MONOLITHS MUST DIE!

A VERT.X TALE ON REACTIVE MICROSERVICES

by Paulo Lopes / RedHat Principal Software Engineer

@jetdrone

CHAPTER 1

MONOLITHS APPLICATIONS ARE REAL

MSA KILLER APP

- The services are easy to replace
- Services are organized around capabilities
- Services can be polyglot
- Architectures are symmetrical rather than hierarchical

CHAPTER 2

SOLVING THE PROBLEM THE "STANDARD" WAY

APPLICATION CONTAINER

APPLICATION CONTAINER

- Security
- Threading
- Transaction Management
- Load balancing

CONTAINER API

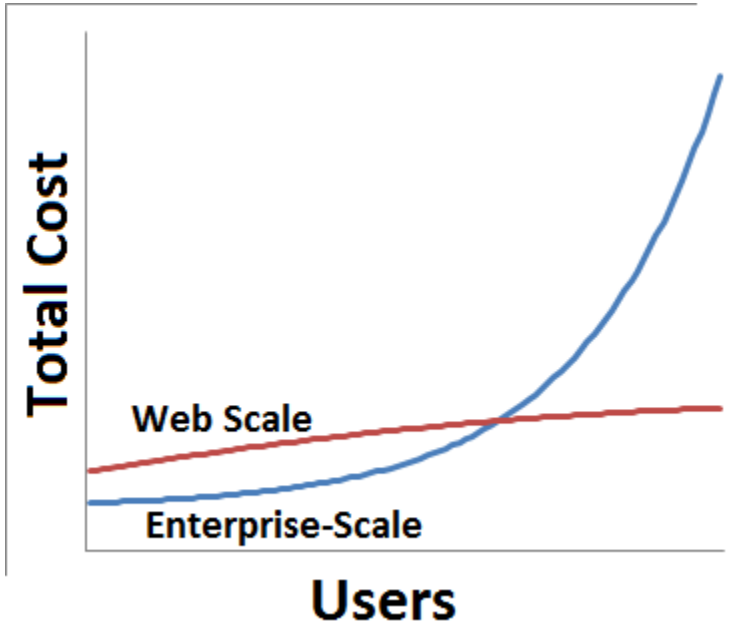
PERFORMANCE

- Blocking
- Thread safety
- Locking
- C10K

SCALE

- Load balancer
- Sticky sessions
- Single point of failure

WHY CAN'T THIS SCALE?



CHAPTER 3

HELLO VERT.X

VERT.X

TOOLKIT

```
<dependency>
    <groupid>io.vertx</groupid>
    <artifactid>core</artifactid>
    <version>3.2.1</version>
</dependency>
```

UNOPINIONATED

```
import io.vertx.core.AbstractVerticle;
public class Server extends AbstractVerticle {
  public void start() {
    vertx.createHttpServer().requestHandler(req -> {
      req.response()
        .putHeader("content-type", "text/plain")
        .end("Hello from Vert.x!");
    }).listen(8080);
  }
}
```

REACTIVE

```
conn.query("SELECT * from EMP", res -> {
  if (res.succeeded()) {
    ResultSet resultSet = res.result();
  } else {
    // Failed!
  }
});
```

POLYGLOT

```
vertx.createHttpServer()
  .requestHandler(function (req) {
    req.response()
      .putHeader("content-type", "text/plain")
      .end("Hello from JS Vert.x!")
}).listen(8080)
```

DISTRIBUTED

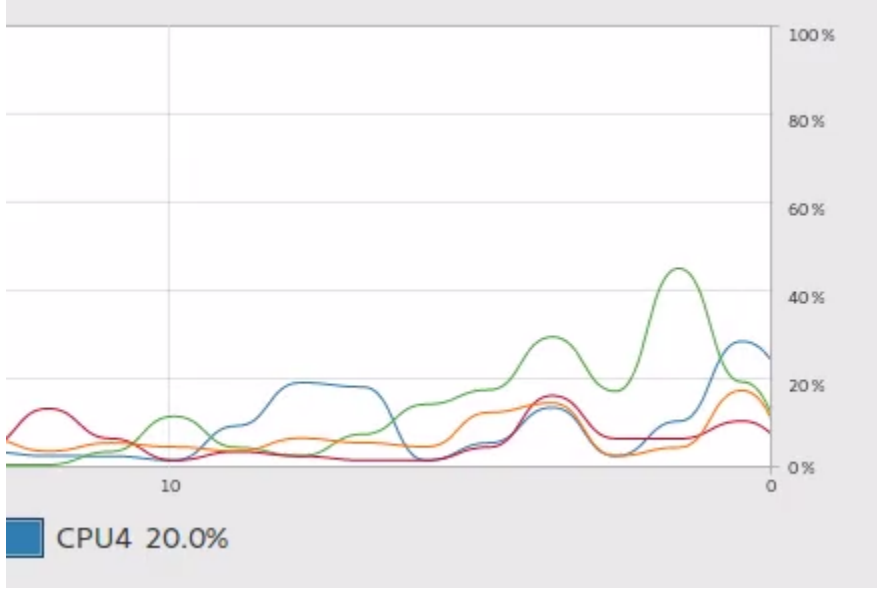
```
EventBus eb = vertx.eventBus();

eb.consumer("news.sport", message -> {
  println("I have received a message: " + message.body());
});
```

