BLA BLA
MICROSERVICES
BLA BLA

JONAS BONÉR (AUTHOR)

DUNCAN DEVORE

@IRONFISH

ENGINEER @ LIGHTBEND

KNOCK. KNOCK. WHO'S THERE?

REALITY

TRADITIONAL APPLICATION ARCHITECTURES AND PLATFORMS ARE OBSOLETE.

– GARTNER

MICROSERVICES.

WHAT EXACLTY ARE THEY?

ONE DEFINITION

SUPPORTS MULTIPLE AUTONOMOUS TEAMS
ORGANIZED TO SCALE THE DEVELOPMENT
WHERE THE TEAMS CAN DEVELOP, DEPLOY AND MANAGE
THEIR SERVICES INDEPENDENTLY

THE ARCHITECTURAL CONTEXT OF MICROSERVICES:
DISTRIBUTED SYSTEMS

ANOTHER DEFINITION

A SYSTEM OF AUTONOMOUS COLLABORATIVE DISTRIBUTED SERVICES

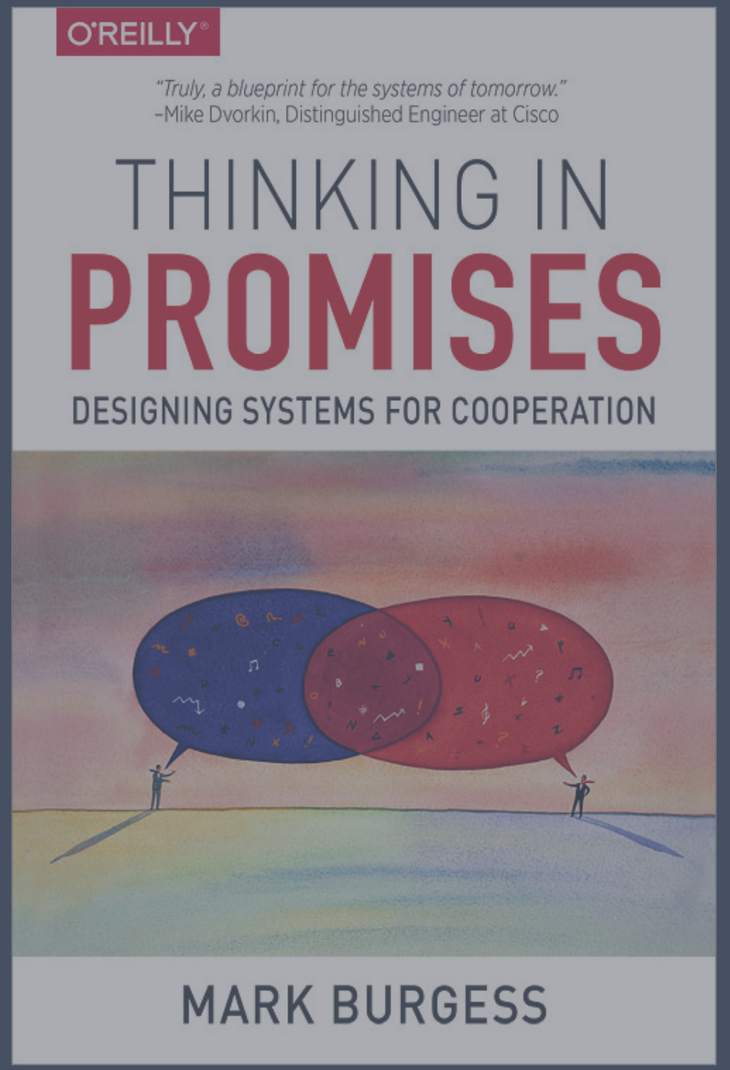
AUTONOMY

FROM GREEK AUTO-NOMOS:

