



# Building Distributed Applications using Microsoft Orleans

Praveen Raghuvanshi  
@praveenraghuvan

LET'S BEGIN NOW!

# INTRODUCTION!



- Cloud Architect @  HARMAN  
A SAMSUNG COMPANY
- Domain: Professional Audio, Video & Control
- Area of Expertise: Cloud, Distributed computing
- Area of Interest: AI/ML, Cloud and IoT
- Location: Bangalore, India
- Azure certified
- Member  NET  
foundation

# AGENDA

**01** Actor Model  
Threading, Concurrency, Actors, Different APM frameworks

**02** Project 'Microsoft Orleans'  
History, About, Use cases

**03** Components of Orleans  
Grains(Virtual Actors), Silo, Cluster

**04** Deployment Models  
In-process, Single or Multiple host, Cloud

**05** Dashboard

**06** Demo

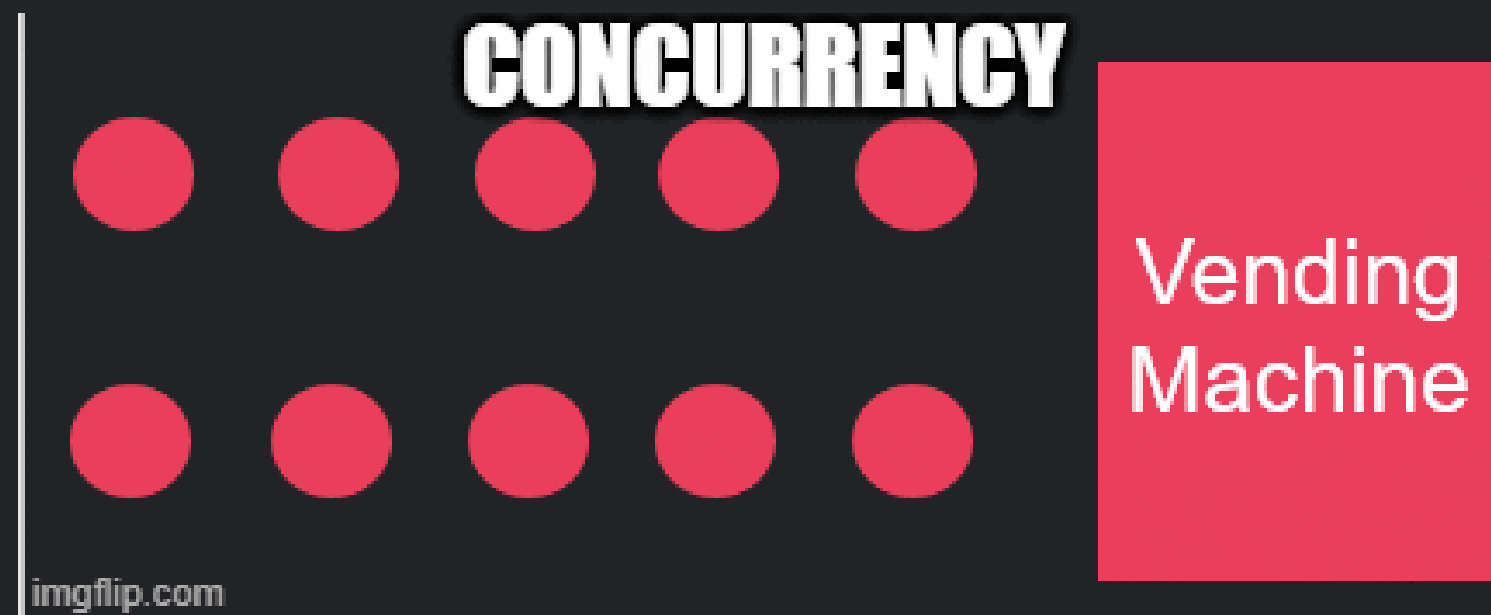




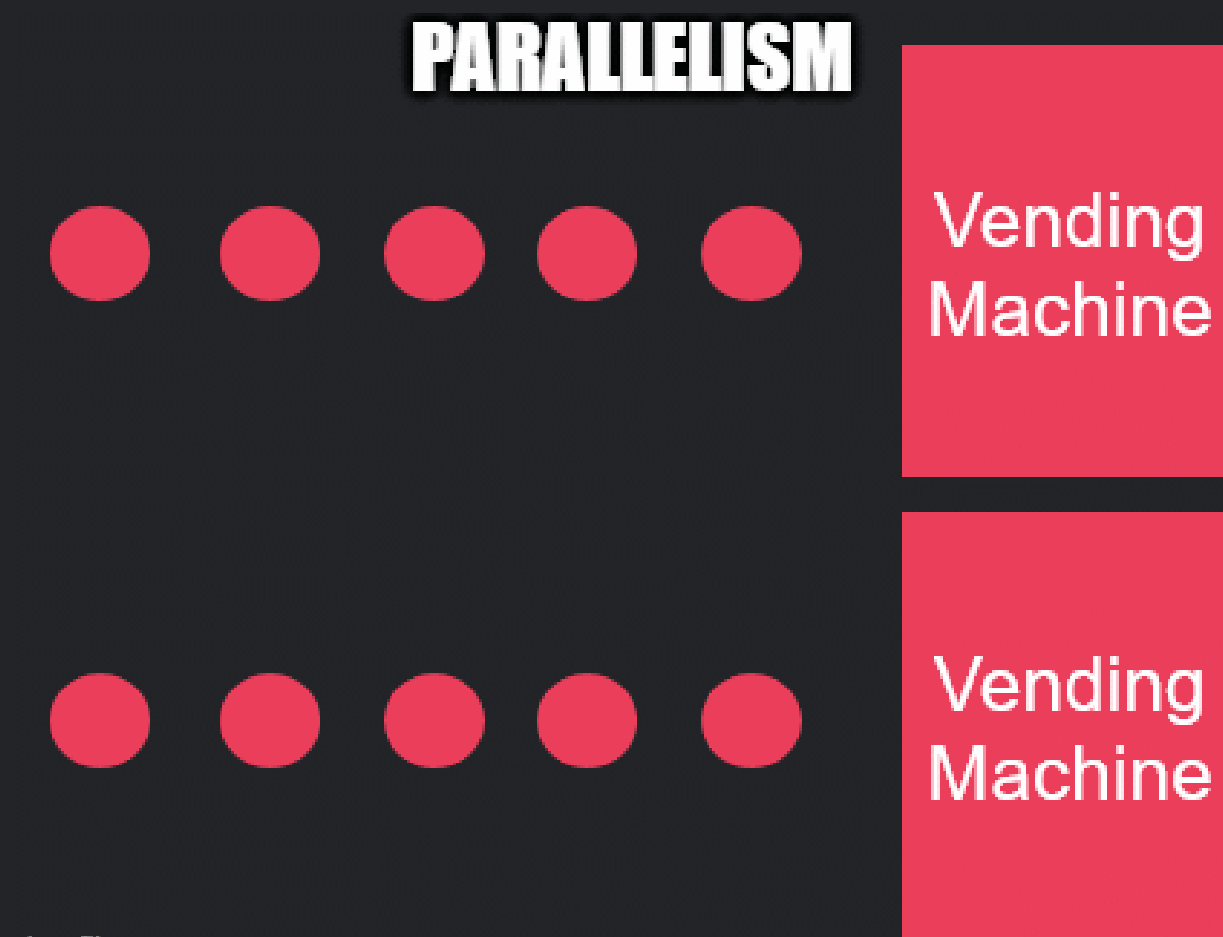
# 01 ACTOR MODEL



# THREADING, CONCURRENCY AND PARALLELISM

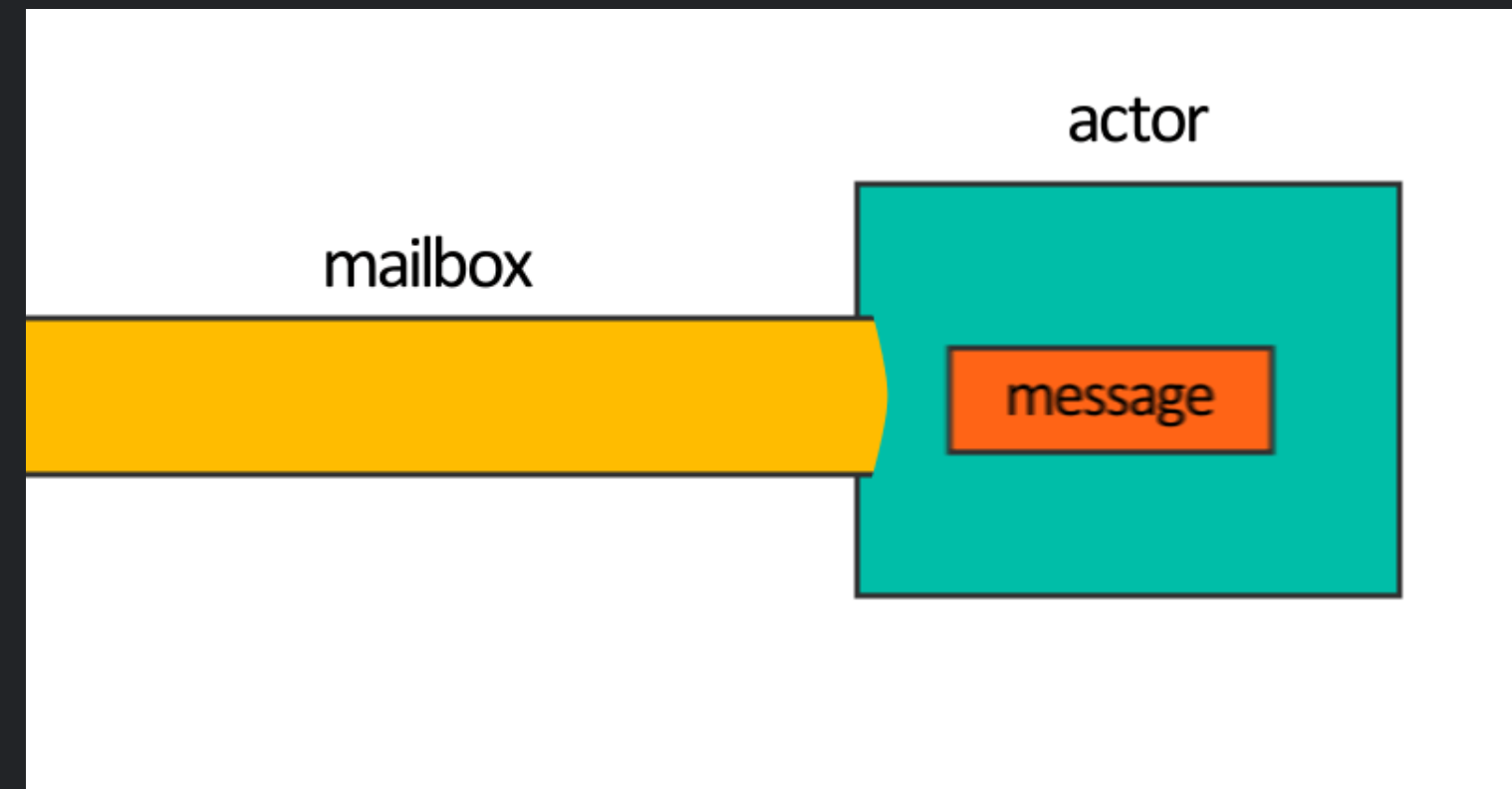
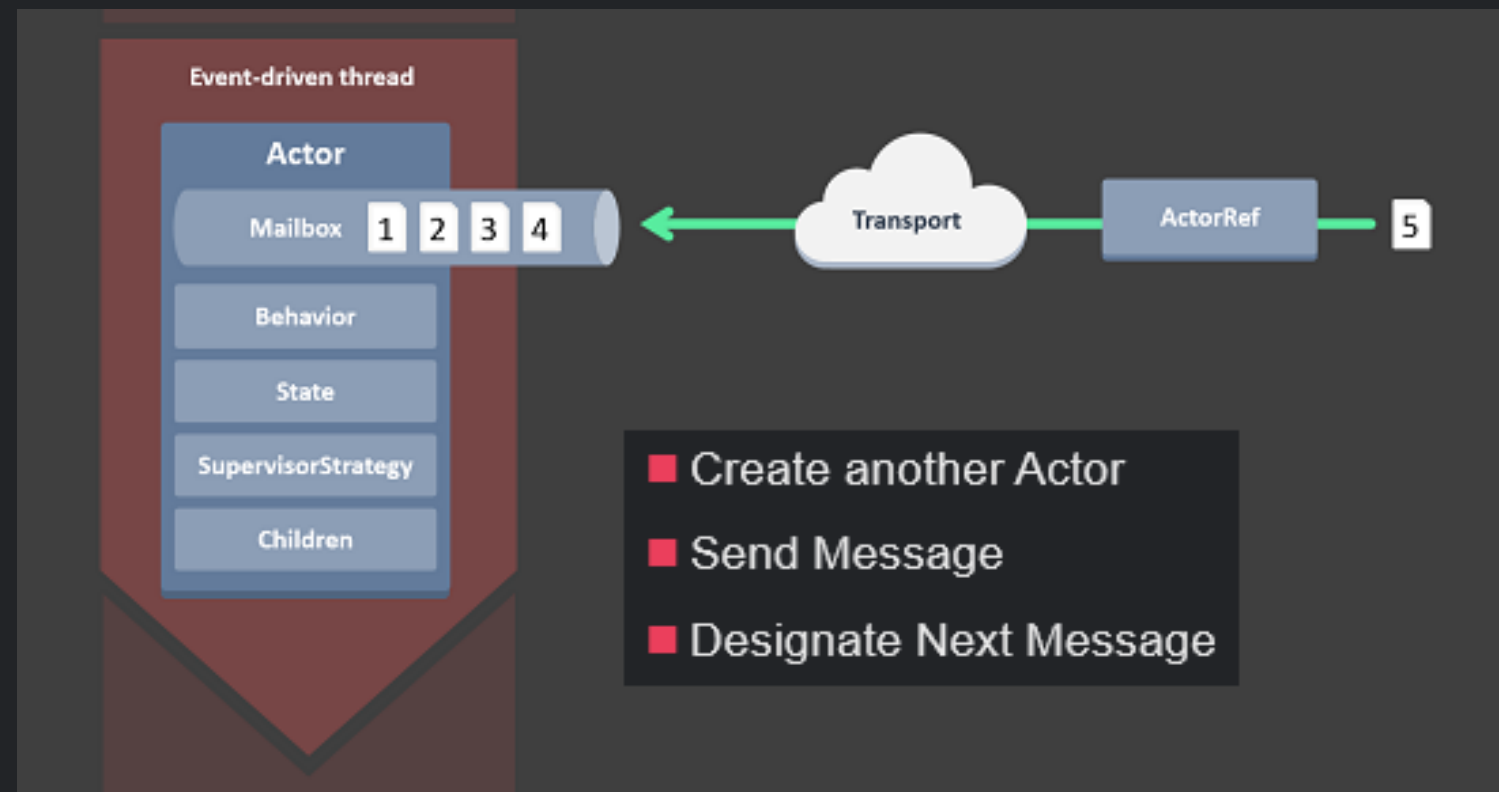


