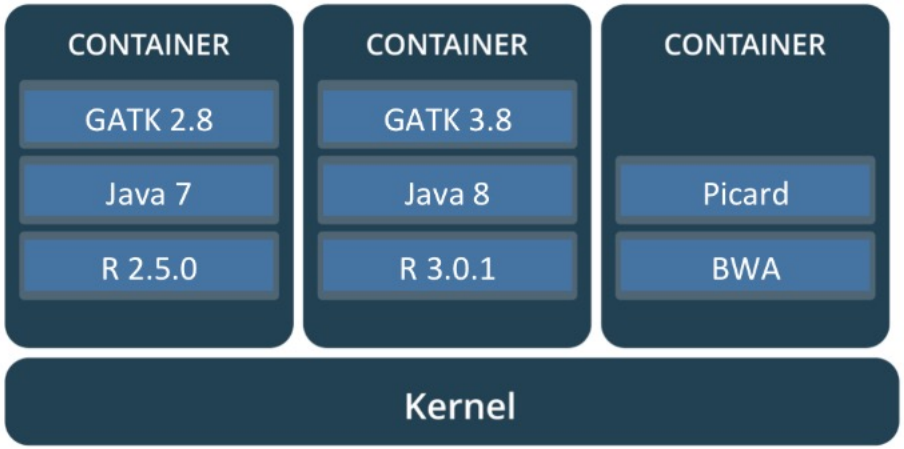
- Lots of microservices
- Lots of stages
- Deployment automation mandatory

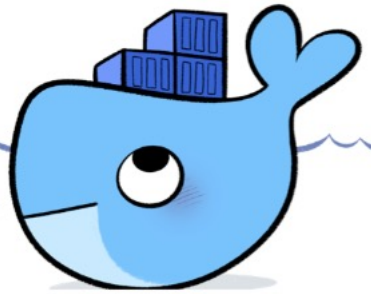


Why containers are needed

Containers

Containers allow to Package Software into **Standardized Units** for Development, Shipment and Deployment





A container encapsulates **all the software dependencies** associated with running a program

Takes the guesswork out of running on different platforms

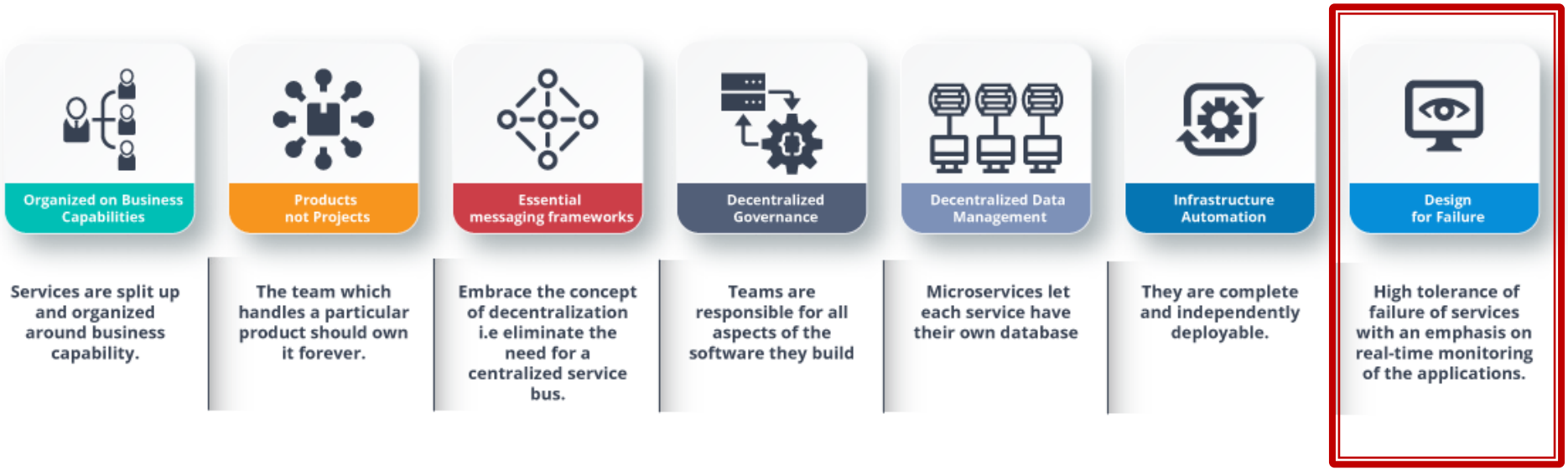
Each microservice should run inside its own container