AUTO MEANING SELF

NOMOS MEANING LAW

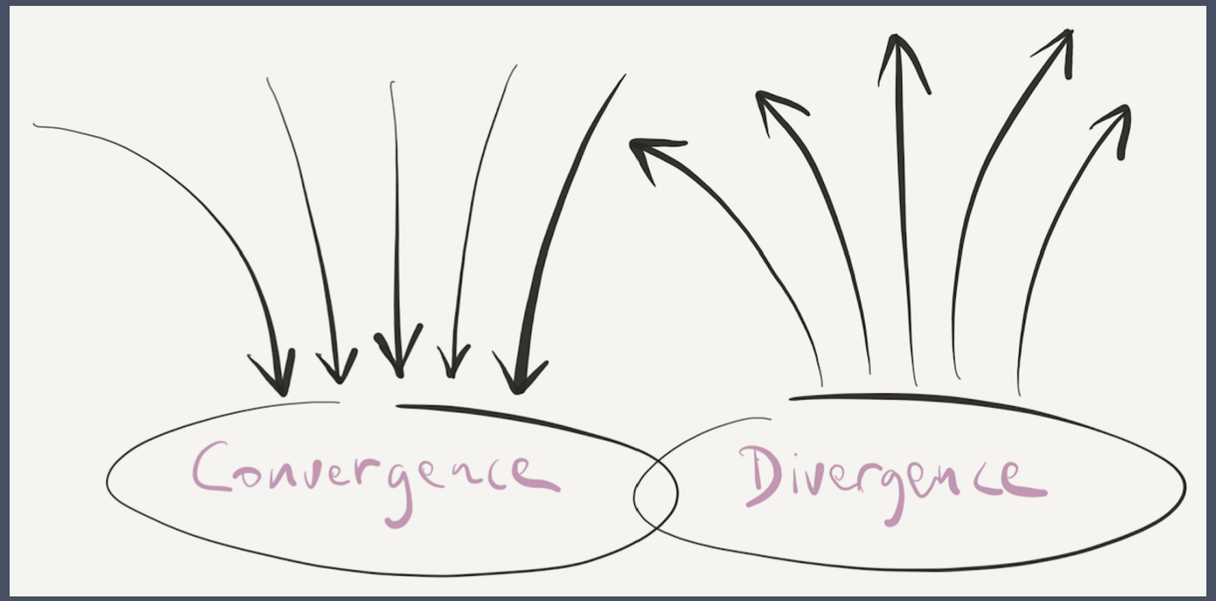
PROMISE THEORY
LEADS THE WAY



THINK IN PROMISES NOT OBLIGATIONS

AUTONOMY MAKES INFORMATION LOCAL. LEADING TO GREATER CERTAINTY AND STABILITY

–IN SEARCH FOR CERTAINTY BY MARK BURGESS



- OBLIGATIONS DIVERGE INTO UNPREDICTABLE OUTCOMES FROM DEFINITE BEGINNINGS ⇒ DECREASED CERTAINTY
- PROMISES CONVERGE TOWARDS A DEFINITE OUTCOME FROM UNPREDICTABLE BEGINNINGS ⇒ IMPROVED CERTAINTY


Meri Williams
@Geek_Manager
Follow
"Promises are not guarantees. Guarantees are not possible" @markburgess_osl #operability #OlO16
RETWEETS
10
LIKES
9
11:27 AM - 19 Sep 2016