CHAPTER 4

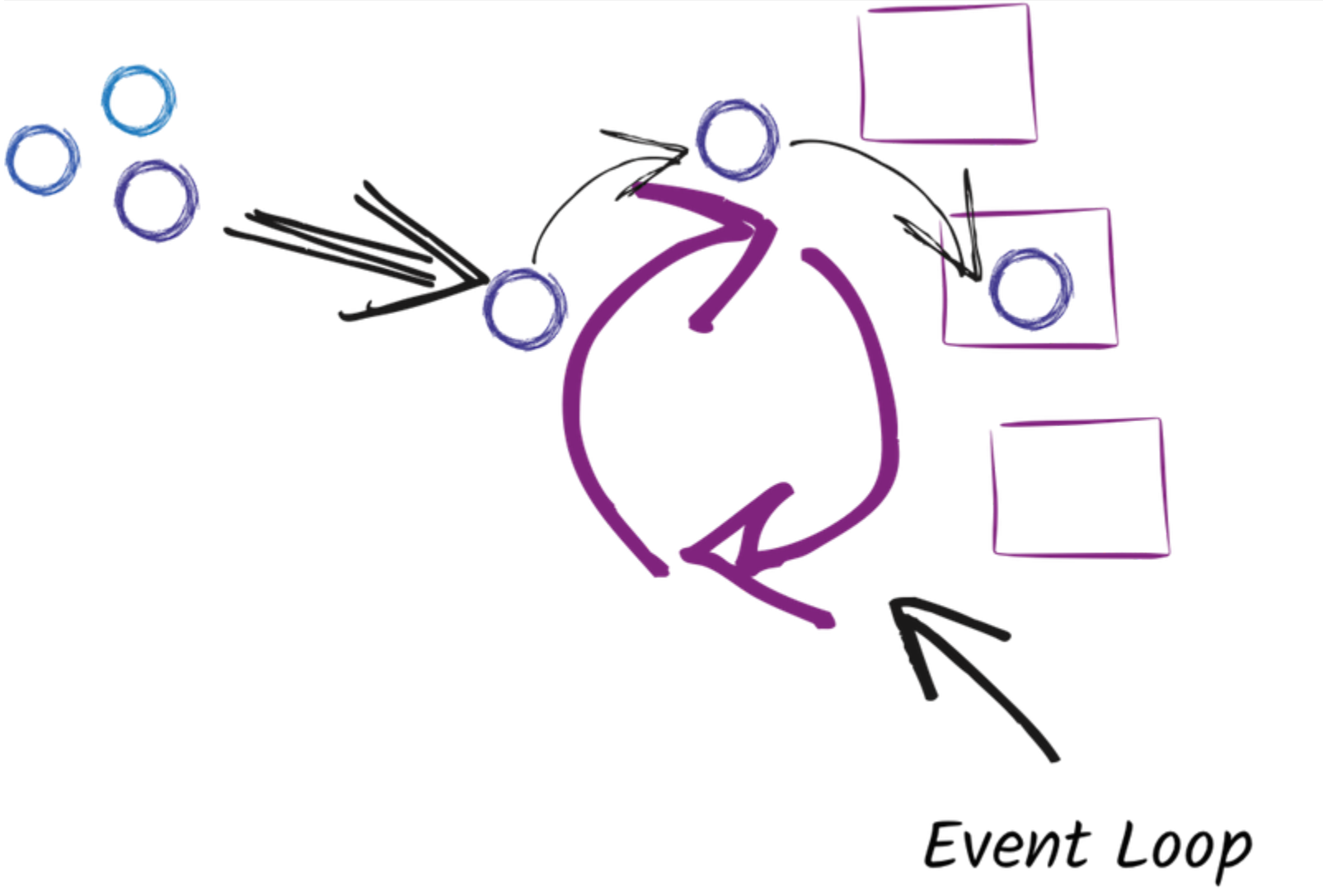
VERT.X SAVES THE DAY

VERT.X API

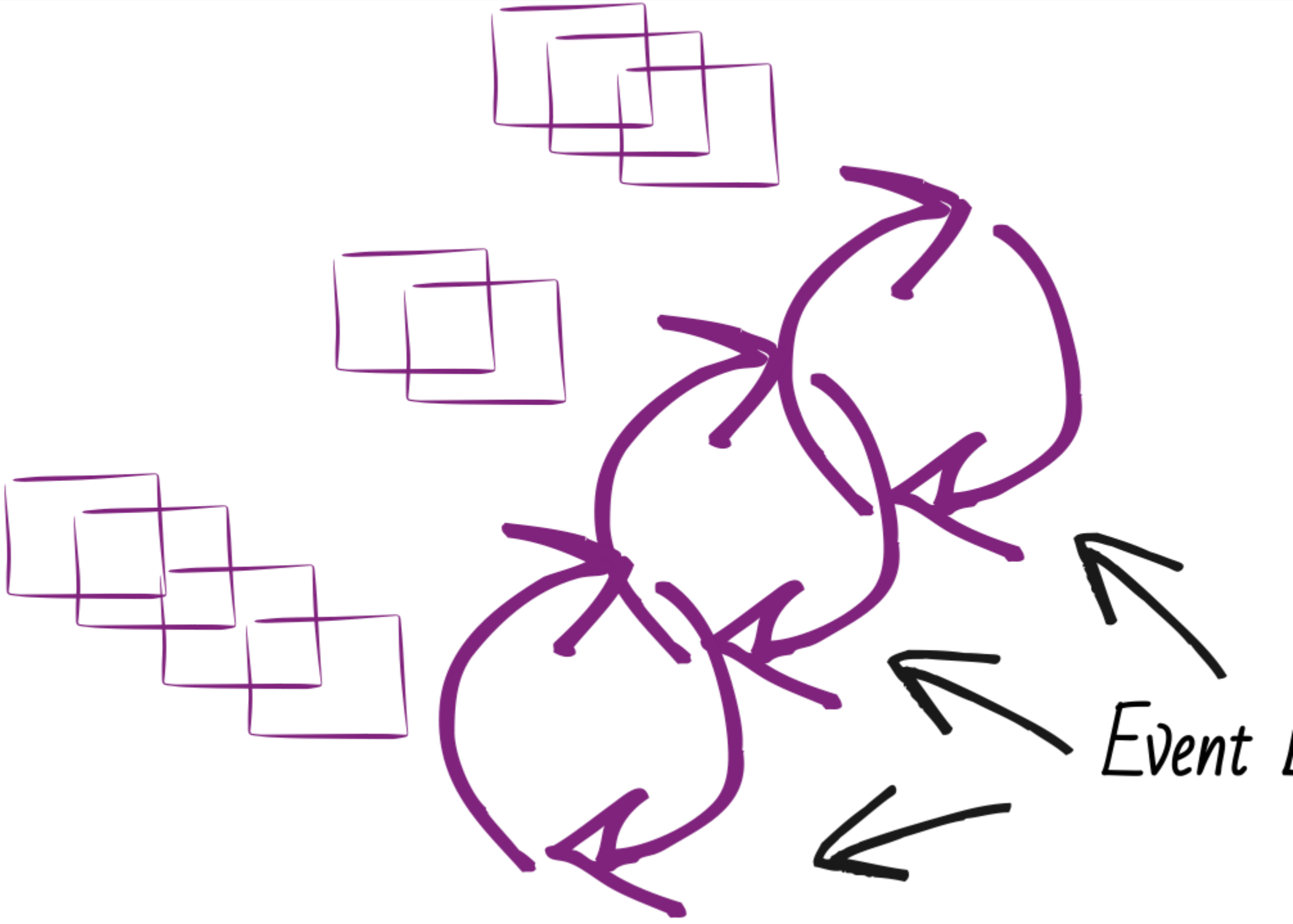
C10K? NO PROBLEM!



HOW IS THIS POSSIBLE?

