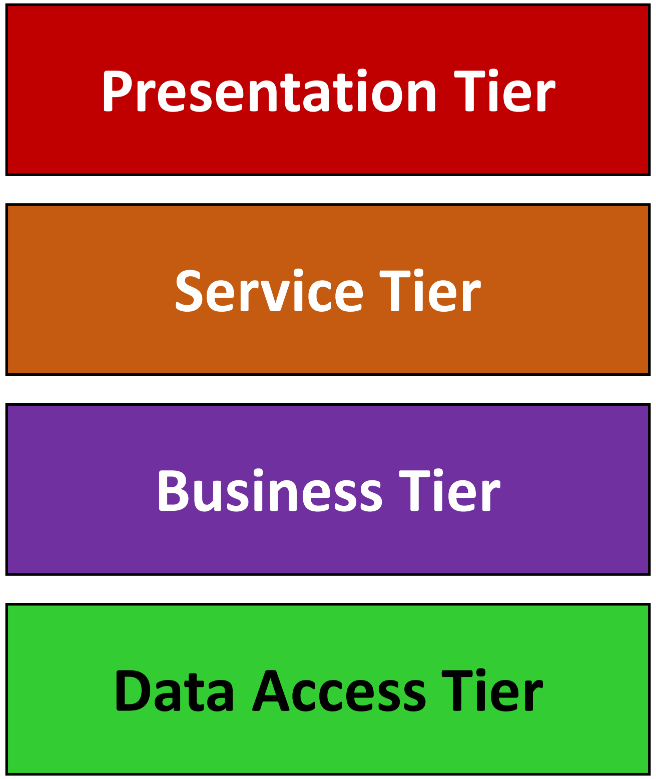
STACKUP

The 12 Factor App

Traditional Tiered Application

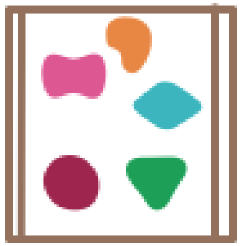
- Application implements all the requirements
- Application is structured around tiers
  - Each tier is responsible for some aspects of the total application
- Tiers are independent of each other logically
  - Coupled at the code
- A single database is shared across all tiers



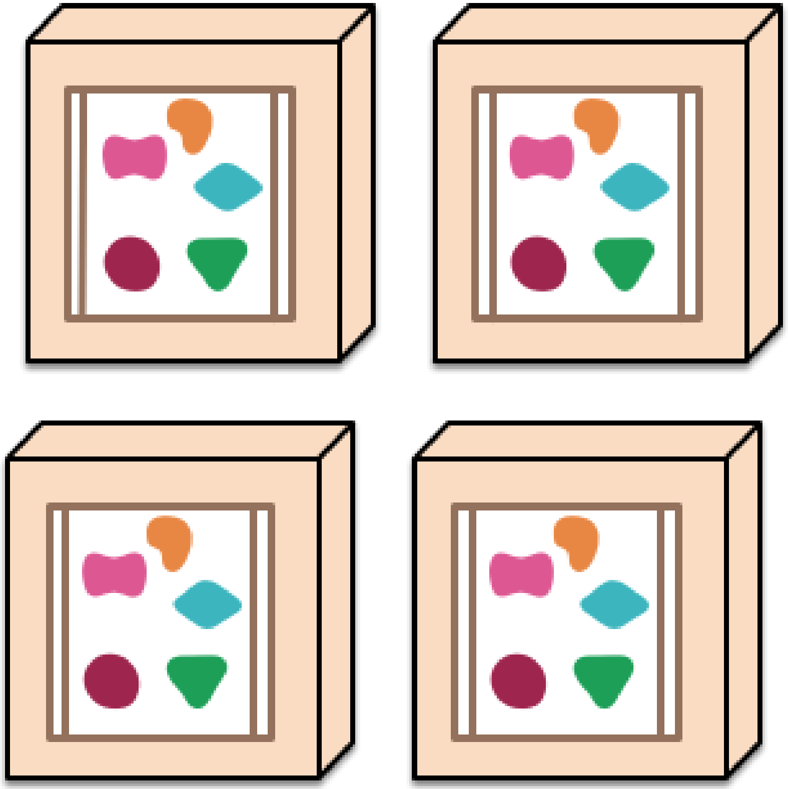
Monolithic Applications

- Monolithic application contains all the functionalities in a single application
- Application is scaled by cloning and running the entire application on multiple servers/VM/containers
- Applications typically organized around a service bus
  - Applications are services
  - Bus is the backbone

A monolithic application puts all its functionality into a single process...