- Simultaneously, not parallel. (context switch)
- A logical distinction.
- Can be worked with single core



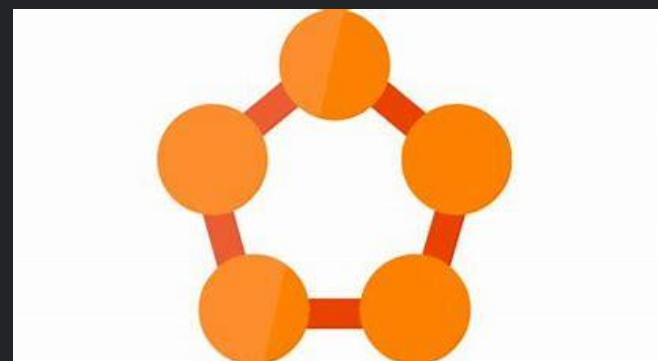
- Processor's power physically
- Multi-core

# ACTOR MODEL



- Invented by Carl Hewitt in 1973
- An actor is a computer process with an address.
- It encapsulates state and behavior within it.
- Message passing and async communication
- Single thread execution

- Sequential message processing
- Location Transparency
- No data sharing.
- No locks, thread management and concurrency issues
- Easy to scale, highly performance and fault-tolerant.



# ACTOR MODEL FRAMEWORK





## 02 MICROSOFT ORLEANS





# GOALS

## SIMPLE

Simple for developers, widely accessible programming model.

## SCALABLE

Scalable, something that can be deployed to single or multiple node.

