Microservices

In the beginning there was Vert.x

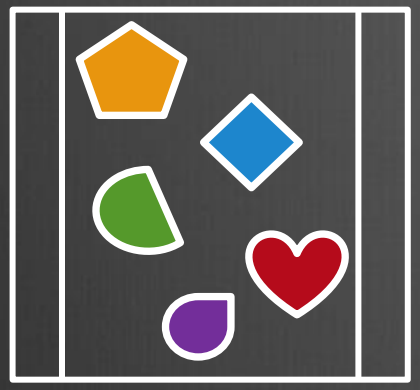
Braz, Anderson

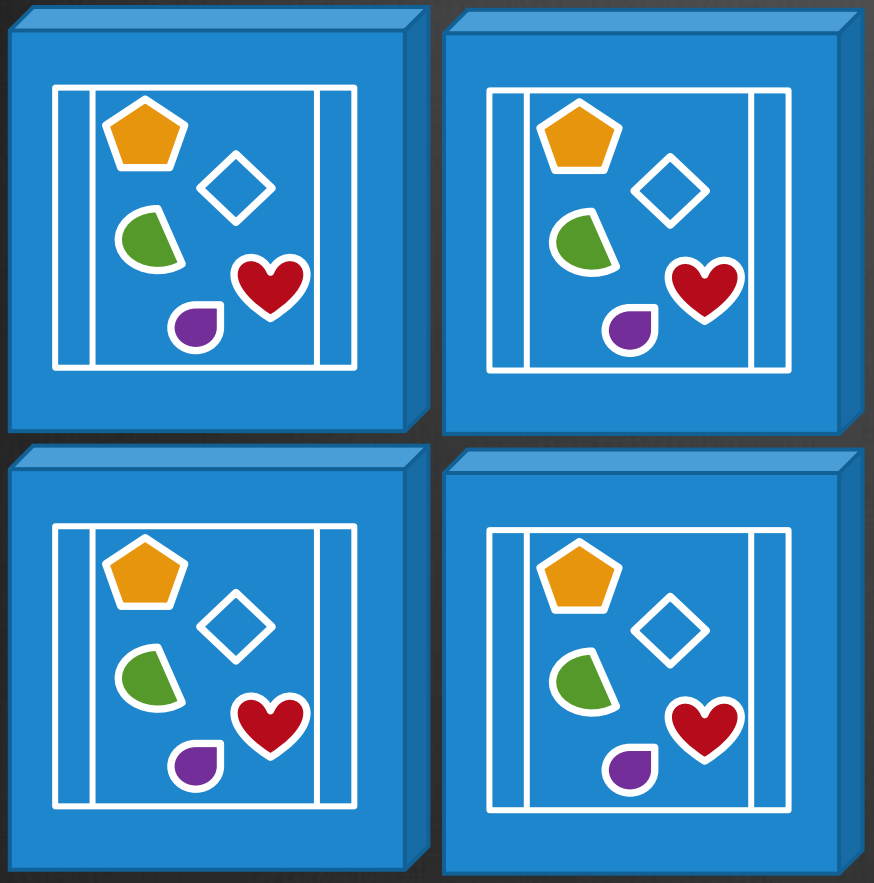
Software Engineer

Java Man since 2000, postgradute, open source contributor, speaker and training consultant