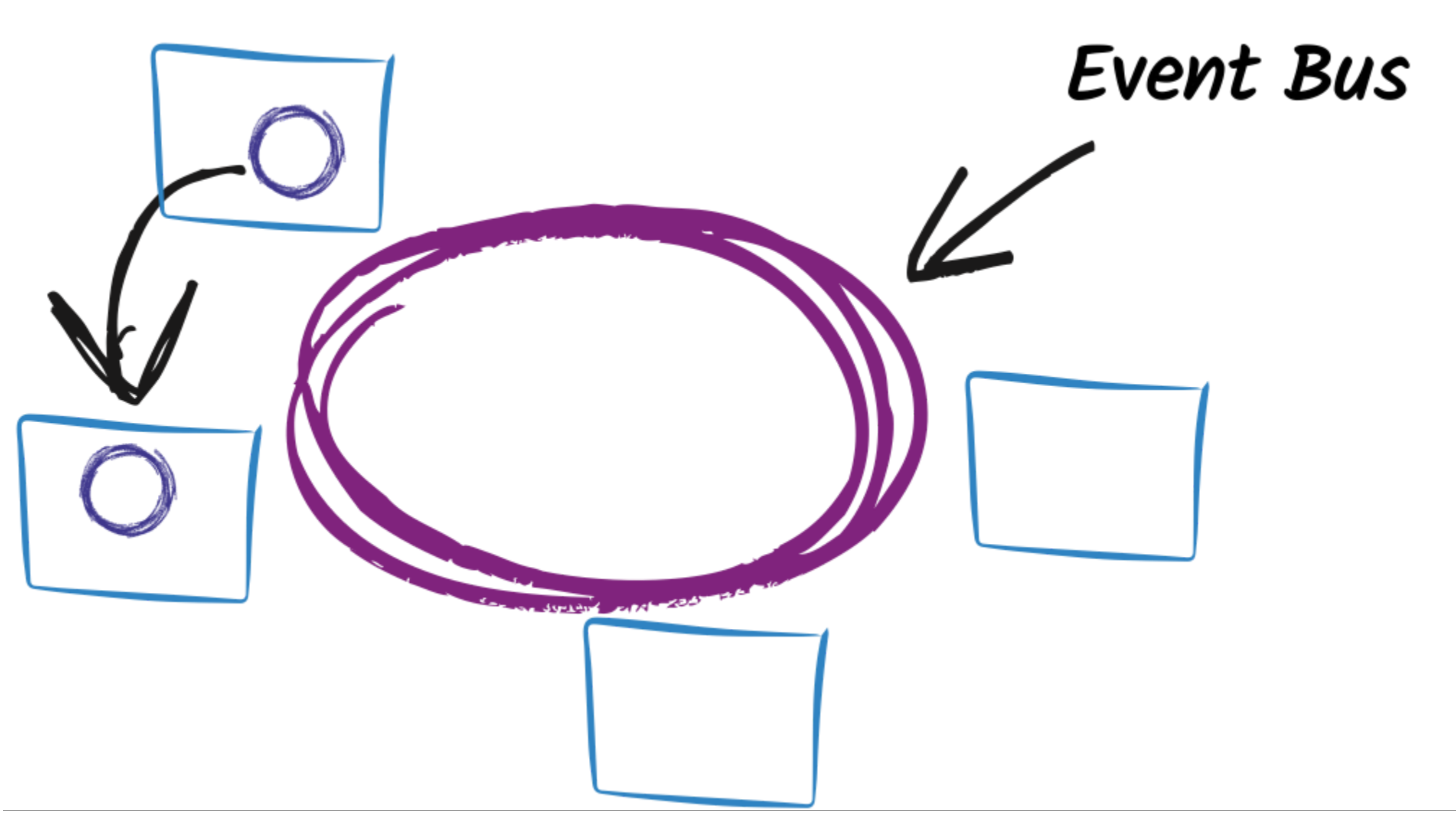


REACTOR PATTERN

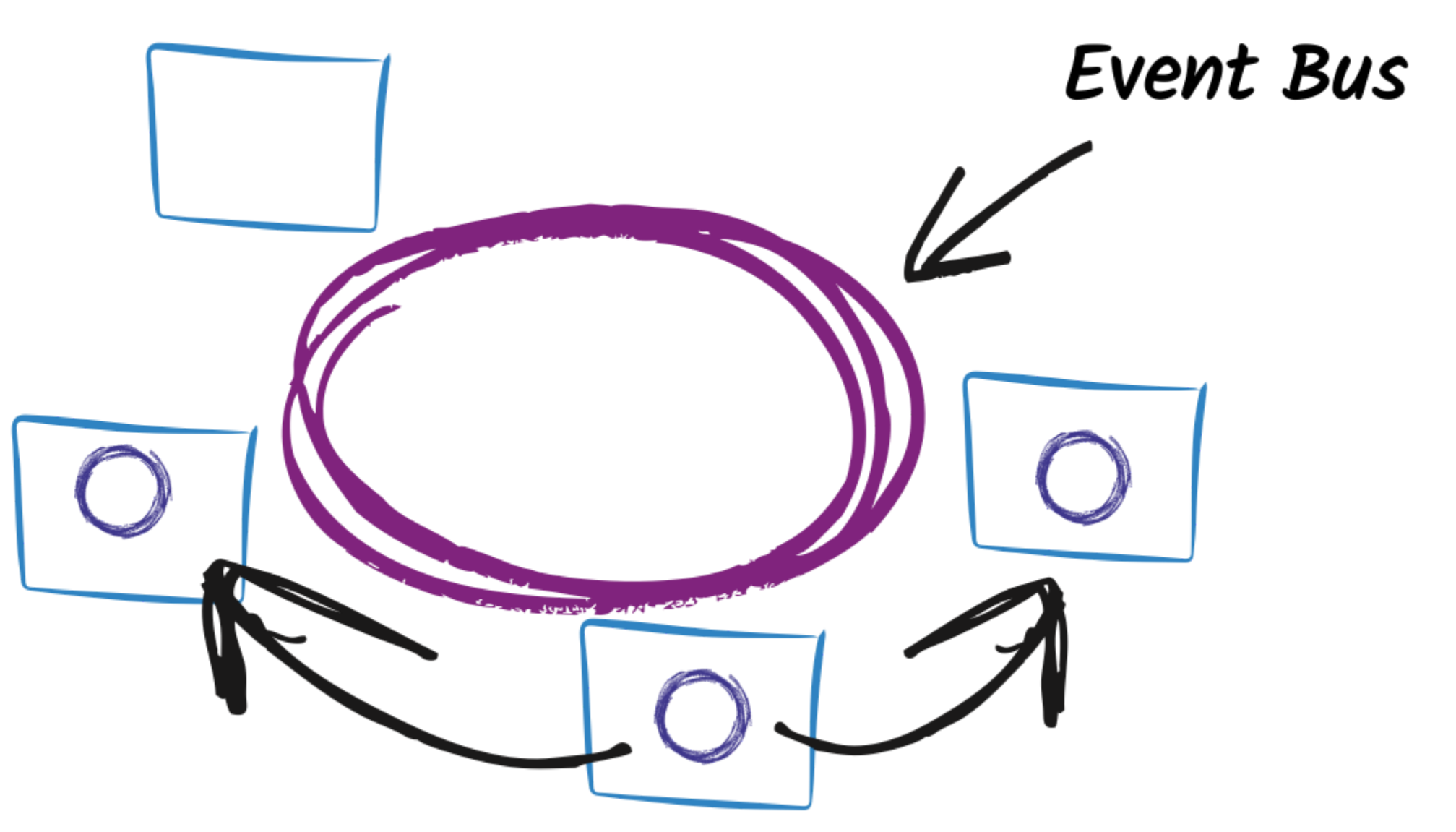

Event

SCALE

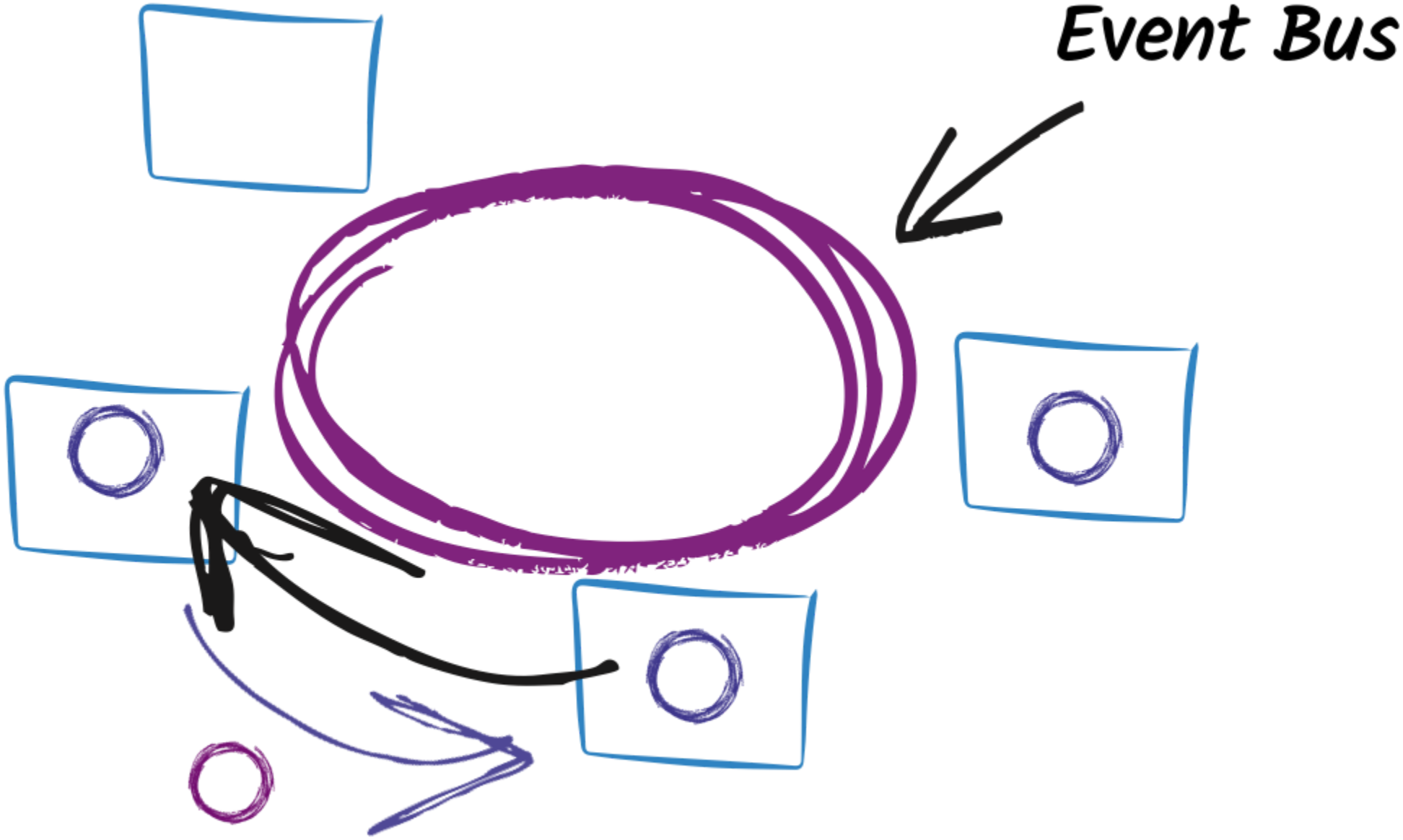
POINT TO POINT