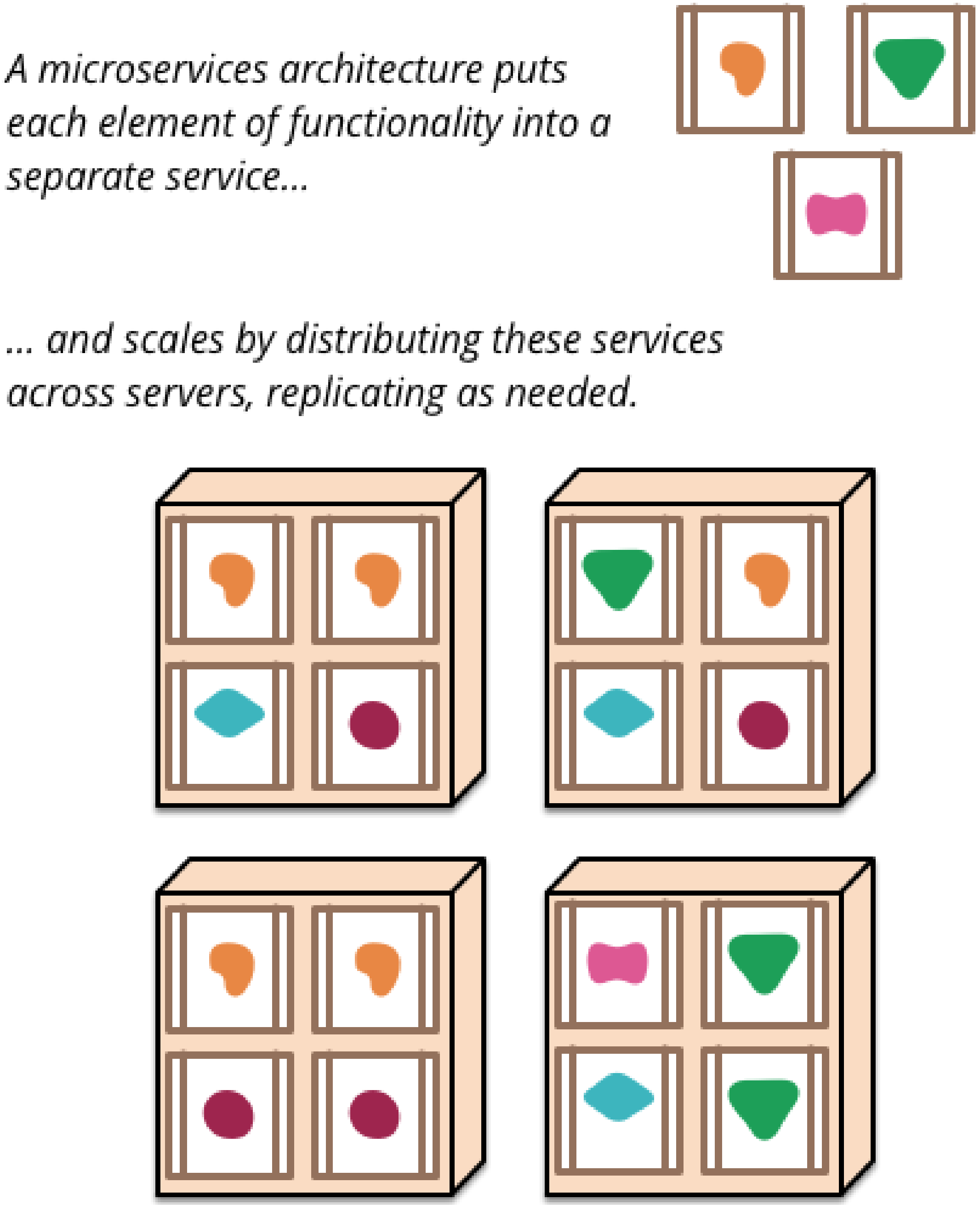
... and scales by replicating the monolith on multiple servers



Images from https://martinfowler.com/articles/microservices.html

Microservices

- Functions in an application are separated in to separate smaller services
- Each service is deployed into its own servers/VM/containers
    - Each service own its own data
- Only need to deploy the application’s services
- One or more services work together to deliver a business function



A microservices architecture puts each element of functionality into a separate service...

... and scales by distributing these services across servers, replicating as needed.

Images from https://martinfowler.com/articles/microservices.html

Refactoring a Monolithic Application

- Breakup applications according to their context
  - Context is dependent on the subject domain
- Service owns and manage the data model and data
  - Bound to the context
- A context cannot update data belonging to another context
- Explicit relationships between contexts/services