Virtual Machines vs Containers

Virtual Machines vs Containers

What are the features of Microservices?

- Organised on Business Capabilities
- Infrastructure Automation (build, test, deploy)
- **Design for failure (resilience to failure)**



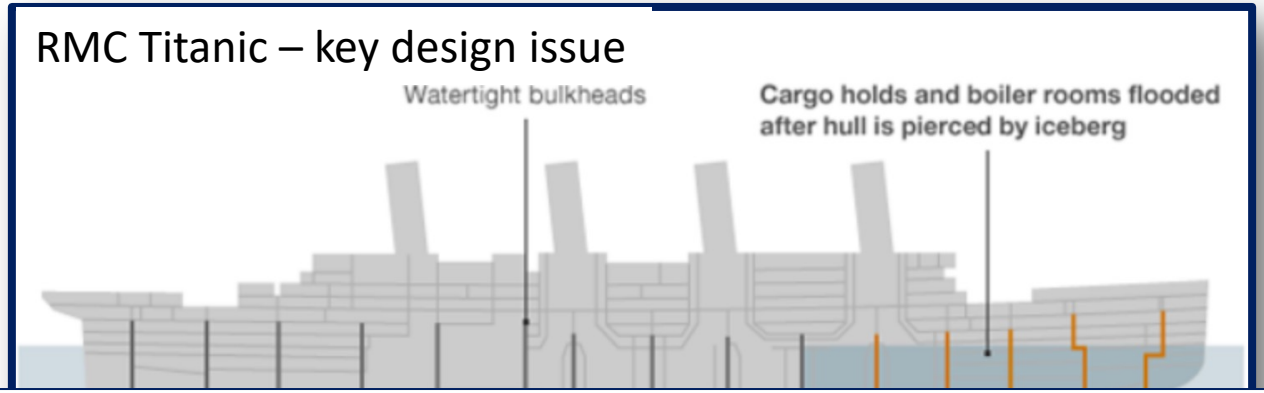
Design for failure

- Test for failure

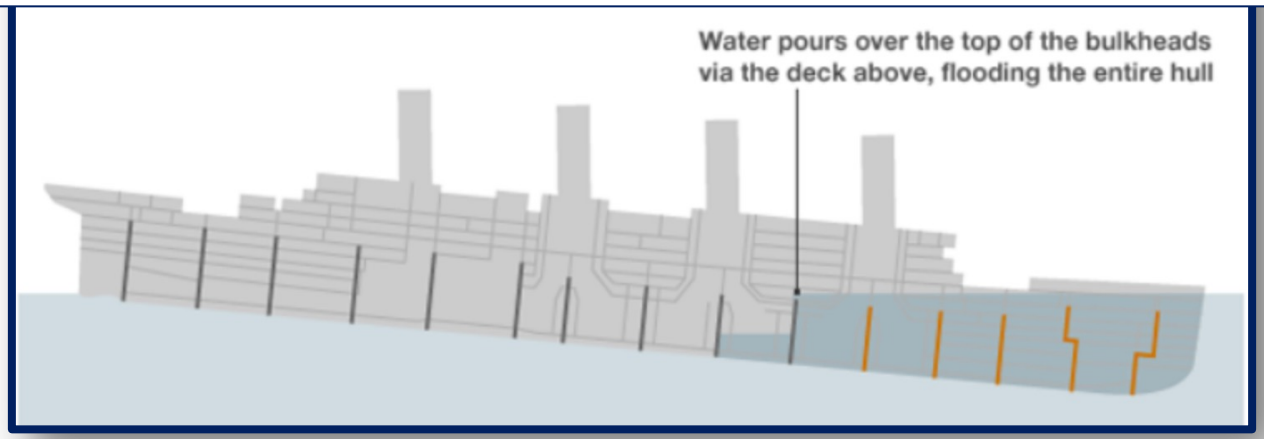
  - If a service fails, the system should stay alive.
  - Gave rise to chaos engineering Netflix has a set of services (Simian Army) that kills services at random to test that the system can stay alive.
  - Services should be stateless and have recovery strategy to survive the Simian Army