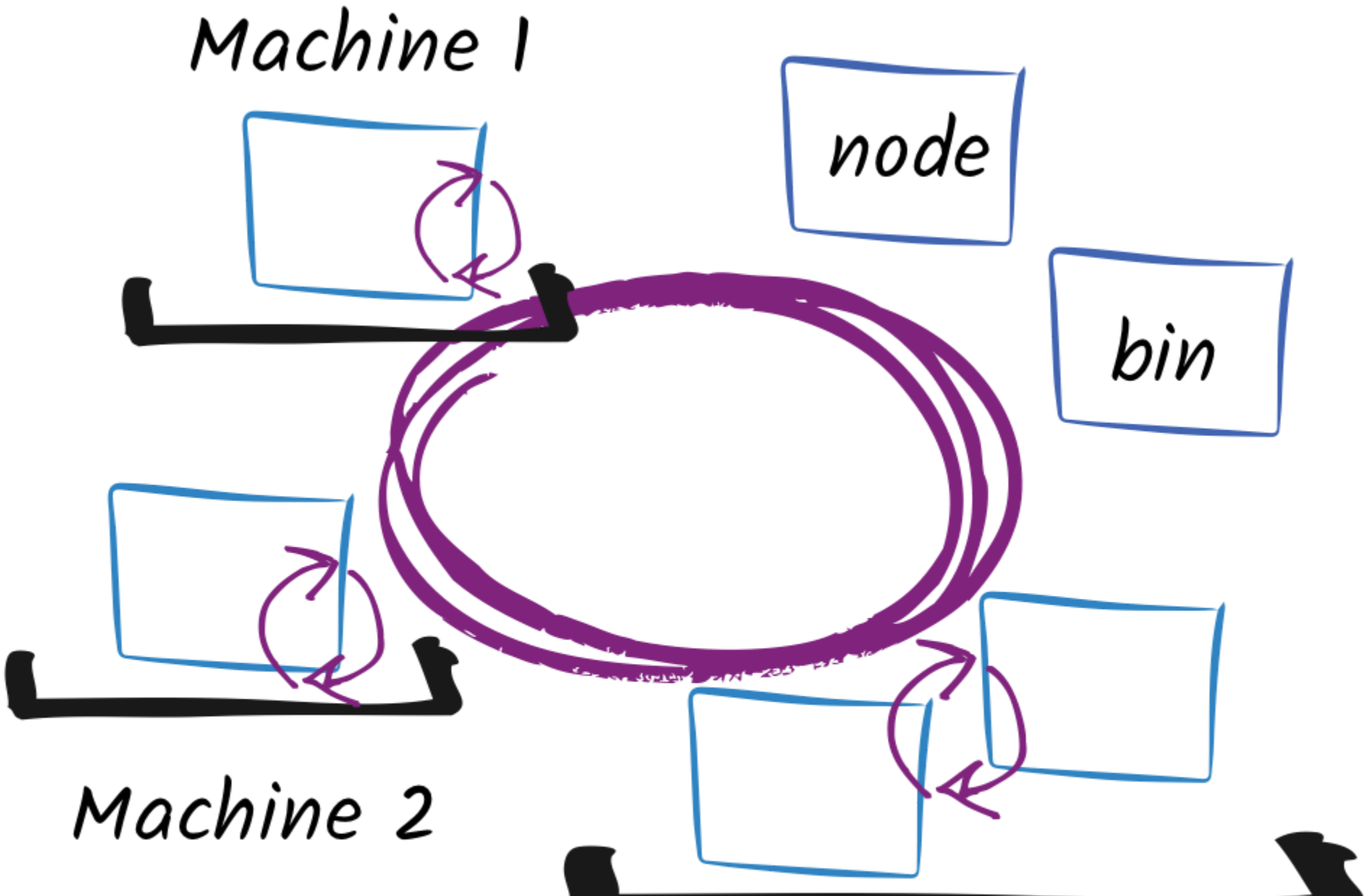
PUBLISH SUBSCRIBE



REQUEST RESPONSE


Event Bus

UNLIMITED HORIZONTAL SCALING



Machine 3

CHAPTER 5

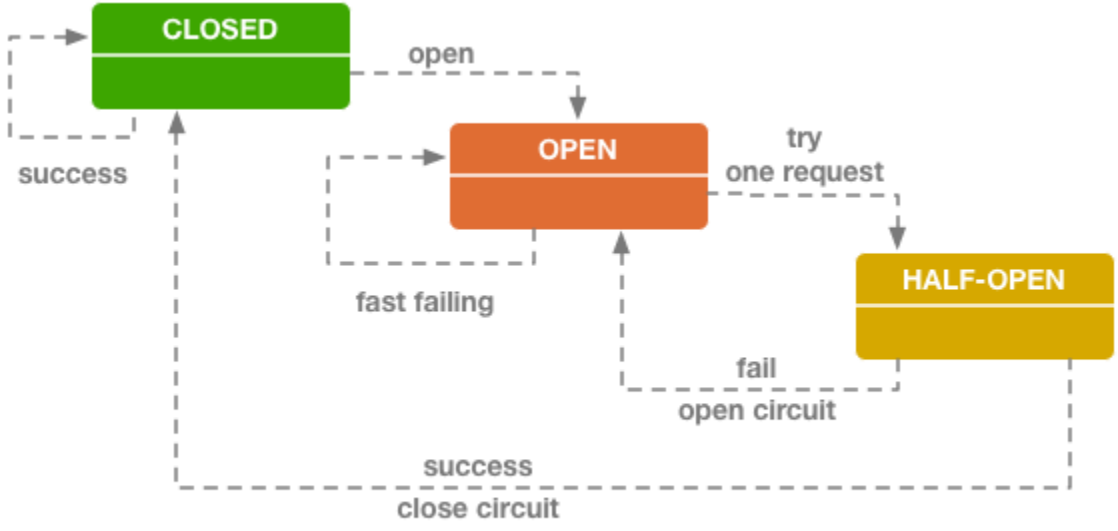
MONOLITHS WATCH OUT!