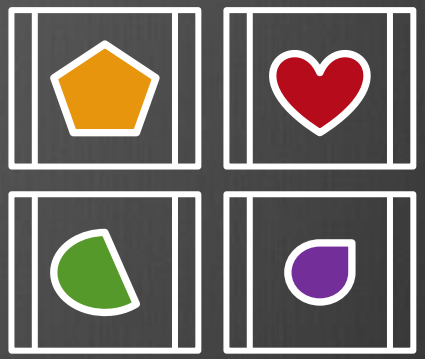
mrbrazjava

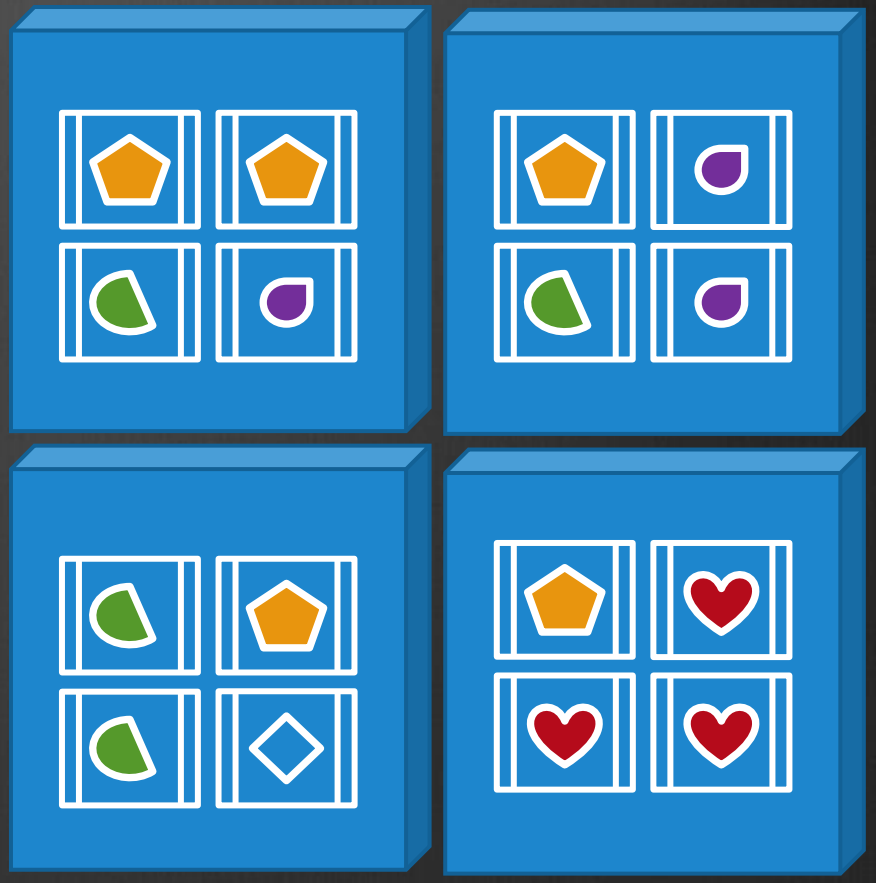
Monolith

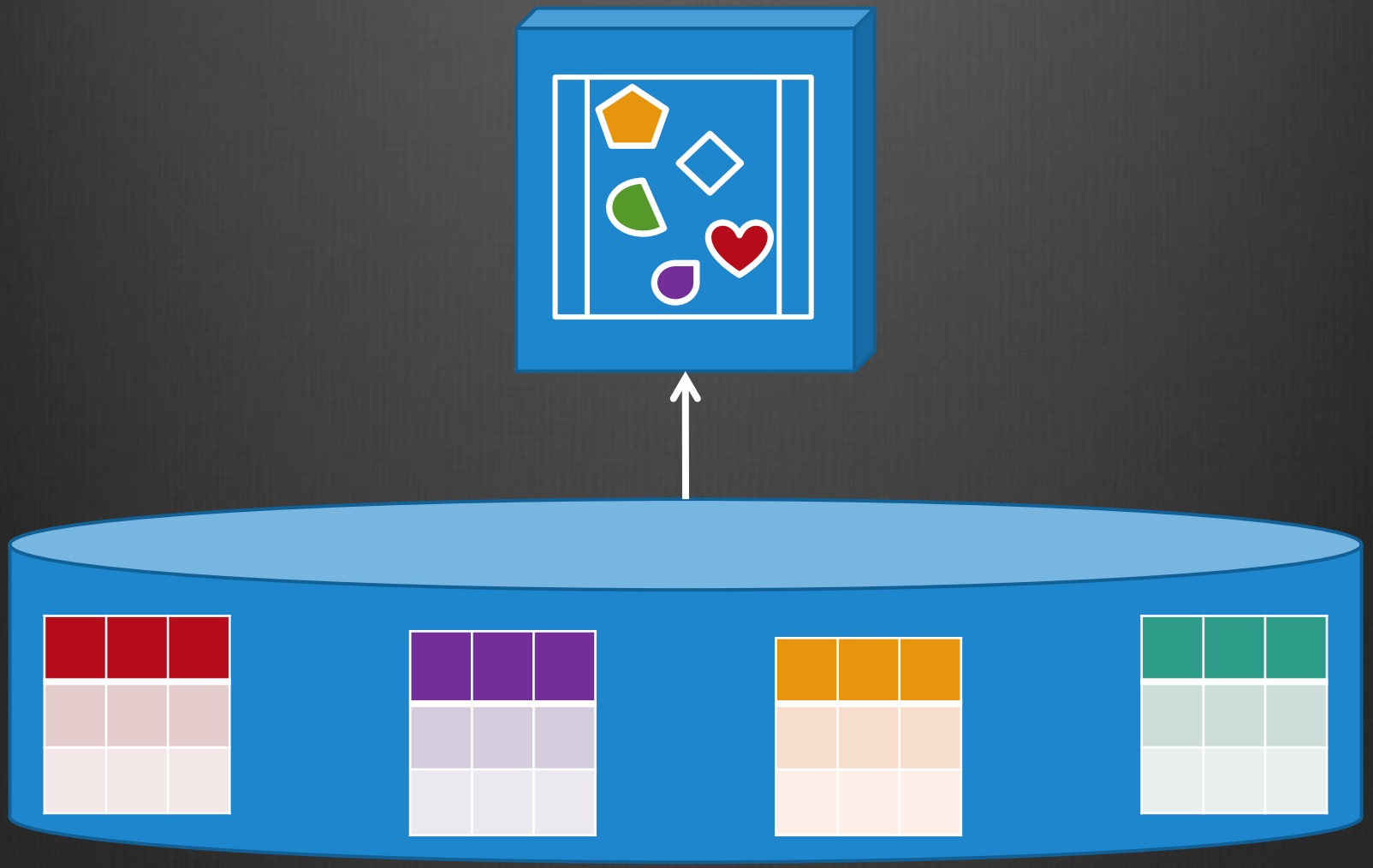


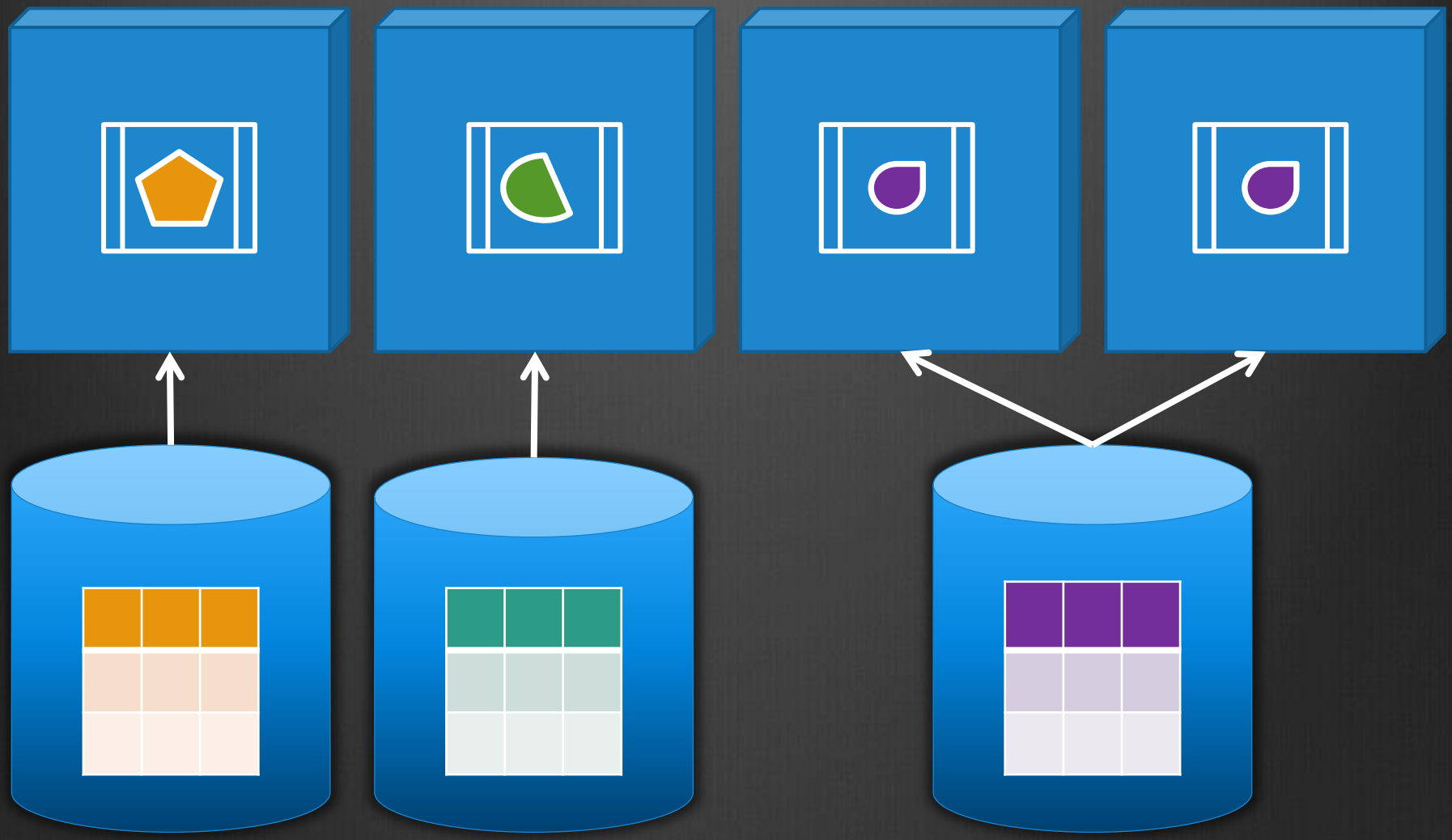


Microservices

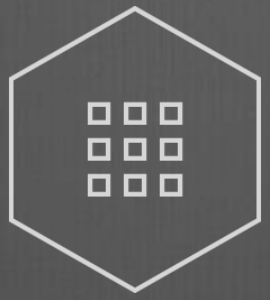




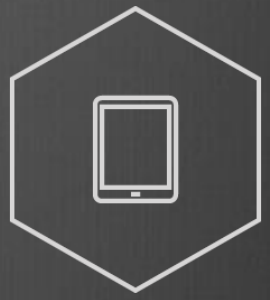




Cloud
Environment



Multicore
Architecture



Mobile
Devices

Why is responsiveness now more important than ever?

Interactive

Collaborative

Real-time

Resilient

(React to Failure)

Elastic

(React to Load)

Reactive

"Readly responsive to a stimulus"