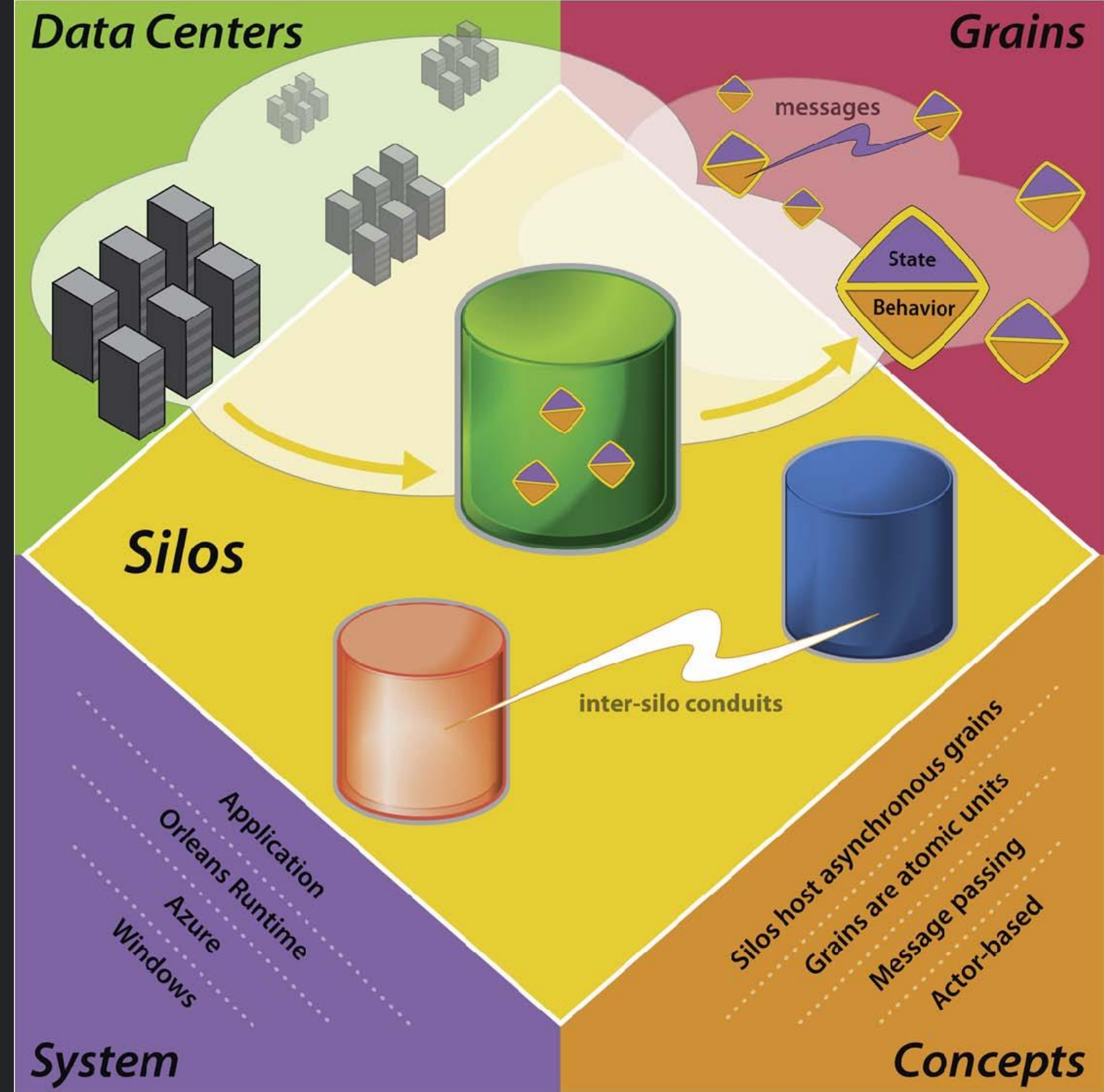
## ABSTRACTED

Abstract the underlying intricacies of distributed computing/programming.

# OVERVIEW

- Created by Microsoft Research
- Open Sourced in January 2015
- Built with Azure in mind
- Based on Virtual actor
- Used in Halo 4 and 5
- Used in various Azure services
- Support multiple deployment
- No Concurrency Issues





# 03 ORLEANS COMPONENTS



User/jack@email.com

In-memory  
or persisted

Grain = identity + behavior [+ state]

```
class User : Grain, IUser
```

## ACTOR

- Grains are implementations of Actors
- They have an identity
- Loosely Coupled(Backed by an Interface)
- Terminated to free compute resources