MONOLITHS MORTAL WEAKNESS

- Message driven problem domain
- HTML5 and Realtime/Websocket problem
- Address services running on browsers
- Flexible actor model
- Scale out
- Mixed async and blocking code

NO COMPARISON

NOT DEAD YET

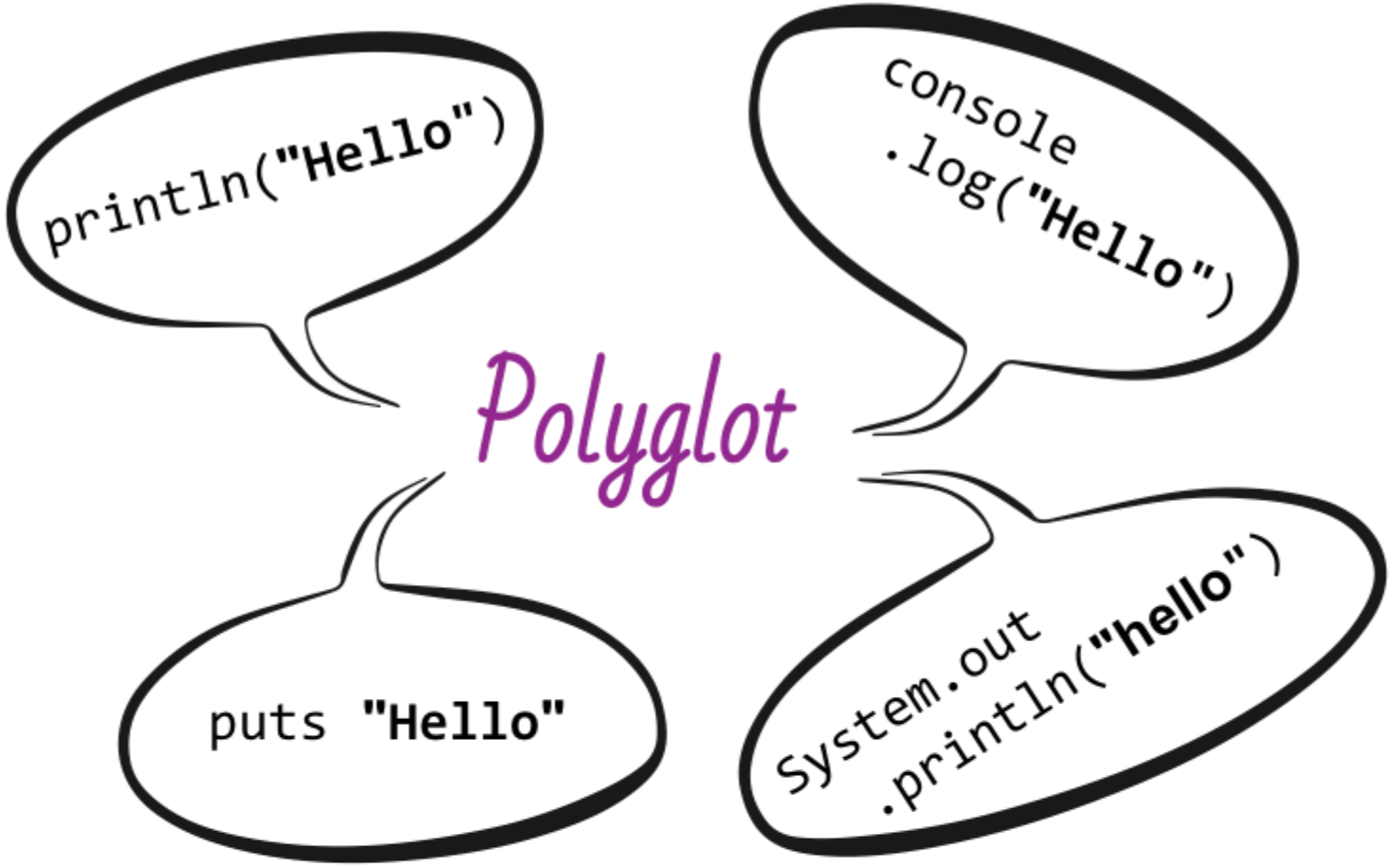


Circuit Breaker State Diagram

CIRCUIT BREAKER

CIRCUIT BREAKER

DON'T FORGET!


Polyglot
println("Hello")
console
.log("Hello")
puts "Hello"
System.out
.println("hello")

Vert

Events

Distributed

THANK YOU!

- http://vertx.io
- https://github.com/eclipse/vert.x
- https://github.com/vert-x3/vertx-examples
- https://groups.google.com/forum/#!forum/vertx
- Vert.x is fun!

THREE

THE ~~SHORTLIST OF~~ ESSENTIAL QUESTIONS

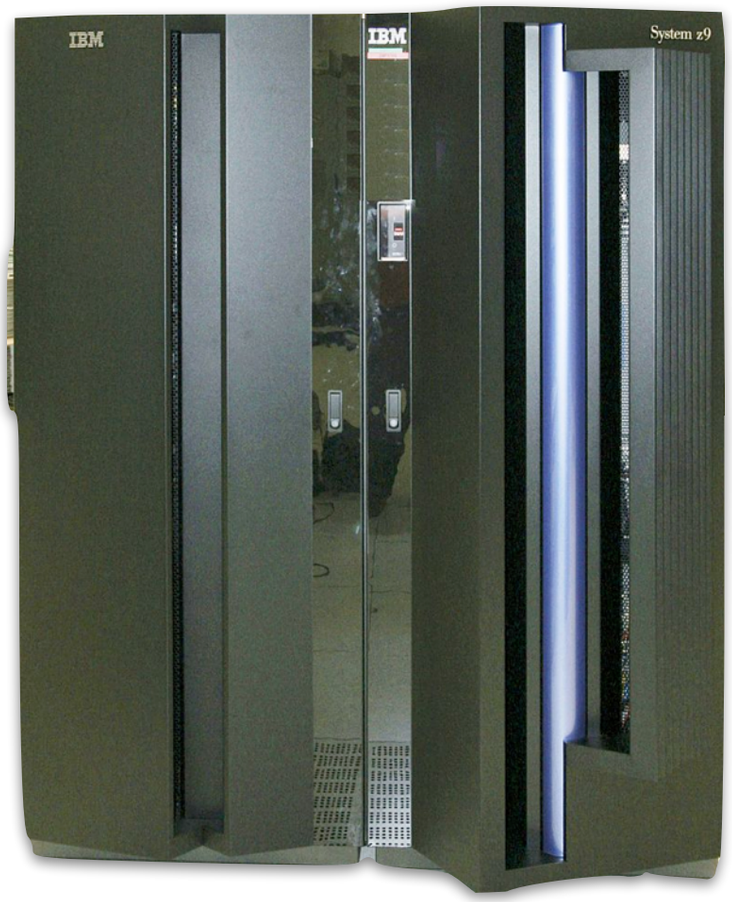
WHY MICROSERVICES MIGHT NOT BE FOR YOU!

@noroute

SURVIVORSHIP BIAS

IS THE LOGICAL ERROR OF CONCENTRATING ON THE PEOPLE OR THINGS THAT "SURVIVED" SOME PROCESS AND INADVERTENTLY OVERLOOKING THOSE THAT DID NOT BECAUSE OF THEIR LACK OF VISIBILITY.