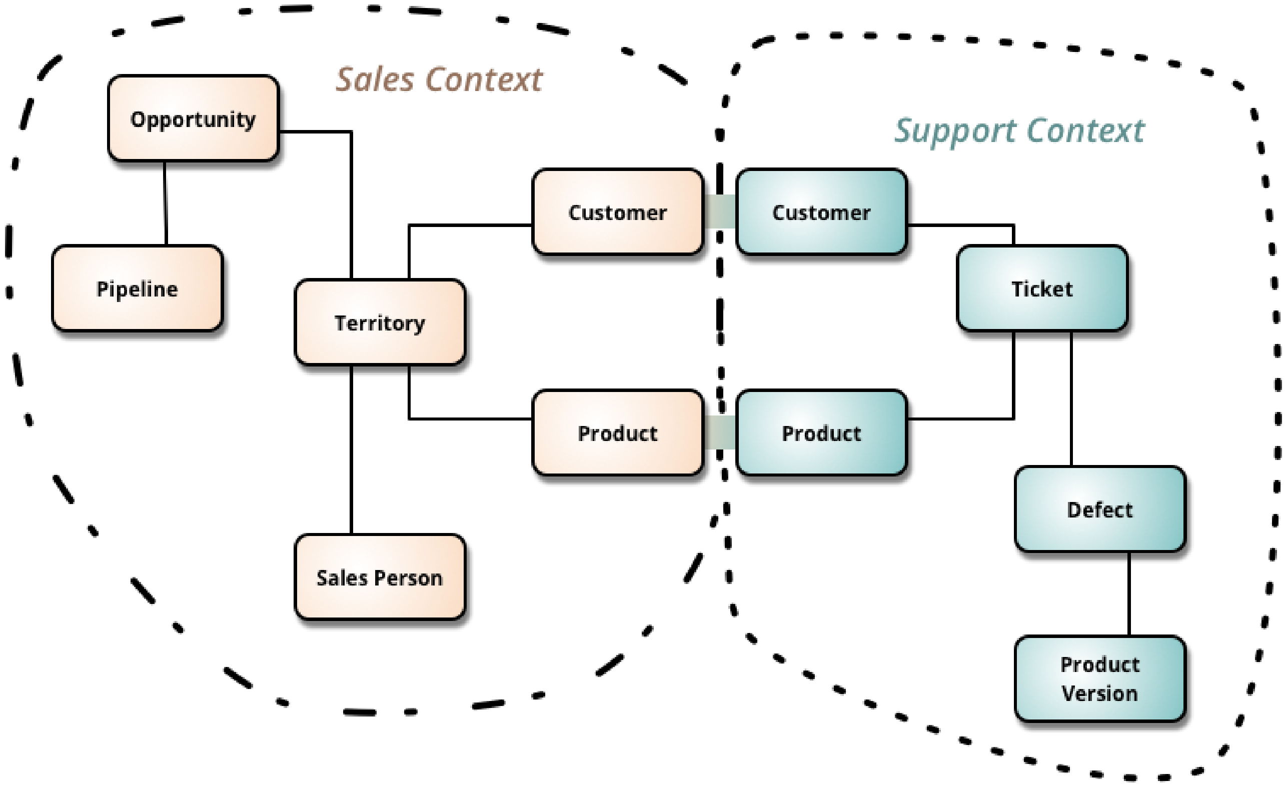


Image from https://martinfowler.com/bliki/BoundedContext.html

Microservices Communications

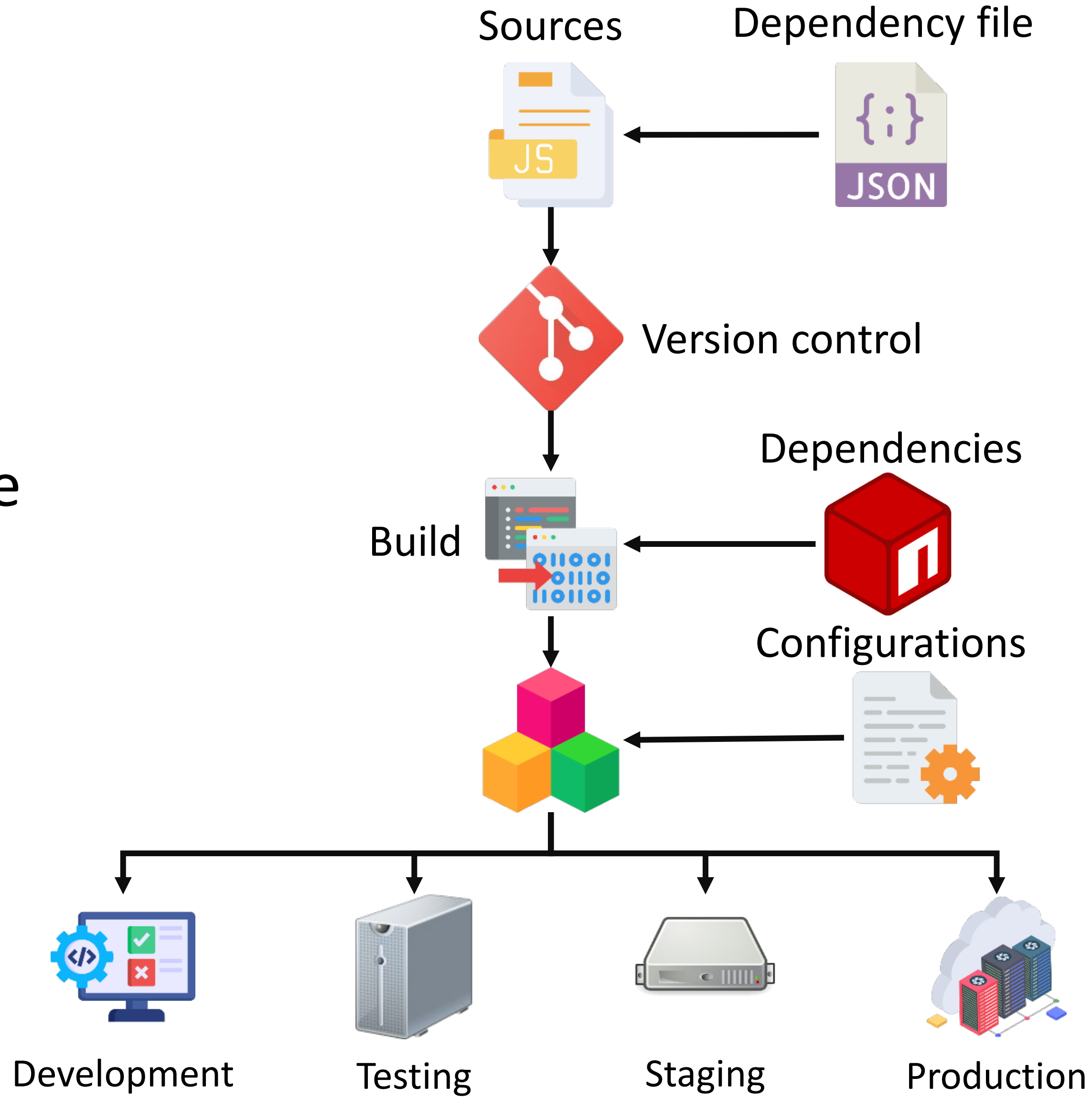
- Can be grouped into synchronous and asynchronous
- Synchronous
  - Request/Response typically over HTTP
- Asynchronous
  - Event driven with queues and message bus
    - Event sourcing
  - File upload typically with object store like S3
    - Use case - batch updates

What is the 12 Factor App?

- Software development methodology for building applications using the microservices approach
    - Drafted by developers in Heroku, presented by Adam Wiggins circa 2011
- Includes best practices to allow application to scale, portable and resilient to failure when deployed to the web
- Consider as part of how to develop a cloud native application
- Most of the ‘factors’ are applicable to popular runtime
    - Python, JavaScript
- Criticism that the methodology is specific to Heroku