Responsive

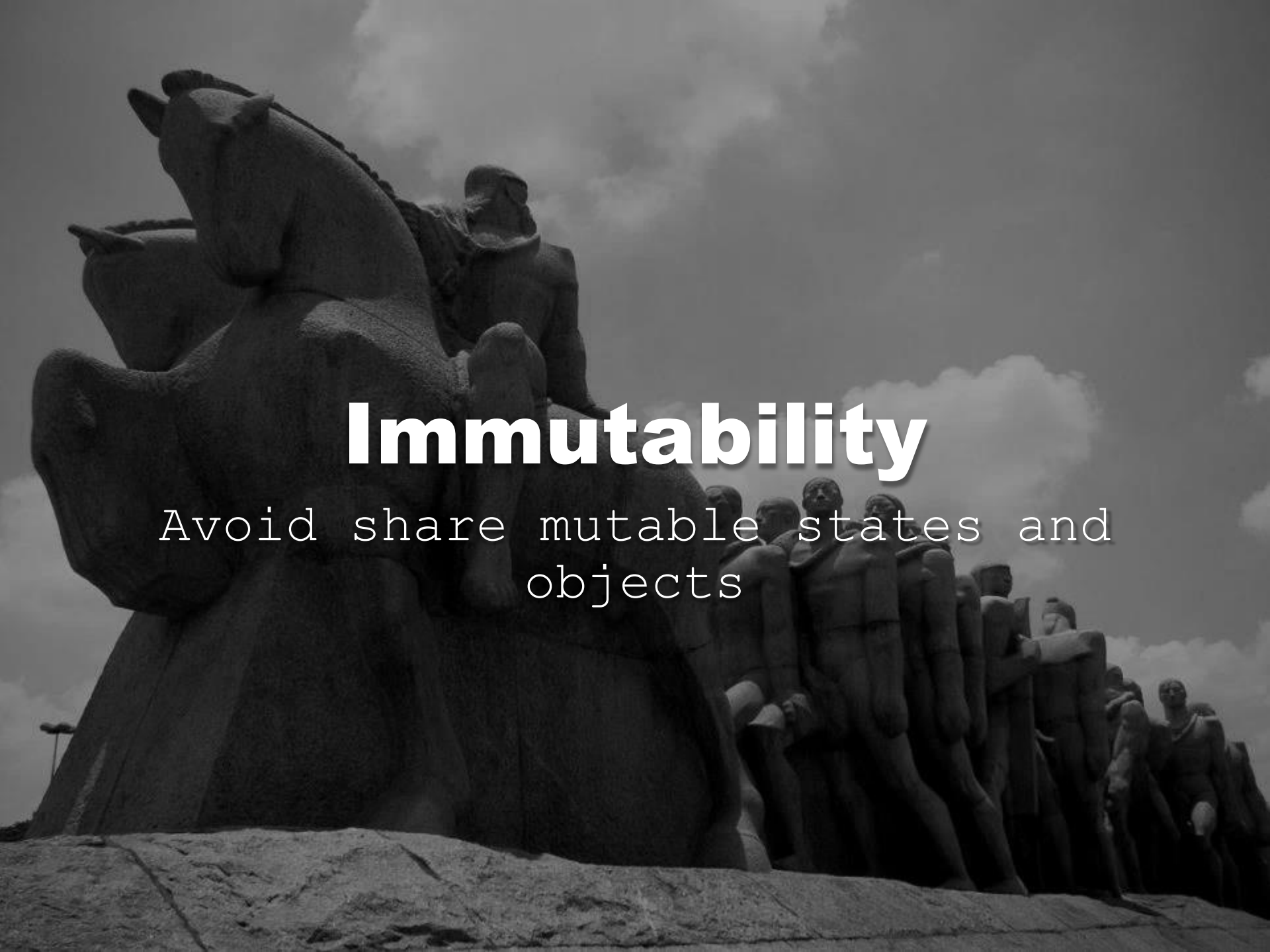
(React to Users)

Message Driven

(React to Events)

Message Driven

"The flow of the program is determined by events"



Immutability

Avoid share mutable states and objects


Avoid Blocking
Kills Scalability and Performance

Elastic

"Capable of being easily expanded or upgraded on demand"

How do I know if I have a performance problem?

How do I know if I have a scalability problem?


The network is
inherently unreliable

Embrace the network

"Be of the web, not behind the web"