Design for failure: Bulkhead pattern

- Apply the bulkhead pattern to protect limited resource

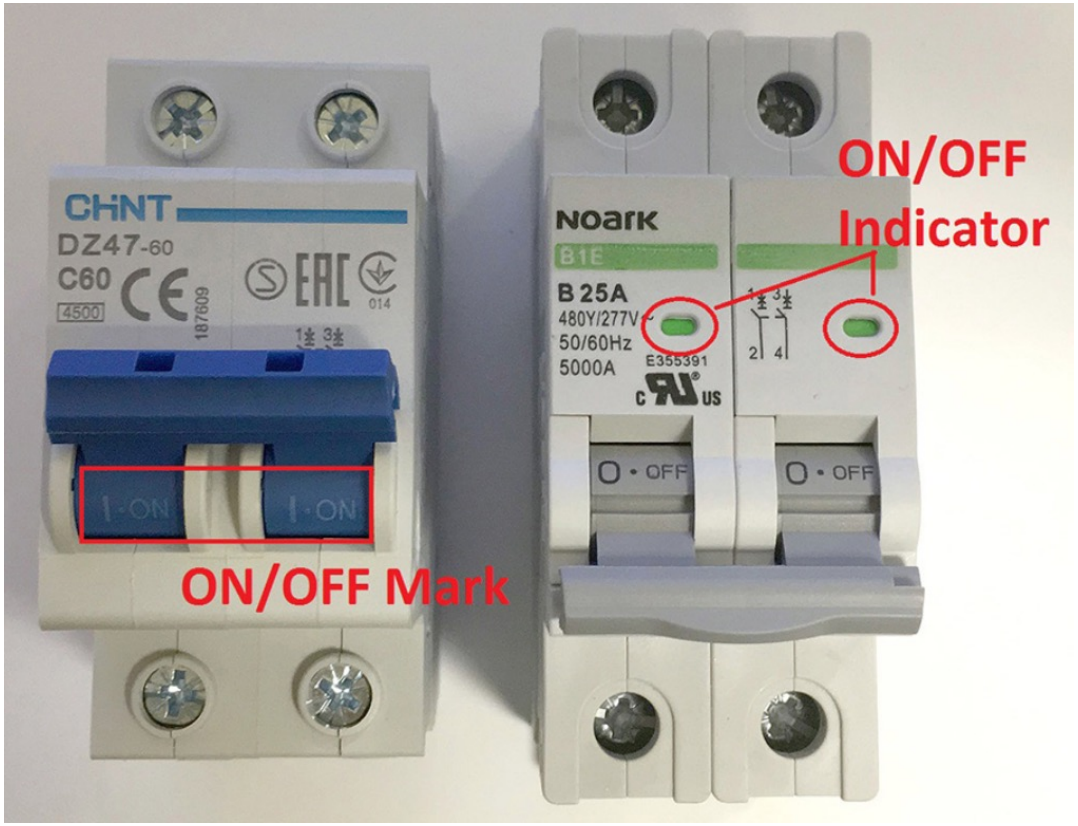


Premise: 'One fault shouldn't bring down the whole ship!