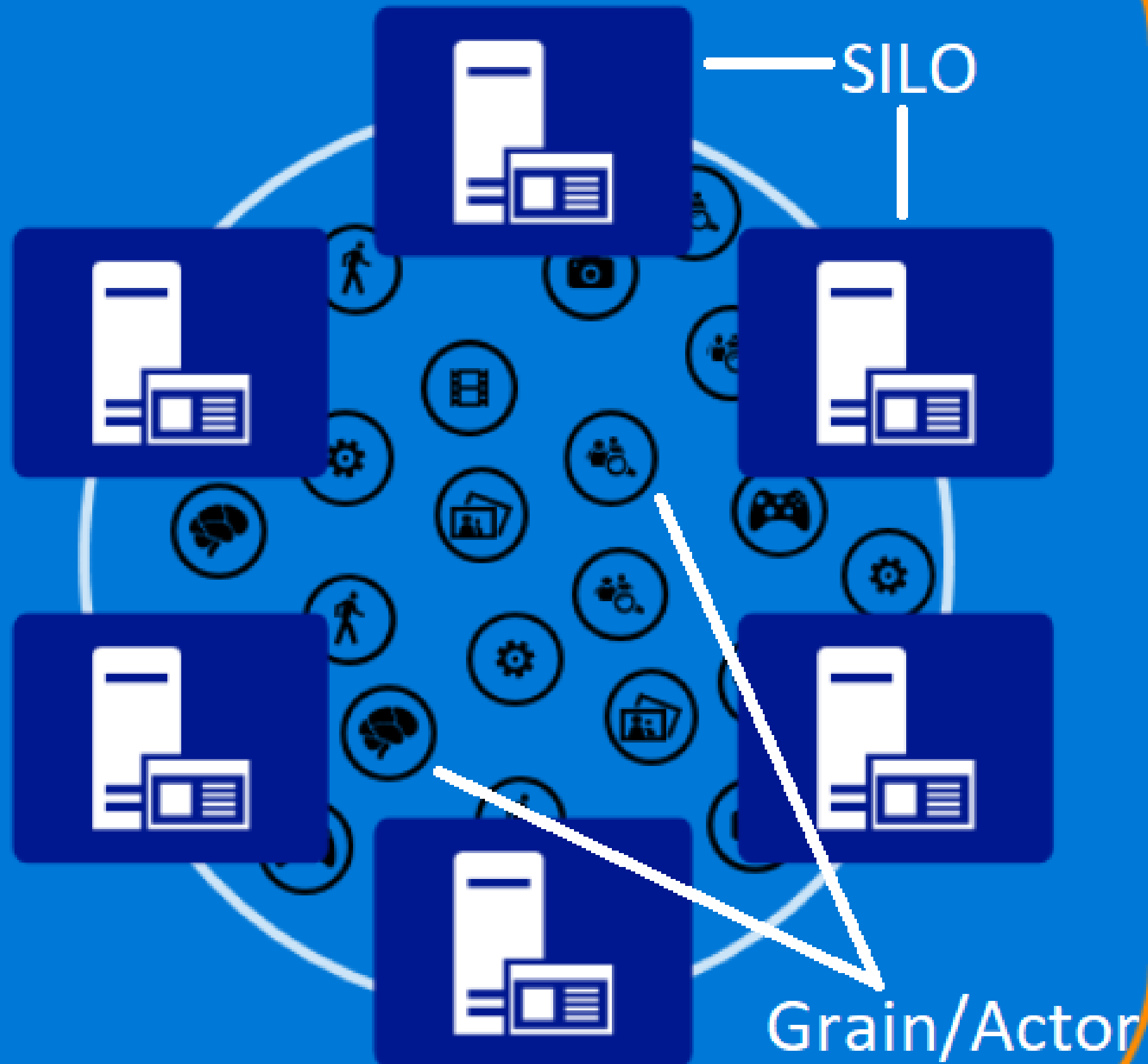
## LIFECYCLE

- Activating
- Active
- Deactivating
- Persisted

## STORAGE

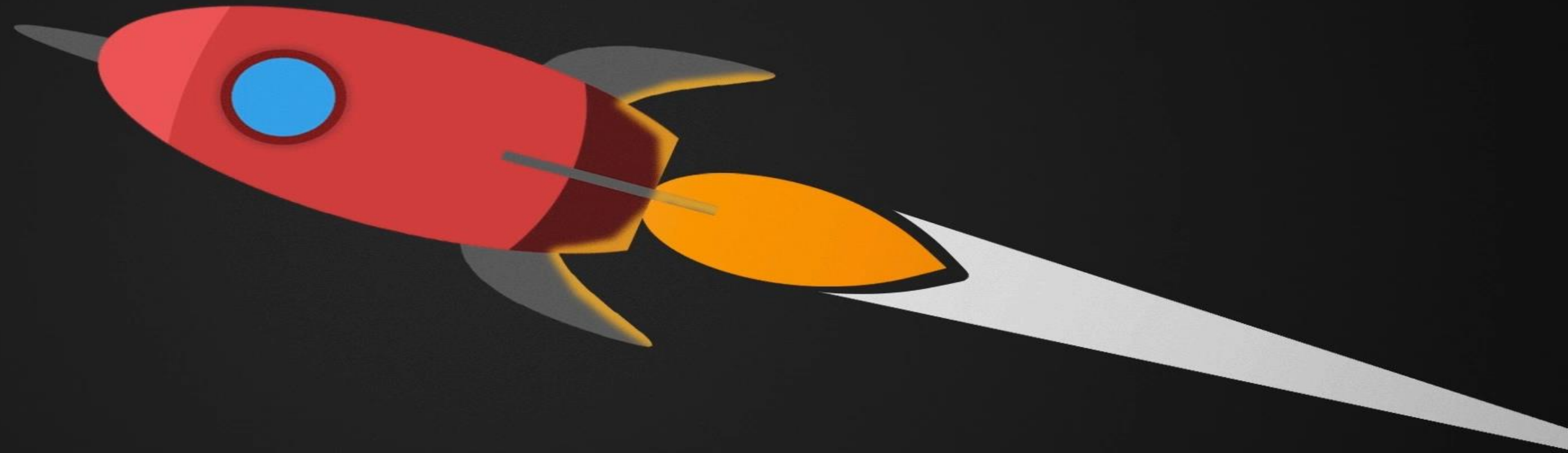
- It can be stored in memory(Volatile)
- Stored in DB for persistence