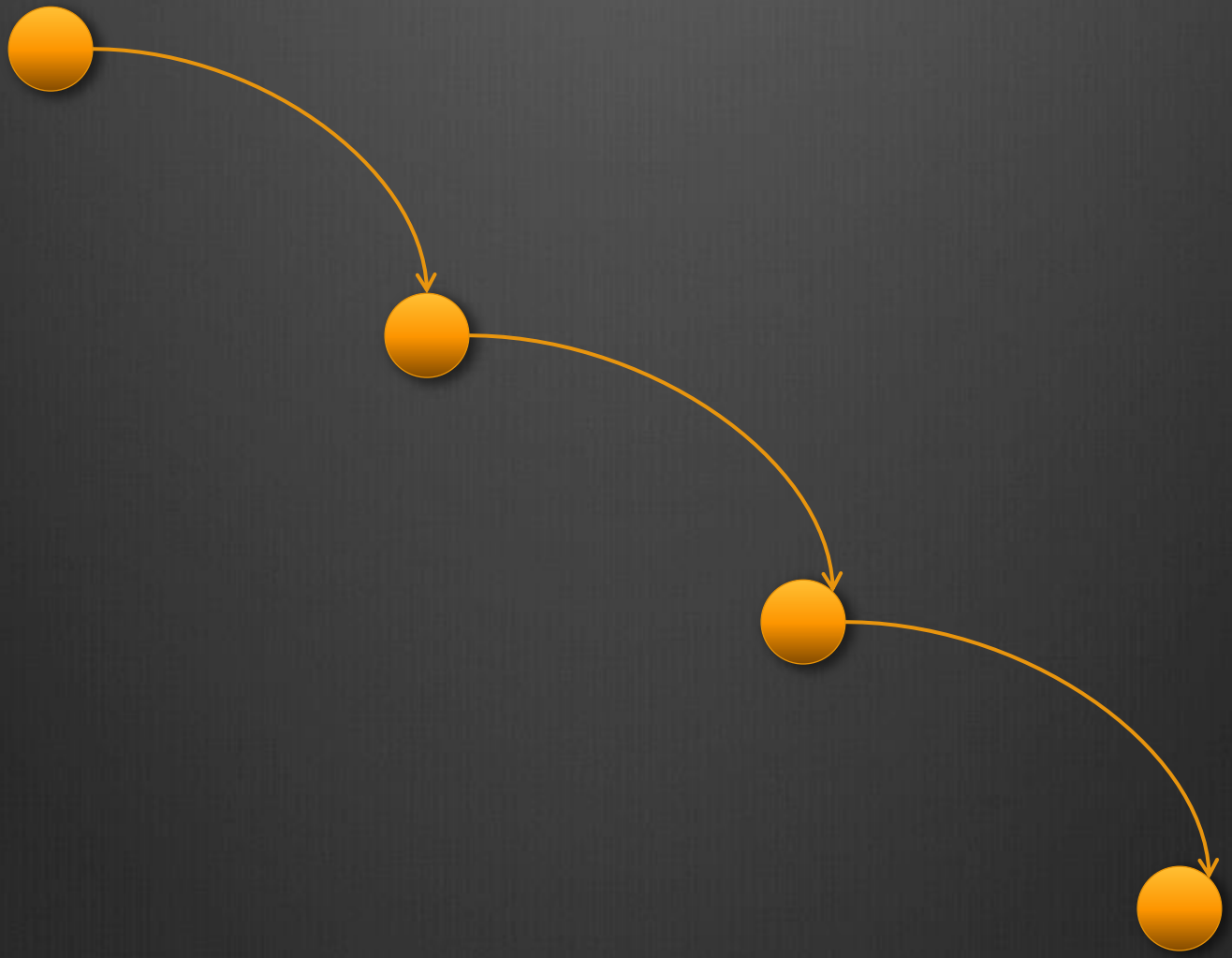
Location Transparency

"It is not about ESB stupid"