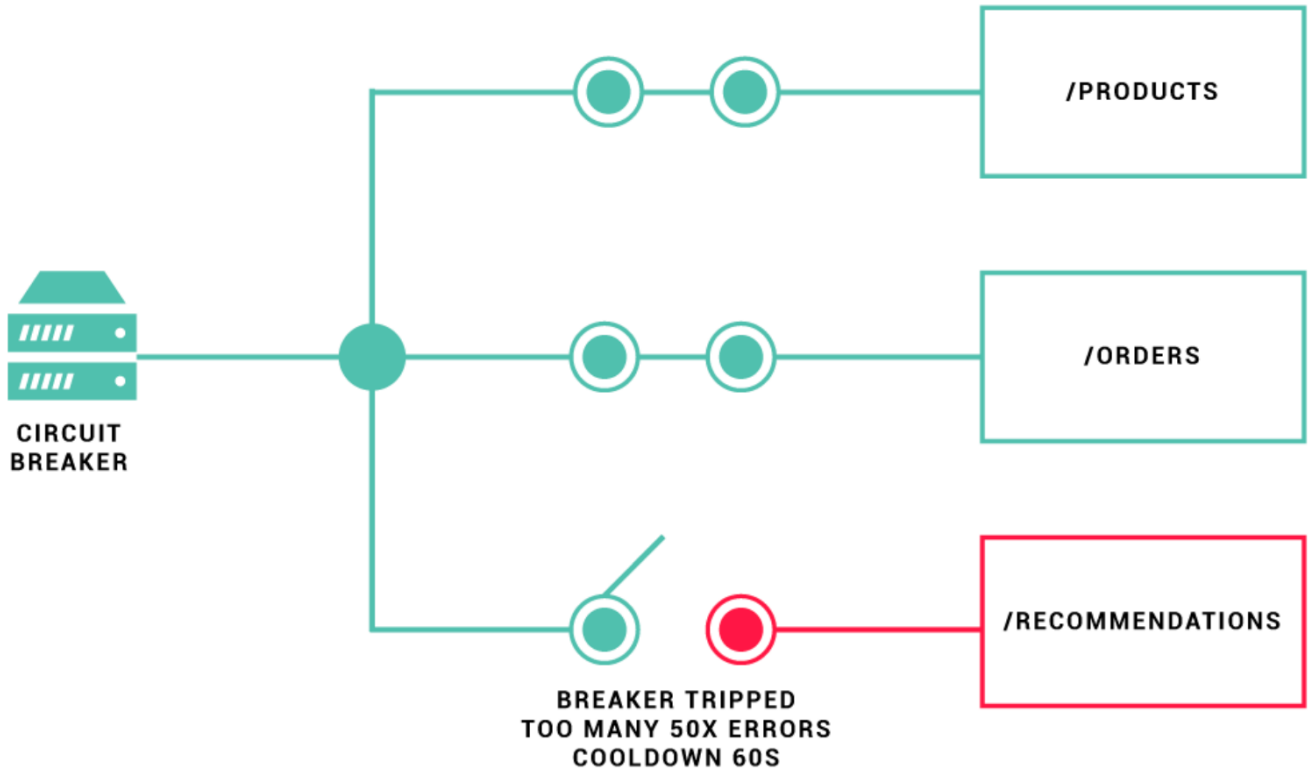
Source: http://writing.engr.psu.edu/uer/bassett.html