## Orleans Cluster

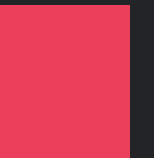


# SILOS

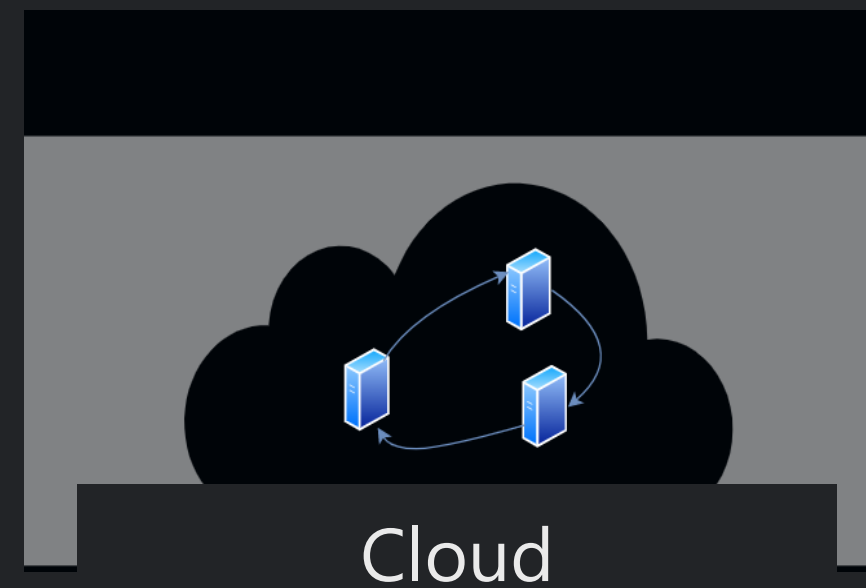
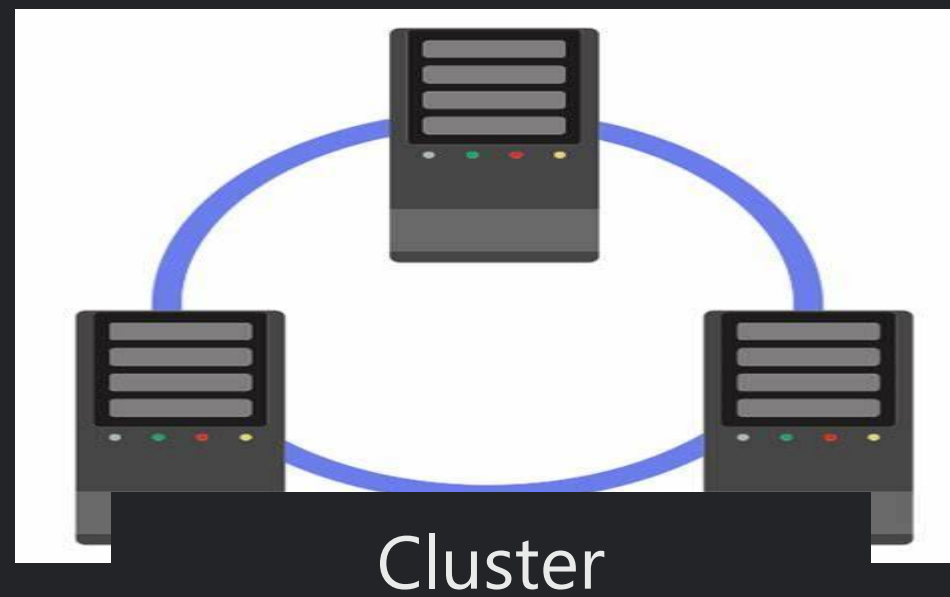
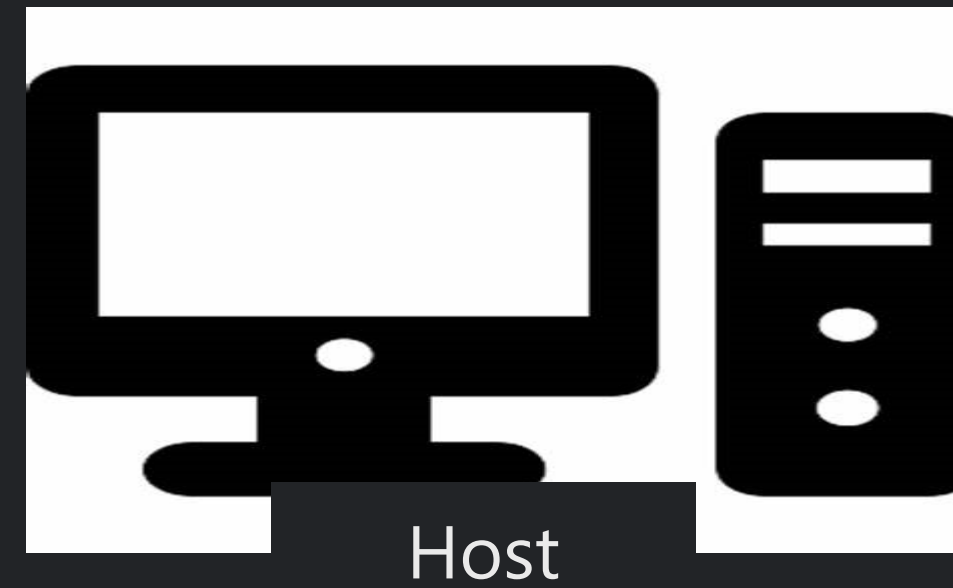
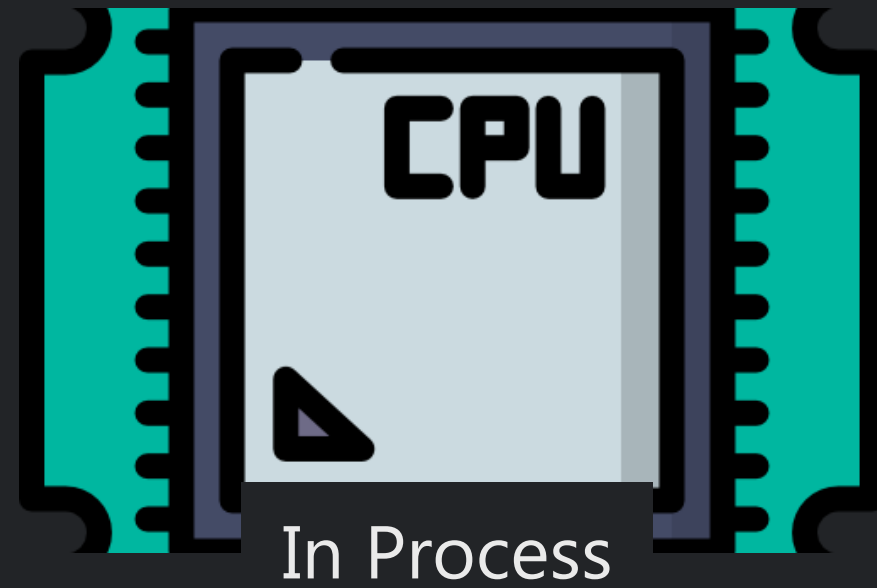
- Its an host for a Grain
- Manages lifecycle of a Grain
- Generally have 1 silo per Node/Container/Machine
- A cluster is used for fault tolerance and Scalability