AN AUTONOMOUS SERVICE
CAN ONLY PROMISE
ITS OWN BEHAVIOUR

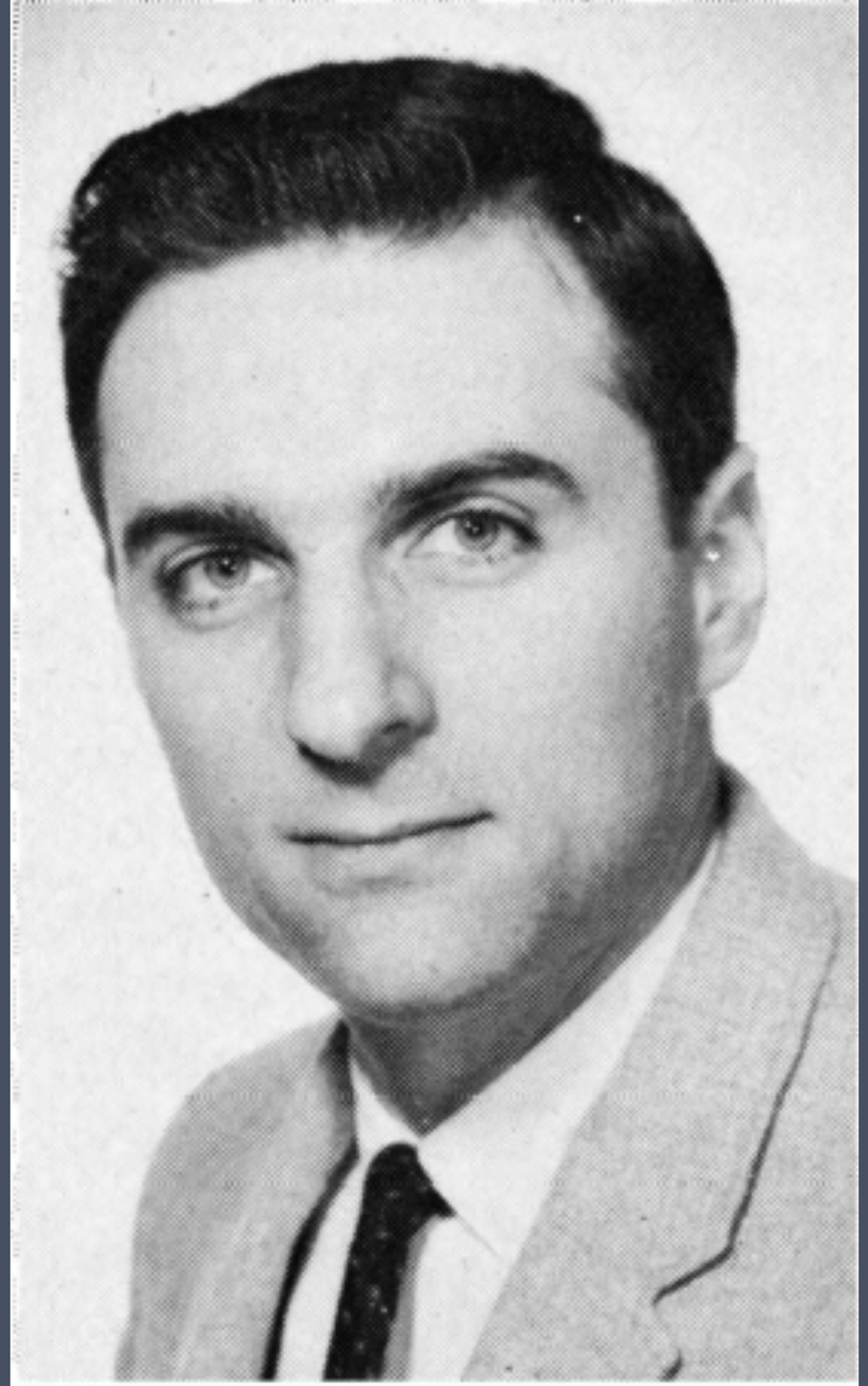

ISOLATE
ALL THE THINGS!