Design for failure: Circuit breakers pattern



Design for failure: Circuit breakers pattern

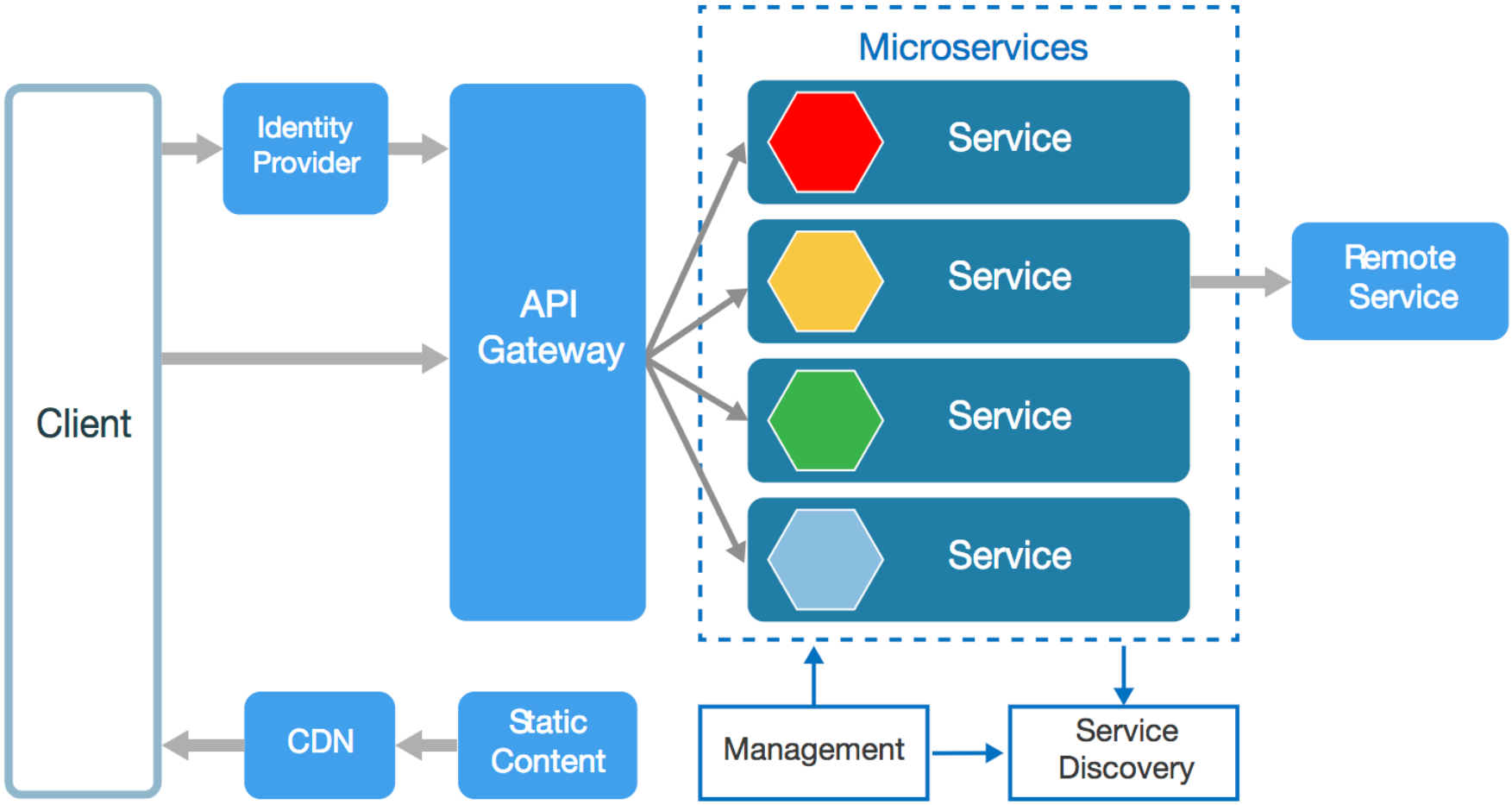
- Apply the circuit breakers pattern stopping requests to a service while it recovers



Design for failure: Blue\Green Deployment

- Maintain two versions of the product and slowly introduce it to chunks of customers