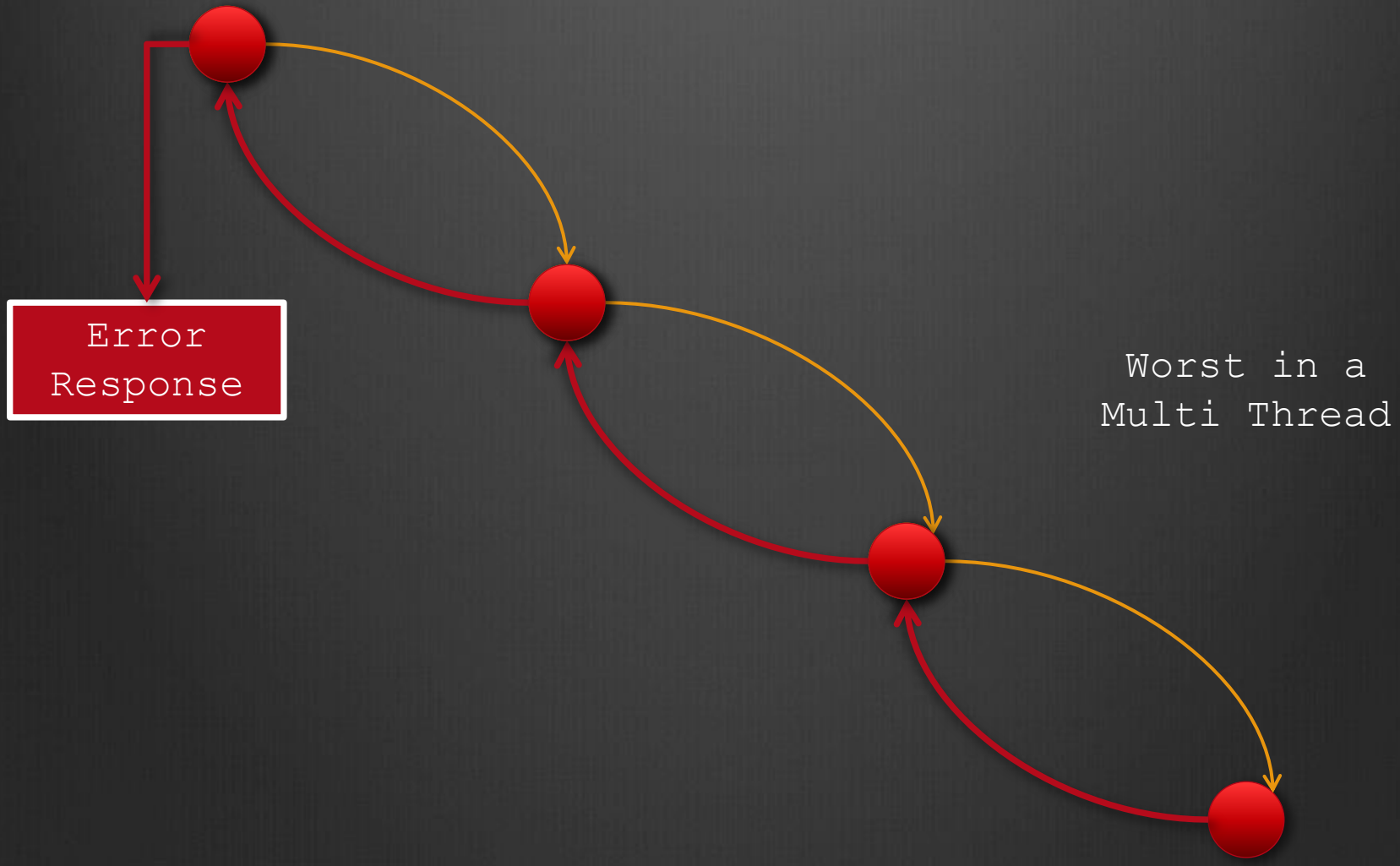
Resilience

"The capacity to recover quickly from difficulties"