IT WILL SLICE UP YOUR

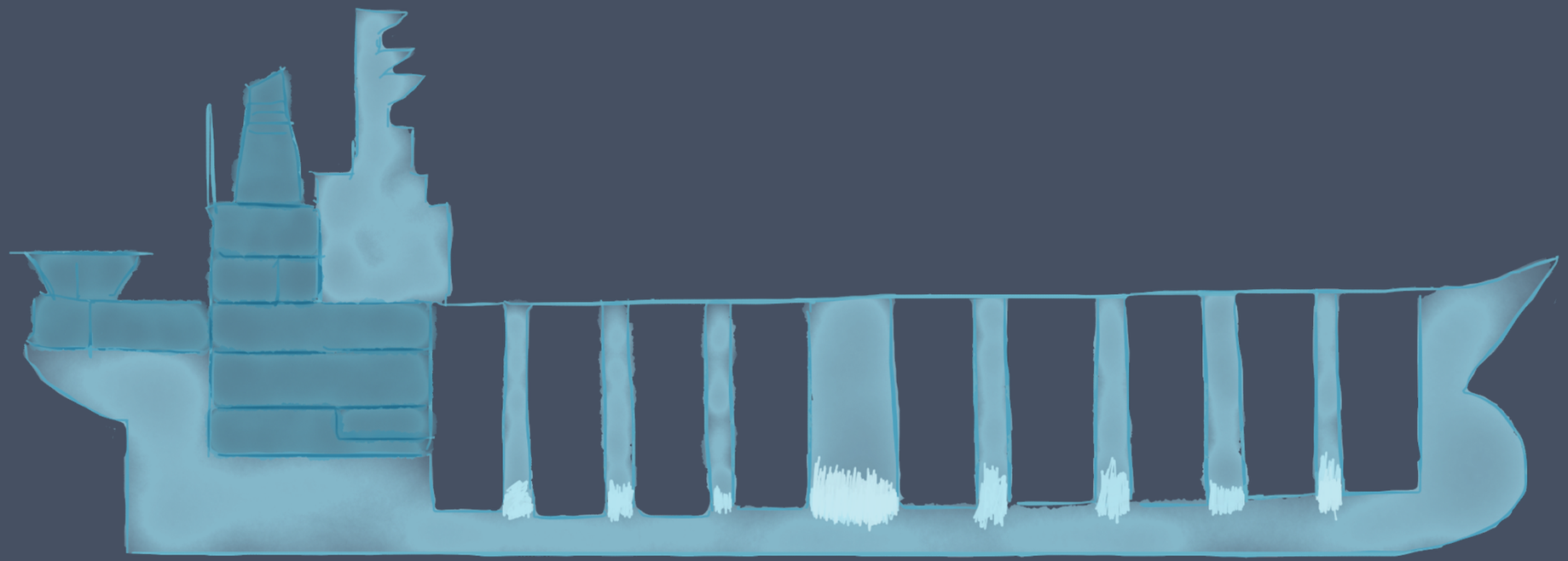
1. ORGANIZATION
2. ARCHITECTURE

ORGANIZATIONS WHICH DESIGN SYSTEMS ...ARE CONSTRAINED TO PRODUCE DESIGNS WHICH ARE COPIES OF THE COMMUNICATION STRUCTURES OF THESE ORGANIZATIONS.