Main components in a microservice architecture



Technologies and tools

Container virtualisation

Container management

Service Discovery

Failure management

Continuous integration with git

…and many many many more…

Microservices – use cases

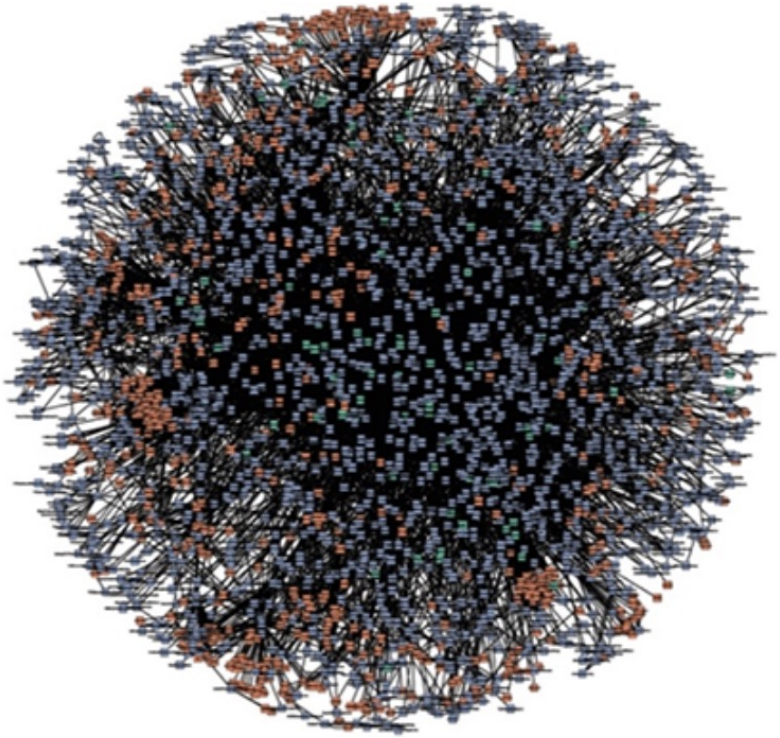
USER CASE STUDY 1/3

USER CASE STUDY 2/2

- Problem Statement
  - All the features had to be re-built, deployed and tested again and again to update a single feature.
  - Fixing bugs became extremely difficult in a single repository as developers had to change the code again and again.
  - Scaling the features simultaneously with the introduction of new features worldwide was quite tough to be handled together.

Microservices are NOT a silver bullet

The Death Star architecture



amazon.com

NETFLIX

Advantages and Disadvantages

Pros of Microservice Architecture	Cons of Microservice Architecture
Freedom to use different technologies	Increases troubleshooting challenges
Each microservices focuses on single capability	Increases delay due to remote calls
Supports individual deployable units	Increased efforts for configuration and other operations
Allow frequent software releases	Difficult to maintain transaction safety
Ensures security of each service	Tough to track data across various boundaries
Mulitple services are parallelly developed and deployed	Difficult to code between services

These lists are not in any way exhaustive!

Summary

- Microservices are a software architectural style
- Main principles
  - designed around business boundaries
  - deployed with Infrastructure automation
  - designed for failure
- Microservices are not a silver bullet
  - They introduce additional complexity

Reading

- **Martin Fowler article (and video) – an in-depth explanation of the main principles of a microsrvices architecture**
  - https://martinfowler.com/articles/microservices.html
  - https://www.youtube.com/watch?v=Irlw-LGIJO4
- **An article on designing microsrvice architecture that are resilient to failure** https://blog.risingstack.com/designing-microservices-architecture-for-failure/
- Daya, S. et al. (2015). "Microservices from Theory to Practice." IBM Redbook (SG24-8275-00).
  - https://www.redbooks.ibm.com/redbooks.nsf/RedbookAbstracts/sg248275.html
  - Chapter 1 & 2
- The UBER case study (from the slides):
  - https://dzone.com/articles/microservice-architecture-learn-build-and-deploy-a
- https://github.com/microservices-demo/microservices-demo
- **Look at the notes below the slides!**