— Wikipedia

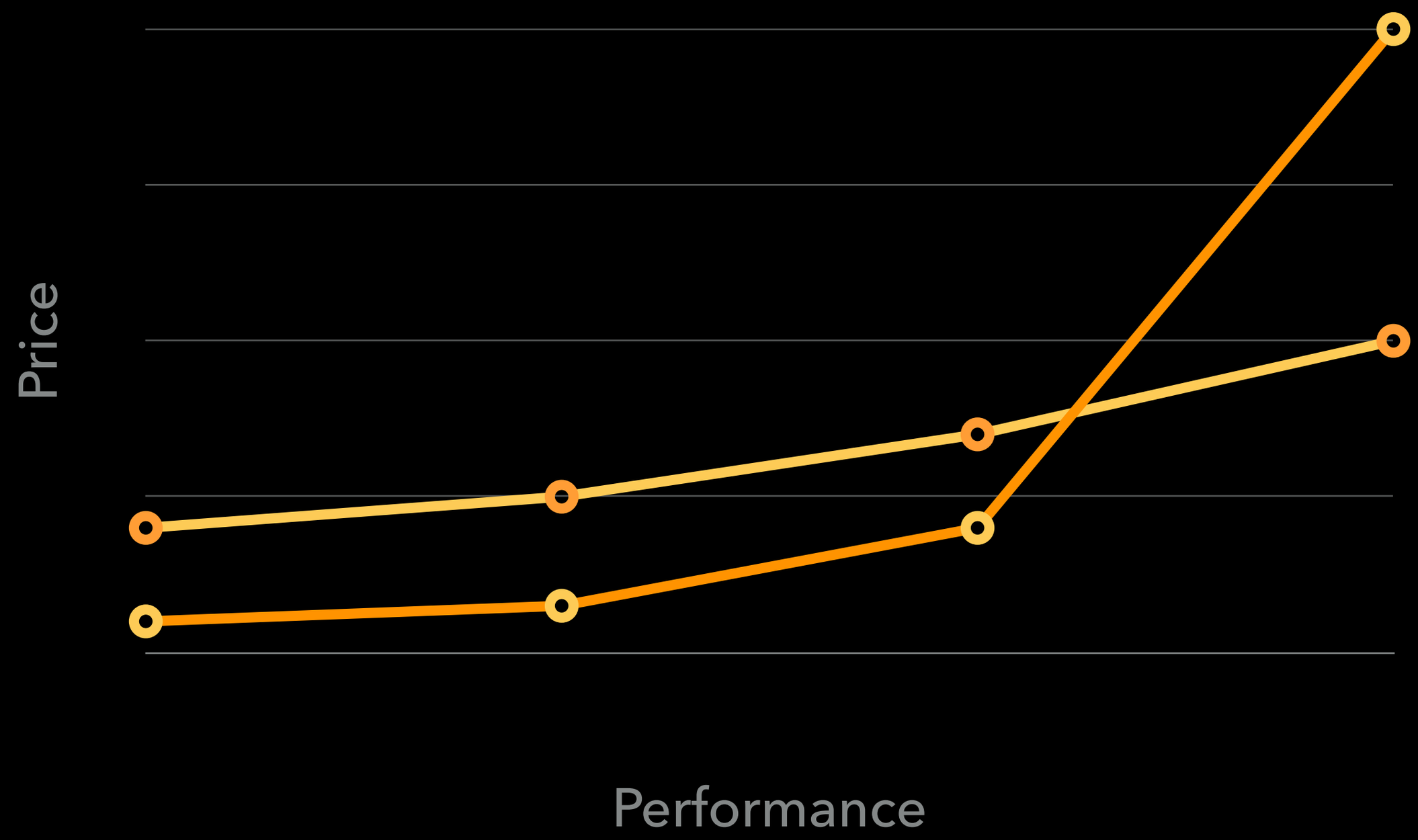


QUESTION #1

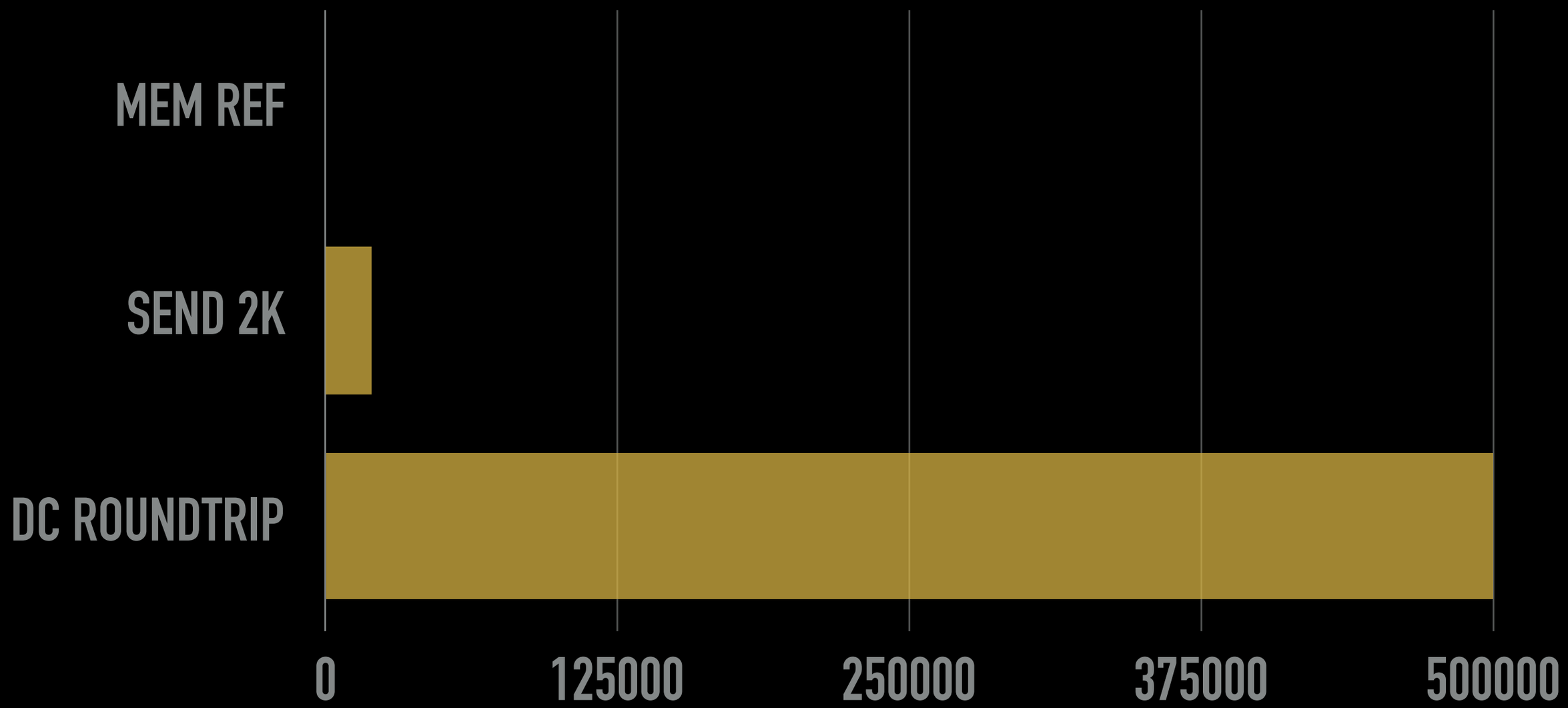
APPLICATION PERFORMANCE

THINK UX!