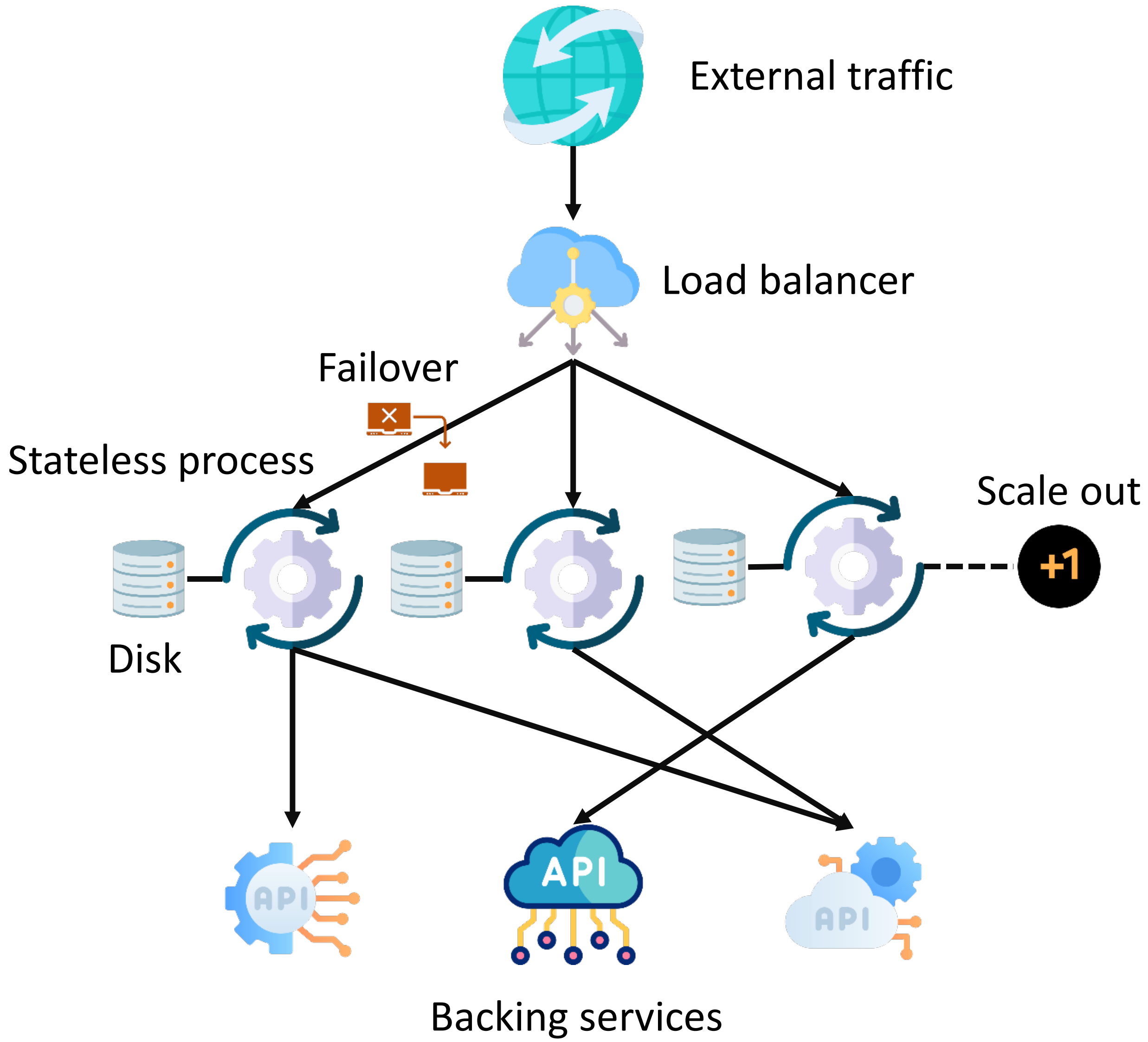
Development Characteristics of Micro Services

- Codebase - One codebase tracked in revision control, many deploys
- Dependencies - Explicitly declare and isolate dependencies
- Config - Store configurations in the environment
- Build, Release, Run - Strictly separate build and run stages
- Dev/Prod Parity - Keep development, staging and production as similar as possible