Old school way



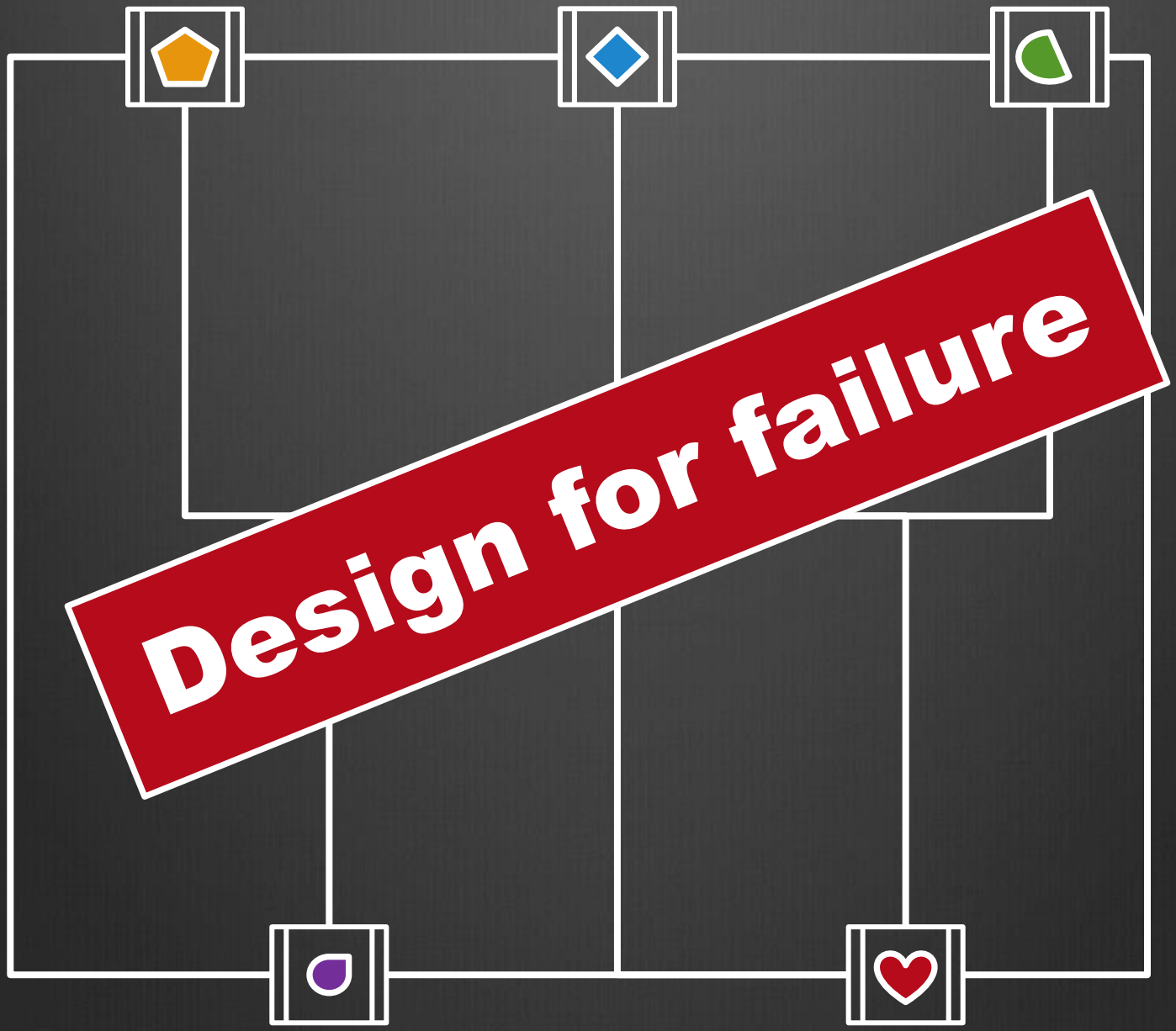
Old school way



Worst in a
Multi Thread

Old school way

Worst in a Multi Thread


Design for failure

Avoid
Cascading

Manage
locally

Doing Better

Failure is
an Event

Isolate
the failure

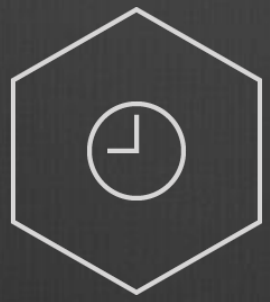
Responsive

"Quick to respond or react appropriately"

Better Throughput

Asynchronous Events

Loosely coupled architecture



Lower Latency

Reactive

Polyglot

Vert.x

Lightweight

Middleware

Event Loop

Scalable

Verticles

Non-Blocking

Simple

Worker Loop

Non-concurrent

Worker

Blocking

Verticle

Pub/Sub

Handlers

Event Bus

Point 2 Point

Addressing

Server and Client

HTTP/S and TCP

File System

Both Blocking and Non-blocking

Shared Data

Local and Distributed using Hazelcast

High Availability

Cluster Manager using Hazelcast with automatic failover

Rich Ecosystem

Asynchronous
JDBC, Mongo, Redis

Autentication &
Ãuthorization

Web &
Microservices

Unit Test

Openshift &
Docker

Metrics
(Dropwizard)

Braz, Anderson

Software Engineer

Java Man since 2000, postgradute, open source contributor, speaker and training consultant

mrbrazjava