COST

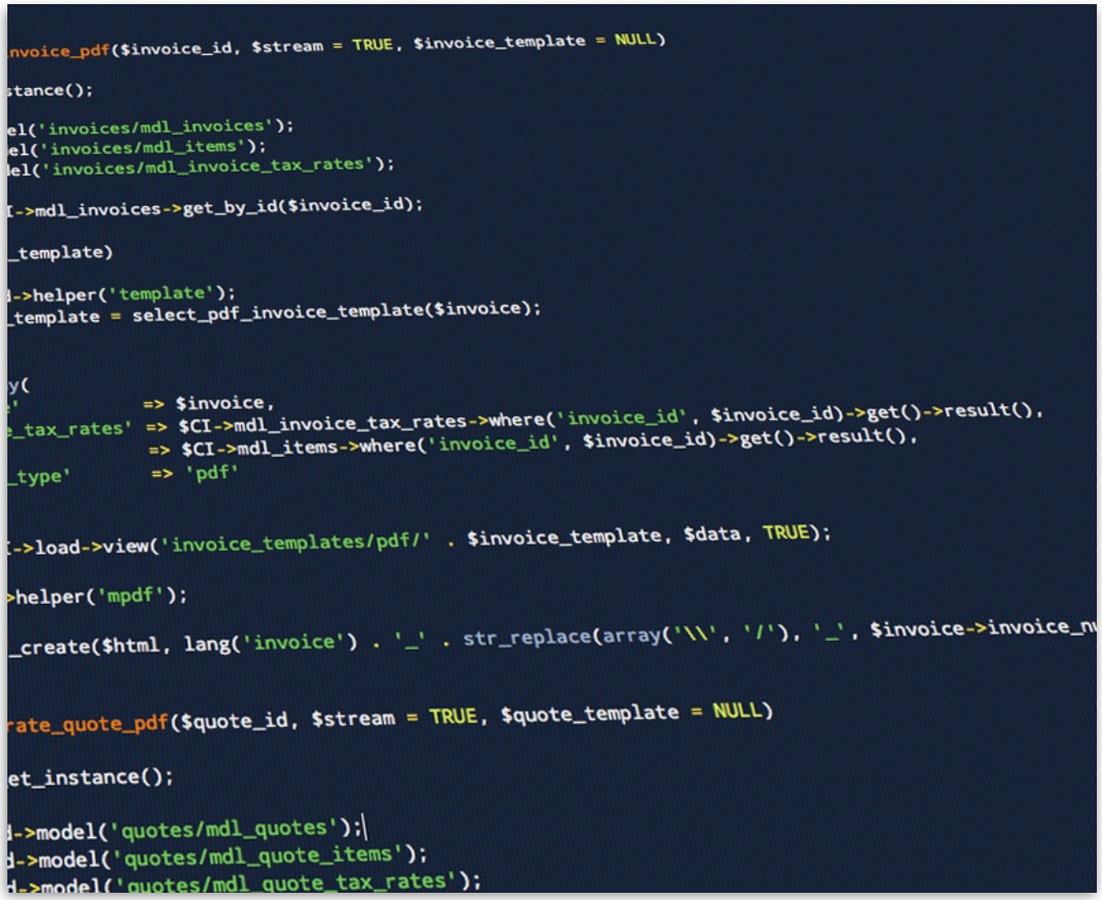


"LATENCY IS ZERO"



SECRET REPORT DECLASSIFIED

MOTEL
OCEANUL
MAREA
ORDULUI
STOCKHOLM
COPENHAGA
HAMBURG
Rostock
Szczecin
Bydgoszcz
Gdańsk
Gdynia
Riga
Tallin
Kaunas
Vilnius
Kaliningrad
Klaipeda
MINSK
Vitebsk
Smolensk
Mogilev
Gomel
Bobruisk
Brasnik
Orlov
Kursk
Tula
Kaluga
Kalinin
Podolsk
Reazan
Cernigov
Bialystok
Helsinki



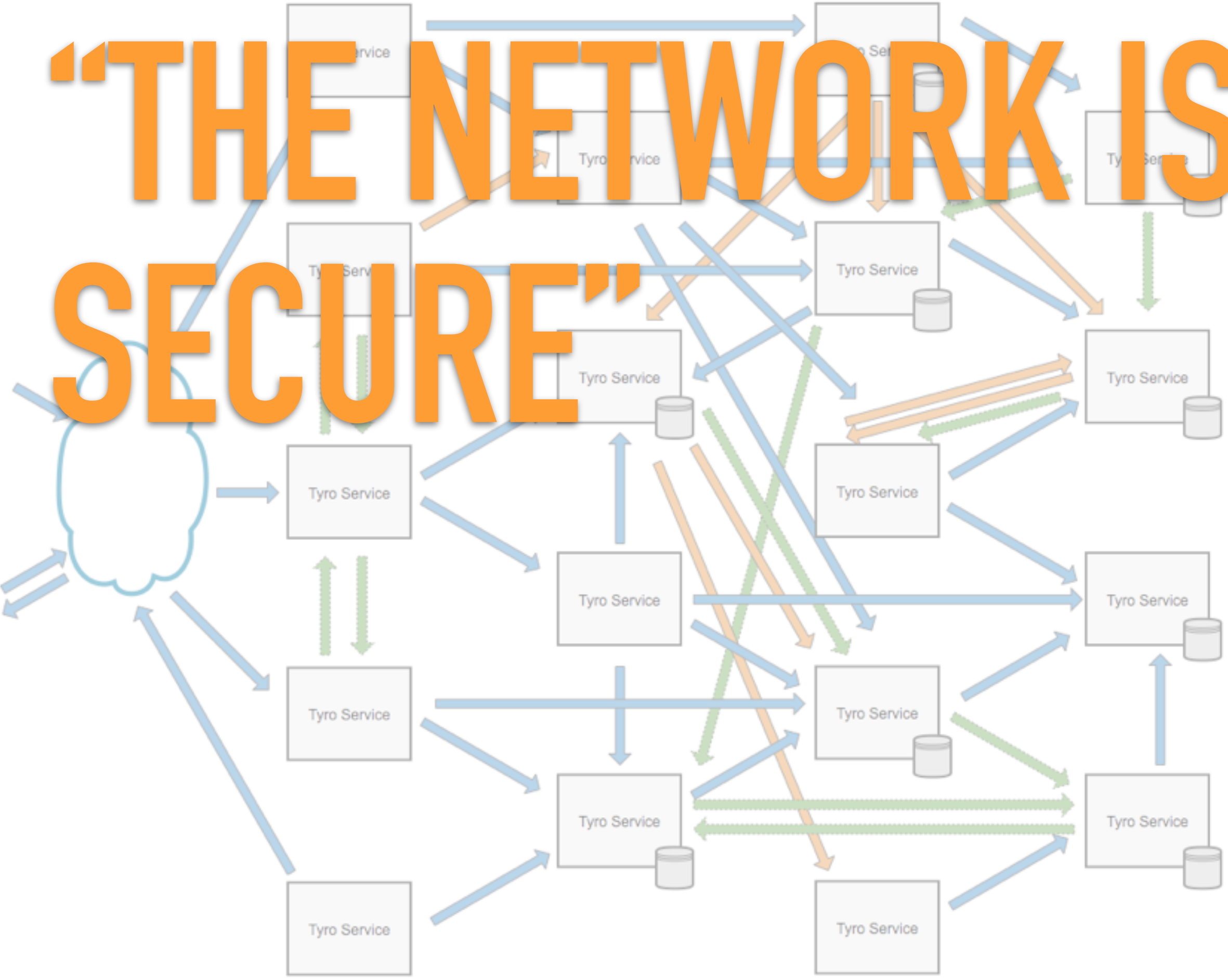
CONSIDERATION #2

COMPLEXITY

TOOLS

PLUMBING

"THE NETWORK IS SECURE"