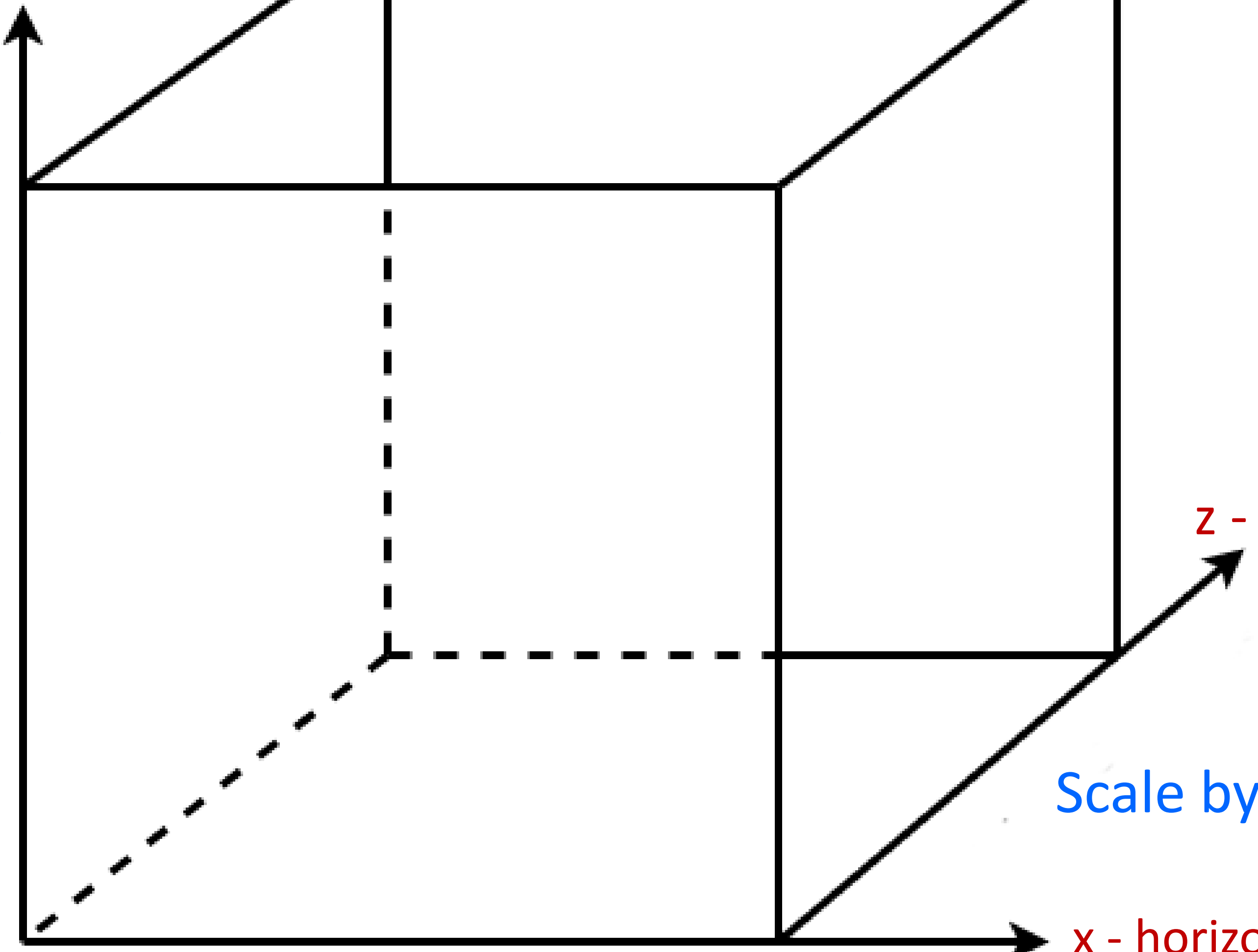


Deployment Characteristics of Micro Services

- Processes - Execute the app as one or more stateless processes
- Concurrency - Scale out via the process model
- Backing Services - Treat backing services as attached resources
- Disposability - Maximize robustness with fast startup and graceful shutdown



How to Scale?



y - functional decomposition

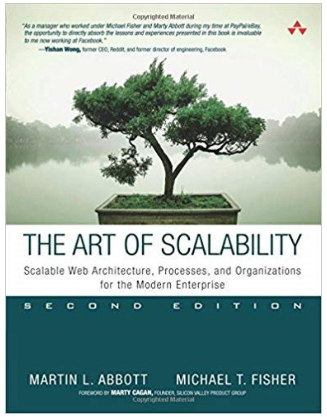
Scale by splitting application into smaller modules

z - data partitioning

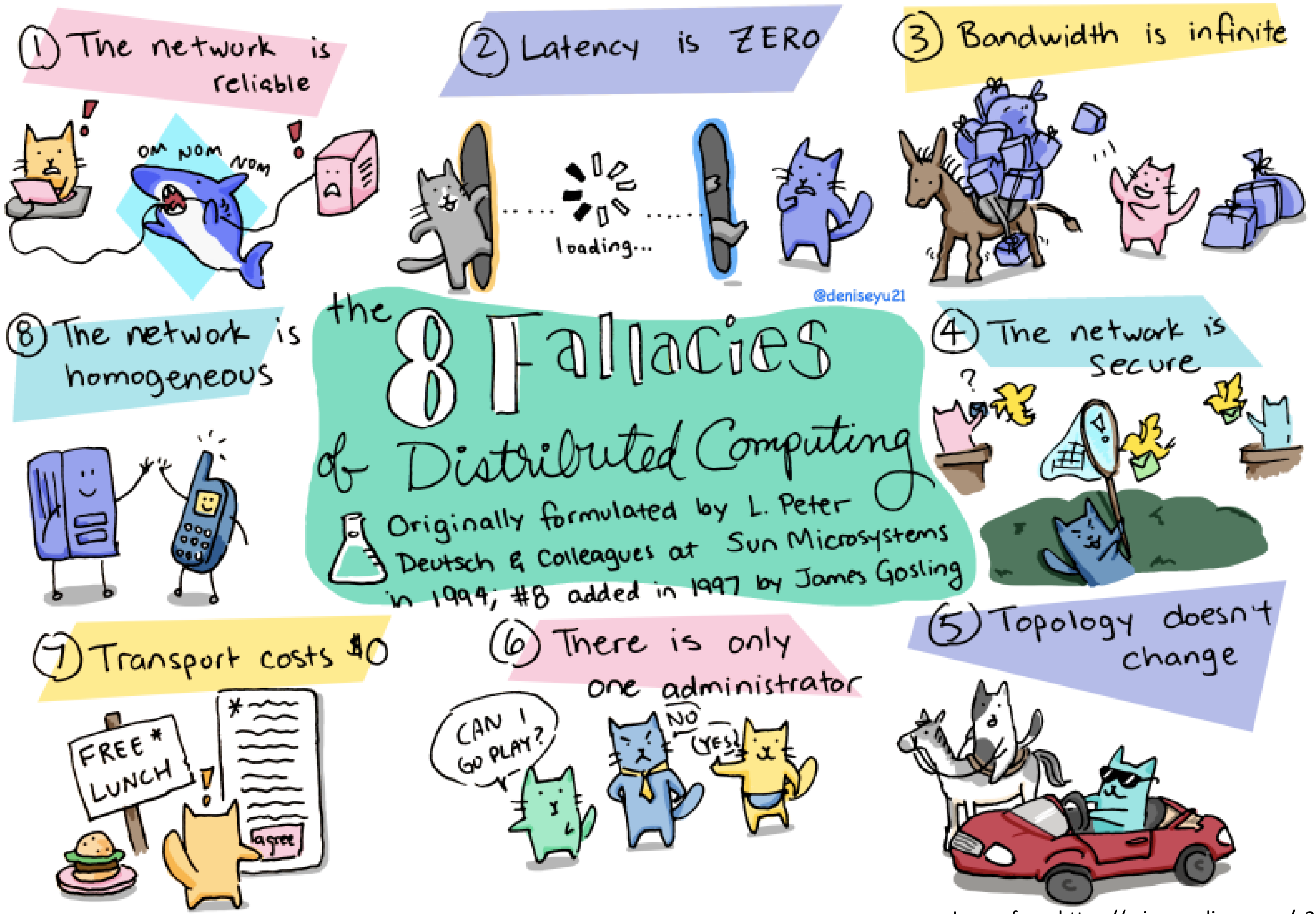
Scale by sharding the database

x - horizontal duplication

Scale by cloning the application


THE ART OF SCALABILITY
Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise
SECOND EDITION
MARTIN L. ABBOTT
MICHAEL T. FISHER