## 04 DEPLOYMENT



# DEPLOYMENTS







# 05 DASHBOARD

 Overview

 Grains

 Silos

 Reminders

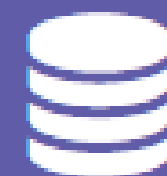
 Log Stream



TOTAL ACTIVATIONS

**471**

[More info](#) 



ACTIVE SILOS

**1**

[More info](#) 



ERROR RATE

**18.53%**



REQ/SEC

**41.88**



AVERAGE RESPON...

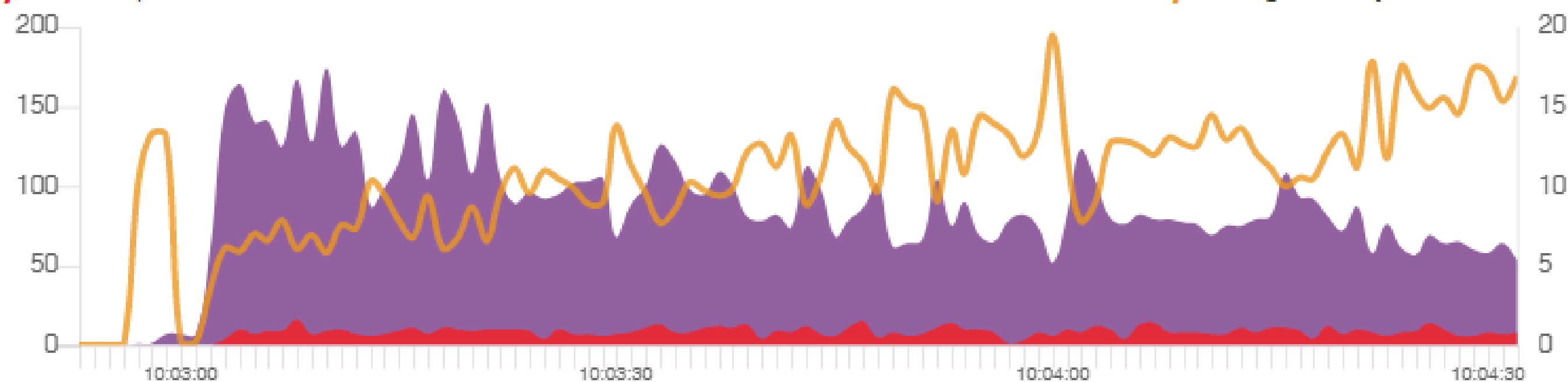
**20.60ms**

## Cluster Profiling

 number of requests per second

 failed requests

 average latency in milliseconds



Methods with Most Calls

Methods with Most  
Exceptions

Methods with Highest  
Latency

LIGHTS  
CAMERA  
ACTION

06 DEMO



# REFERENCES

- FDSA
- FSADFSA



# THANK YOU FOR WATCHING!

ANY QUESTIONS?

<https://linktr.ee/praveenraghuvanshi>

