# Cloud Microservices Programing Introduction

MOSTAFA RASTGAR MAY 2019

#### Workshops Agenda

- Microservices introduction
- Spring boot and rest webservices
- Design patterns for microservices
- Config Servers
- Service Registry
- Client side & Centralized load balancing
- Circuit breaker
- Kafka and Redis Introduction
- Spring cloud streaming
- Service call tracing
- Security

#### Today is Microservices introduction

- What are microservices?
- What is domain driven design?
- Microservices based application architecture
- Run the sample demo
- Our workshops sample project

#### What are microservices?

- Breaking the application into smaller pieces which compose together
- Each component is continuously developed and separately maintained
- The application is then simply the sum of its constituent components
- In contrast to a traditional, "monolithic" application which is all developed all in one piece
- There are other benefits:
- There are other benefits:
  - Developer independence: Small teams work in parallel and can iterate faster than large teams.
  - Isolation and resilience: If a component dies, you spin up another while and the rest of the application continues to function.
  - Scalability: Smaller components take up fewer resources and can be scaled to meet increasing demand of that component only.
  - Lifecycle automation: Individual components are easier to fit into continuous delivery pipelines and complex deployment scenarios not possible with monoliths.
  - ▶ **Relationship to the business**: Microservice architectures are split along business domain boundaries, increasing independence and understanding across the organization.

## How to break the application into smaller pieces



Domain Driven Design can be a good starting point

#### What is domain driven design

- A formal definition:
  - ▶ When we are developing a software our focus shouldn't be primarily on technology, it should be primarily on business or whatever activity we are trying to assist with the software, the domain.
  - Specifically we approach that by trying to develop models of that domain and make our software conformed to that.
- Domain driven design is a way of looking at software from top down

#### What is top down?

Tactical Design Tools

Service

Project

Layers

Modules

Design Patterns

OOP

Classes

Objects

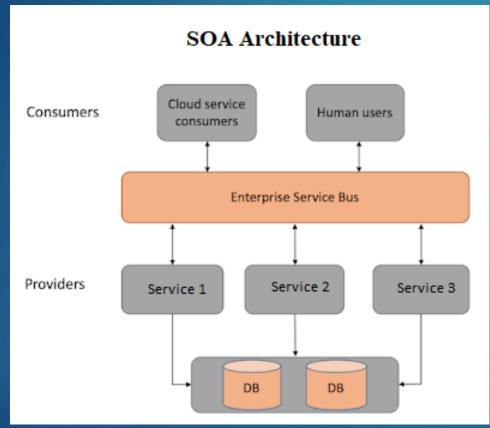
Exe / Jar / Zip

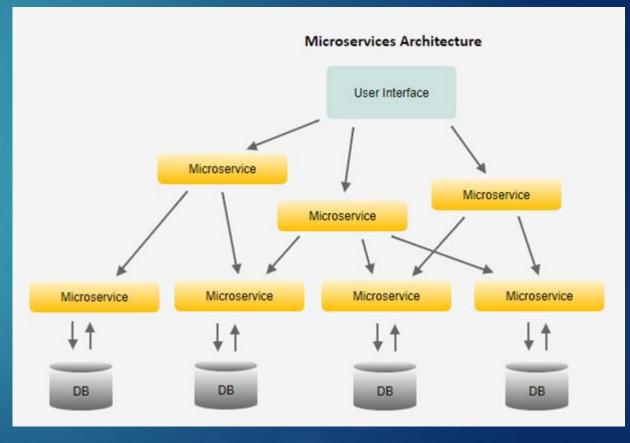
Strategic Design Tools



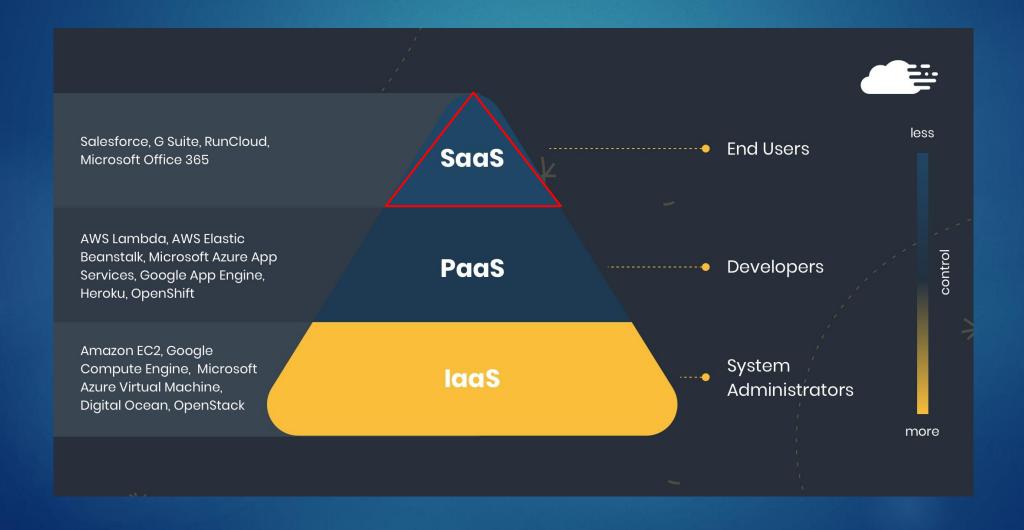
## What will our application be like when we break into microservices?







### Cloud Pyramid



Run sample demo

currency-converter-feign/from/USD/to/SEK/quantity/1000

Currency Conversion

> Eureka Service Registry

Replication

Forex Service

Forex Service

#### Our workshops sample project

#### Payment Project Validation Final Process Receive Payment Order Receive a new Syntax Send to Payment Order beneficiary • Semantic Insert ledger Withdraw the available balance Validation Final Process Receive a Negative Status Receive a Syntax Cancel the negative status original payment Semantic •Send the final Deposit the status to the both available beneficiary and balance originator