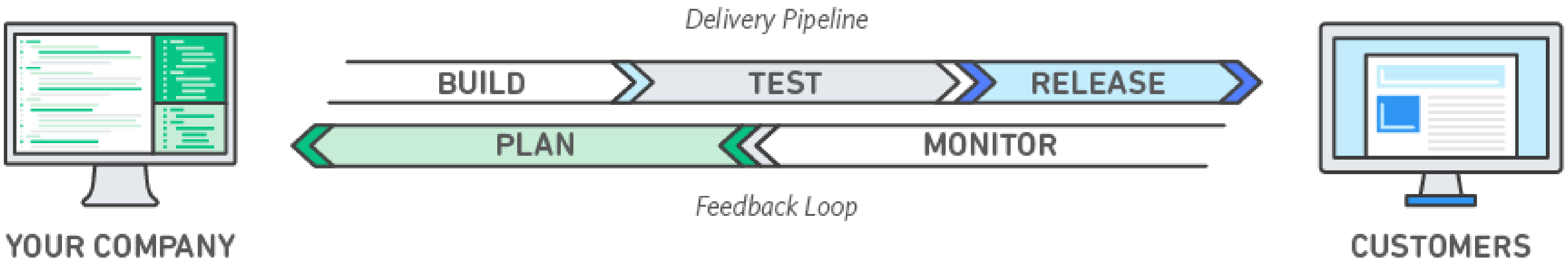
8 Fallacies of Distributed Network Systems



1 The network is reliable
OM NOM NOM
2 Latency is ZERO
loading...
3 Bandwidth is infinite
@deniseyu21
8 The network is homogeneous
the 8 Fallacies of Distributed Computing
Originally formulated by L. Peter Deutsch & Colleagues at Sun Microsystems in 1994; #8 added in 1997 by James Gosling
4 The network is secure
?
7 Transport costs $0
FREE* LUNCH
agree
6 There is only one administrator
CAN I GO PLAY?
NO
YES!
5 Topology doesn't change

Image from https://miro.medium.com/v2/resize:fit:797/0*4u89K8_yoVlwar4o.png

What is DevOps?

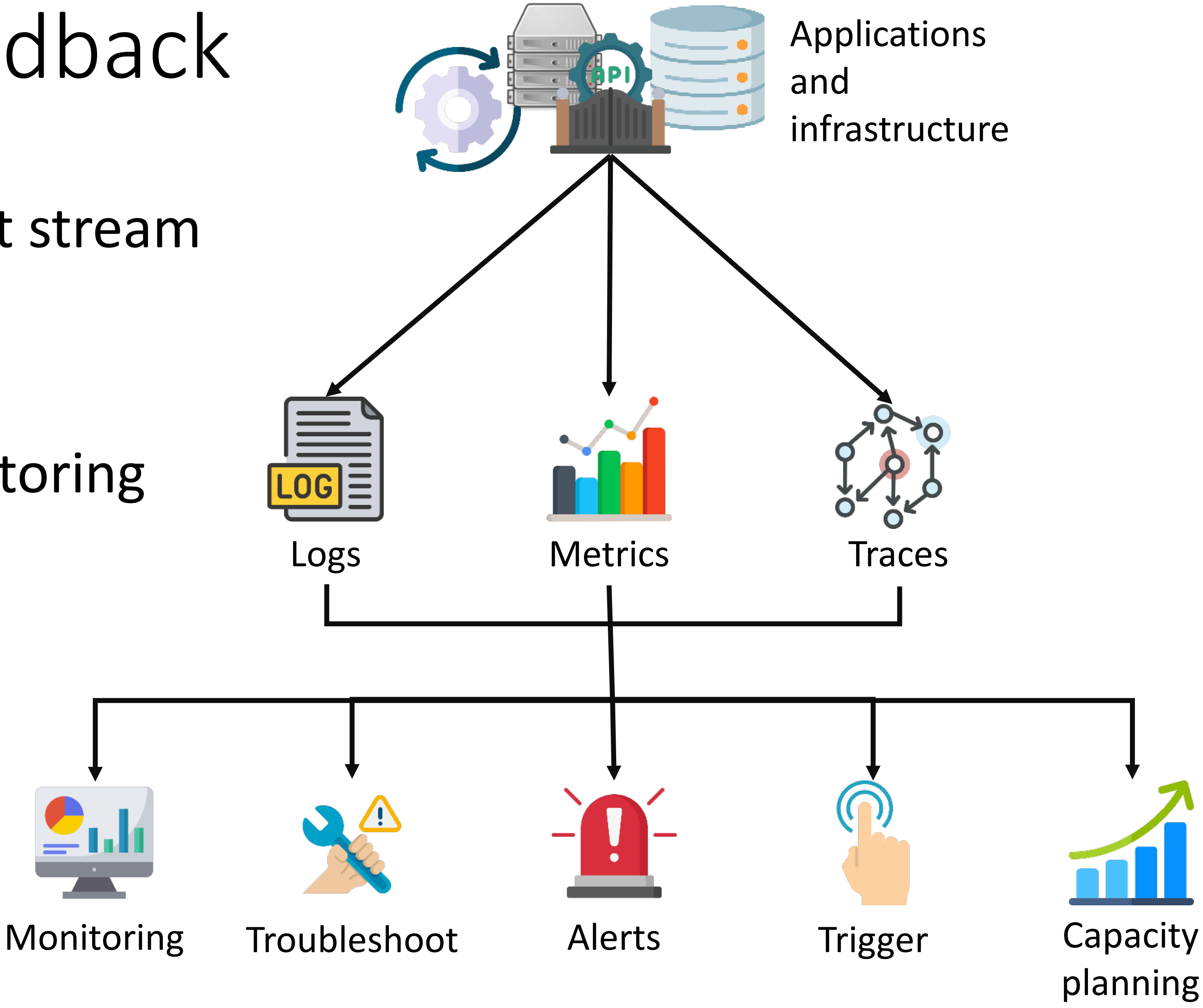


- DevOps is a combination of culture, practices and tools
- Aim is to shorten the systems development lifecycle
  - By streamlining software building, testing and release
- Benefit is the improve and evolve applications at a faster pace