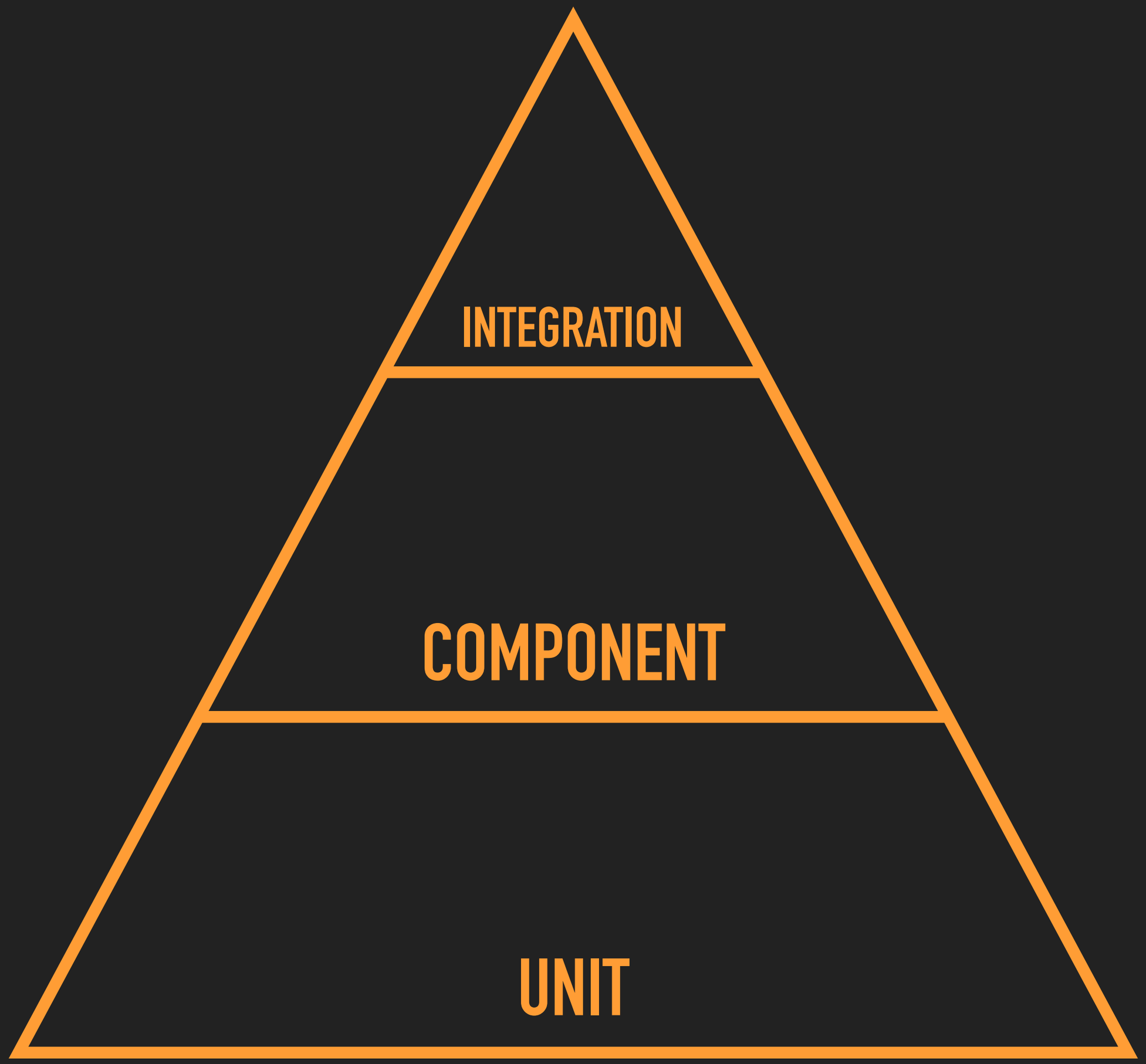
SECRETS MANAGEMENT?

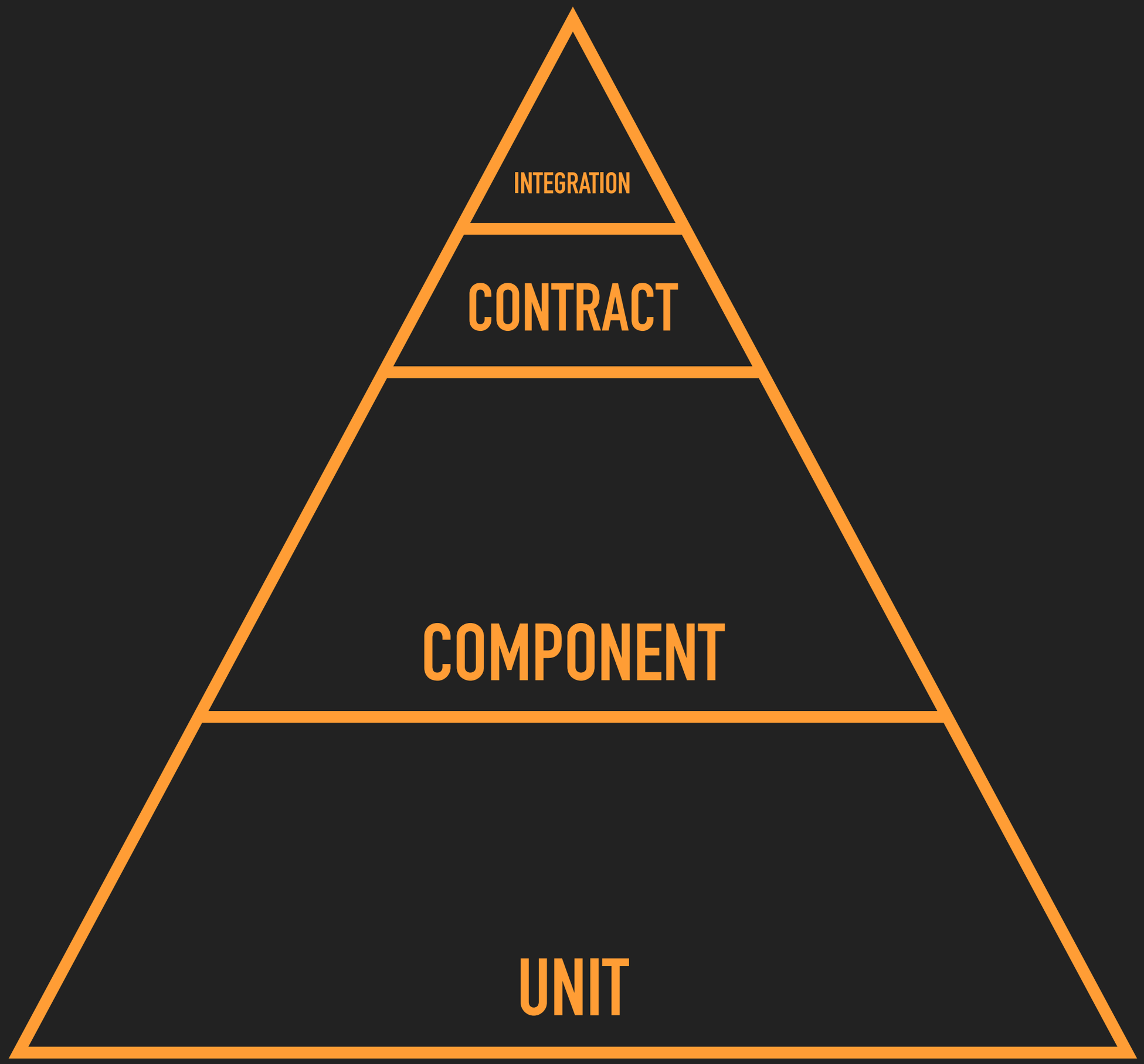
CONSIDERATION #3

DEVELOPMENT VELOCITY

PRODUCT TEAMS

PARALLEL DELIVERY


INTEGRATION
COMPONENT
UNIT


INTEGRATION
CONTRACT
COMPONENT
UNIT

CDC

TAKE HOME

1. What architecture is compatible with your UX?
2. How can you establish defence in depth for your services?
3. How do you enable high delivery velocity with high quality?

SOURCES

Peter Norvig — Numbers every computer engineer should know: http://norvig.com/21-days.html

Wikipedia — Fallacies of distributed computing: https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing

Wikipedia — Economies of scope: https://en.wikipedia.org/wiki/Economies_of_scope

Saltzer, Reed and Clark — END-TO-END ARGUMENTS IN SYSTEM DESIGN: http://web.mit.edu/Saltzer/www/publications/endtoend/endtoend.pdf

Phil Calçado — How we ended up with microservices.: http://philcalcado.com/2015/09/08/how_we_ended_up_with_microservices.html

Brave new geek — Distributed Systems are a UX problem: http://bravenewgeek.com/distributed-systems-are-a-ux-problem/

Ian Robinson — Consumer-Driven Contracts: A Service Evolution Pattern: http://martinfowler.com/articles/consumerDrivenContracts.html

DiUS/pact-jvm — Consumer-Driven Contracts for the JVM: https://github.com/DiUS/pact-jvm

bethesque/pact_broker — Share pacts for Consumer-Driven Contracts: https://github.com/bethesque/pact_broker