Image from https://aws.amazon.com/devops/what-is-devops/

Monitor and Feedback

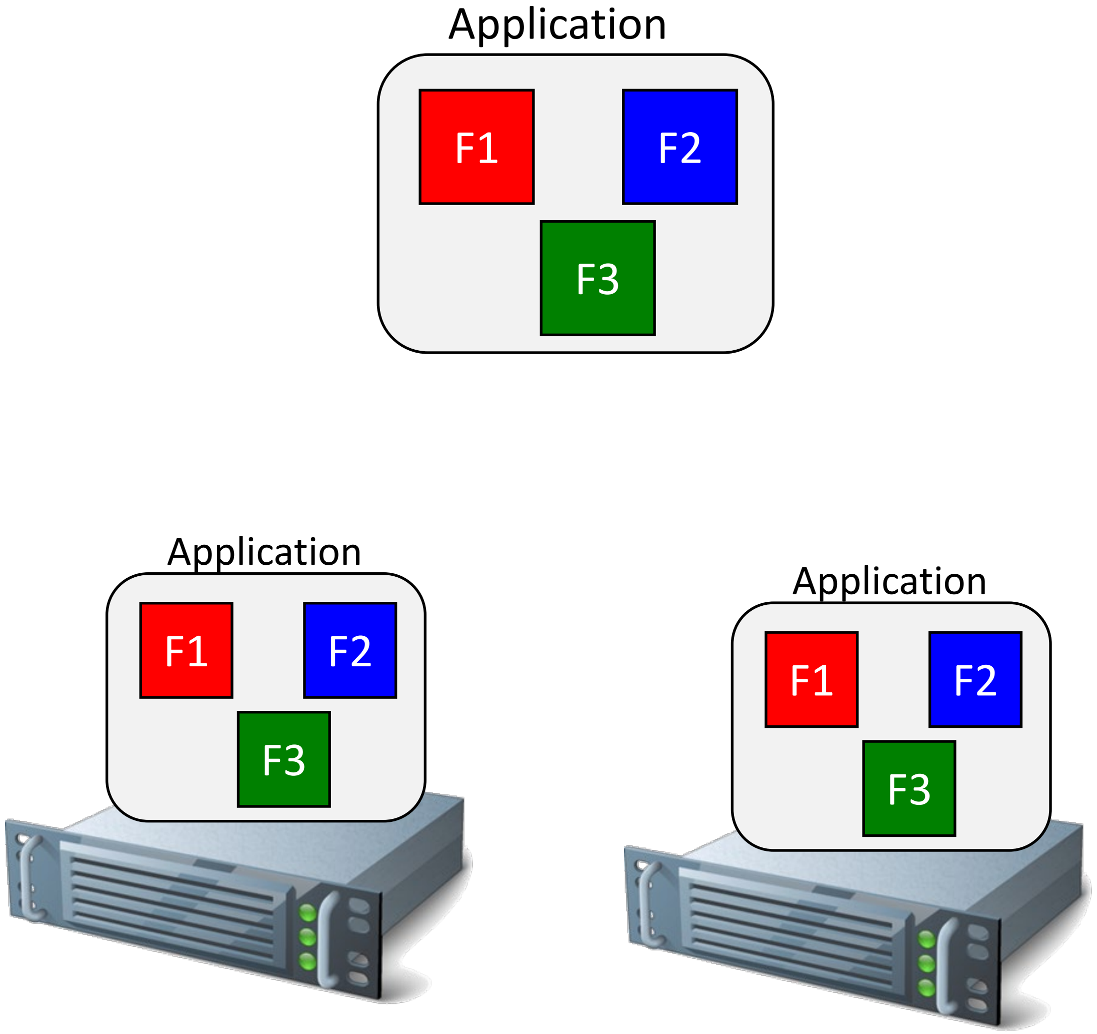
- Logs - Treat logs as event stream
- Observability
  - Logs, traces, metrics
- 4 golden signals of monitoring
  - Latency
  - Traffic
  - Saturation
  - Error



Appendix

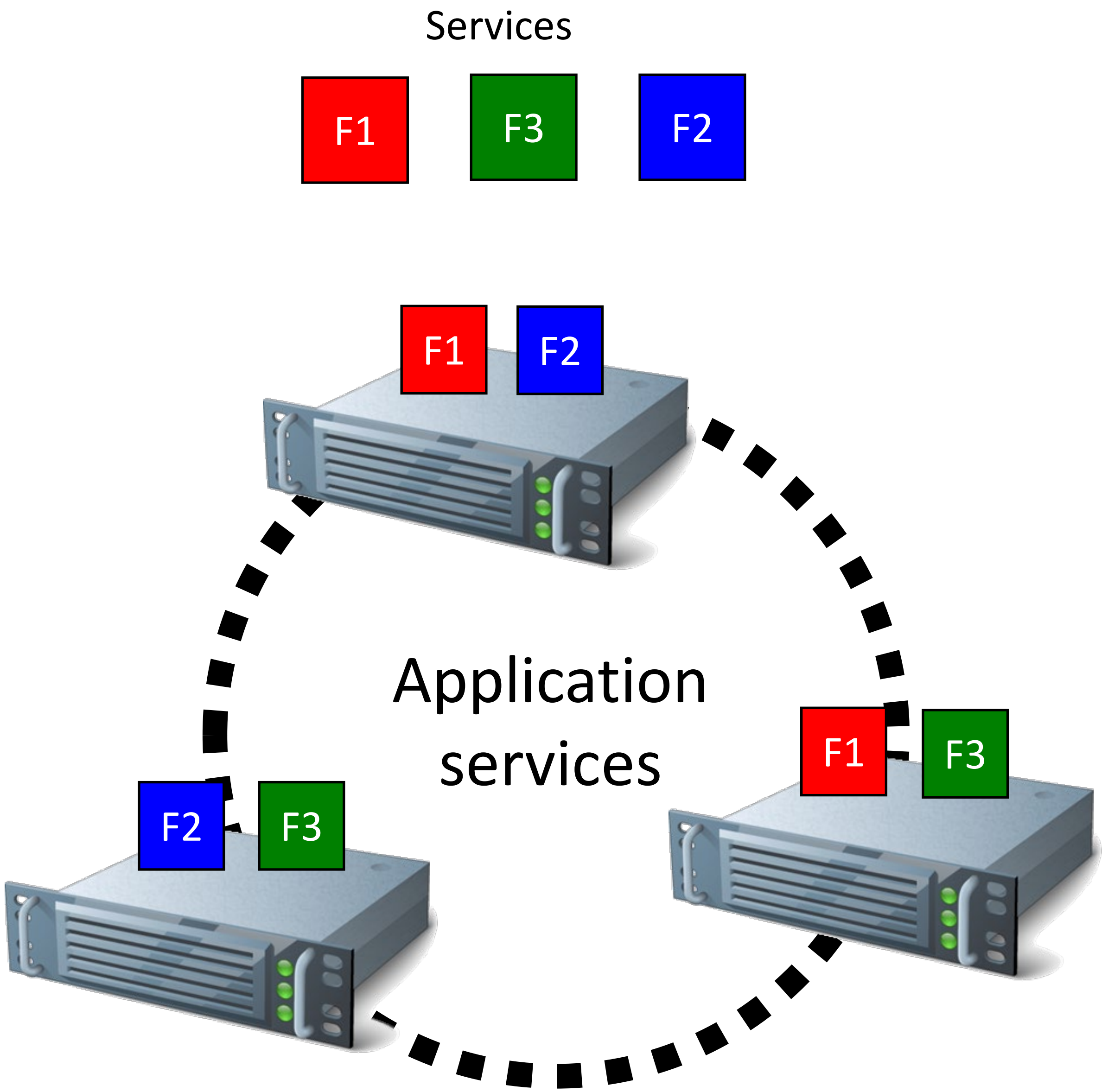
Monolithic Application

- Monolithic application contains all the functionalities in a single application
- Application is scaled by cloning and running it on multiple different servers/VM/containers



Microservices Approach

- Functions in an application are separated in to separate smaller services
- Each service is deployed into its own servers/VM/containers
  - Each service own its own data
- Only need to deploy the application's services
- Services work together to deliver the application service



Service Decomposition

- Loose coupling
  - Changes in one service should not require a change to another service
  - Services should know as little as possible about the service that it is interacting with
- High cohesion
  - Related behaviours to be in the same service

- Bounded context
  - Service owns and is responsible for the data/message

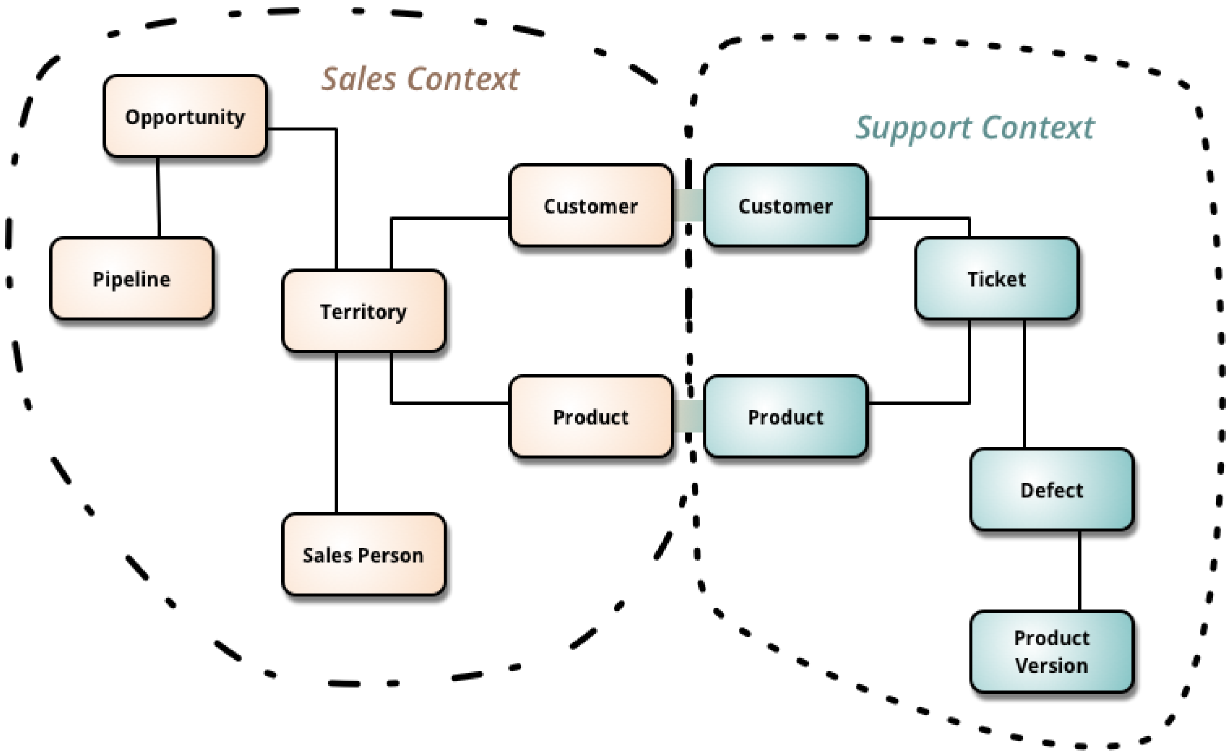


Image from https://martinfowler.com/bliki/BoundedContext.html

DevOps

Dev

- Code base
- Dependencies
- Configurations
- Backing services
- Dev/ops parity
- Build, release, run

Ops

- Processes
- Port binding
- Concurrency
- Disposability
- Dev/ops parity
- Logs
- Admin processes