- MELVIN CONWAY

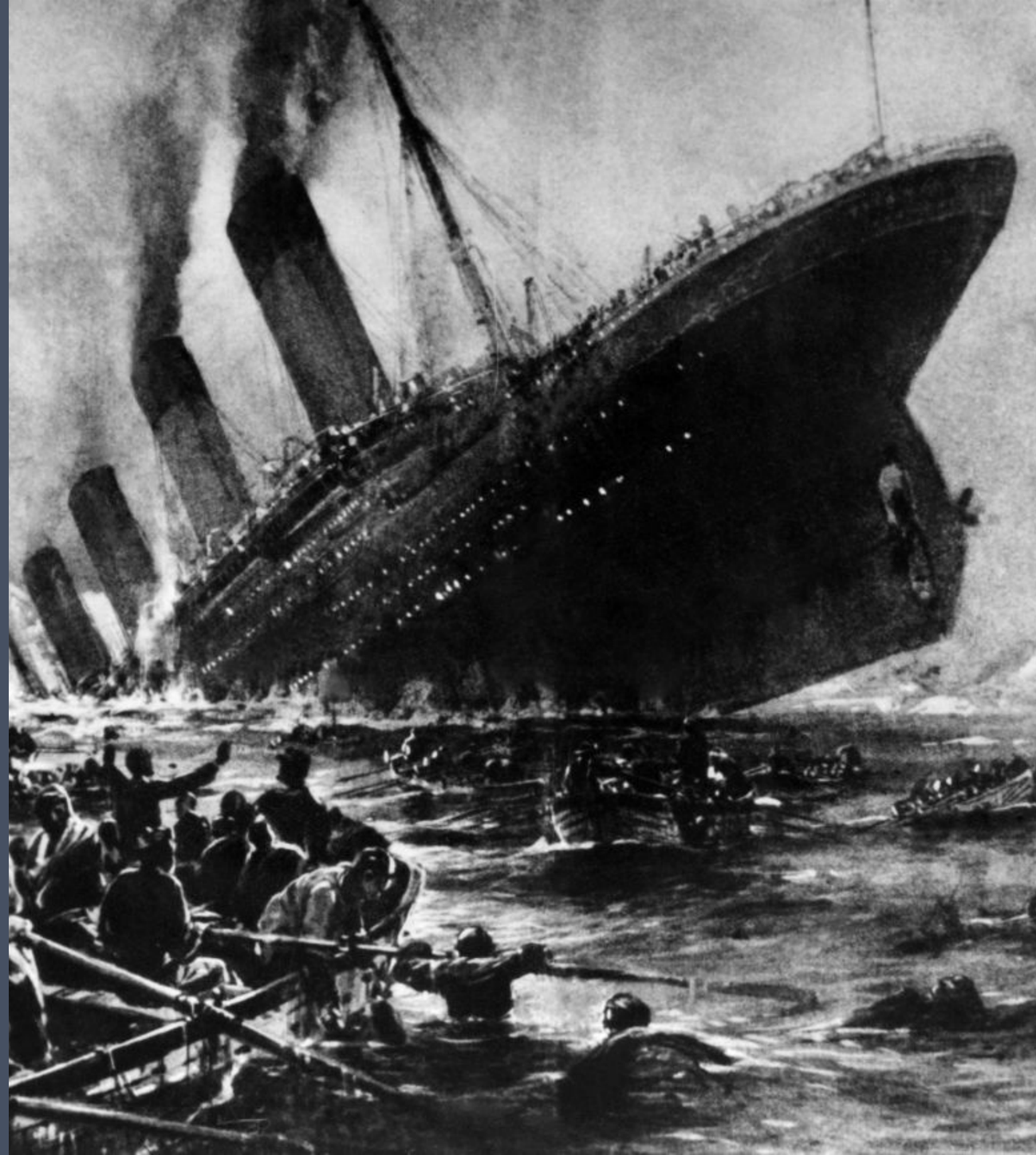


USE BULKHEADING



BUT

WHAT ABOUT THE TITANIC?



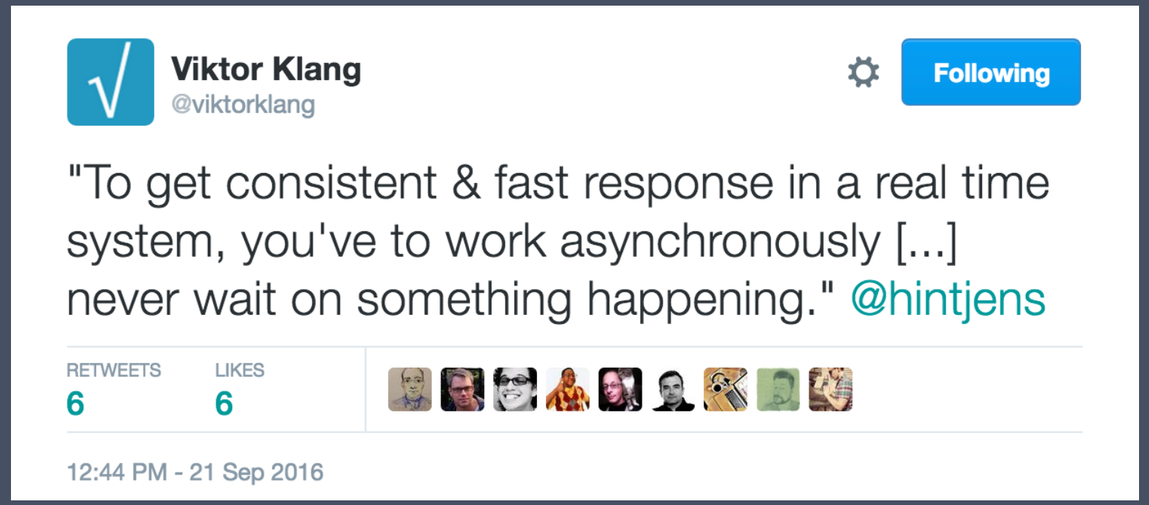
RESILIENCE

IS THE ABILITY TO SELF-HEAL. WHICH REQUIRES

COMPARTMENTALIZATION OF FAILURE

GO

ASYNCHRONOUS



ASYNC IO

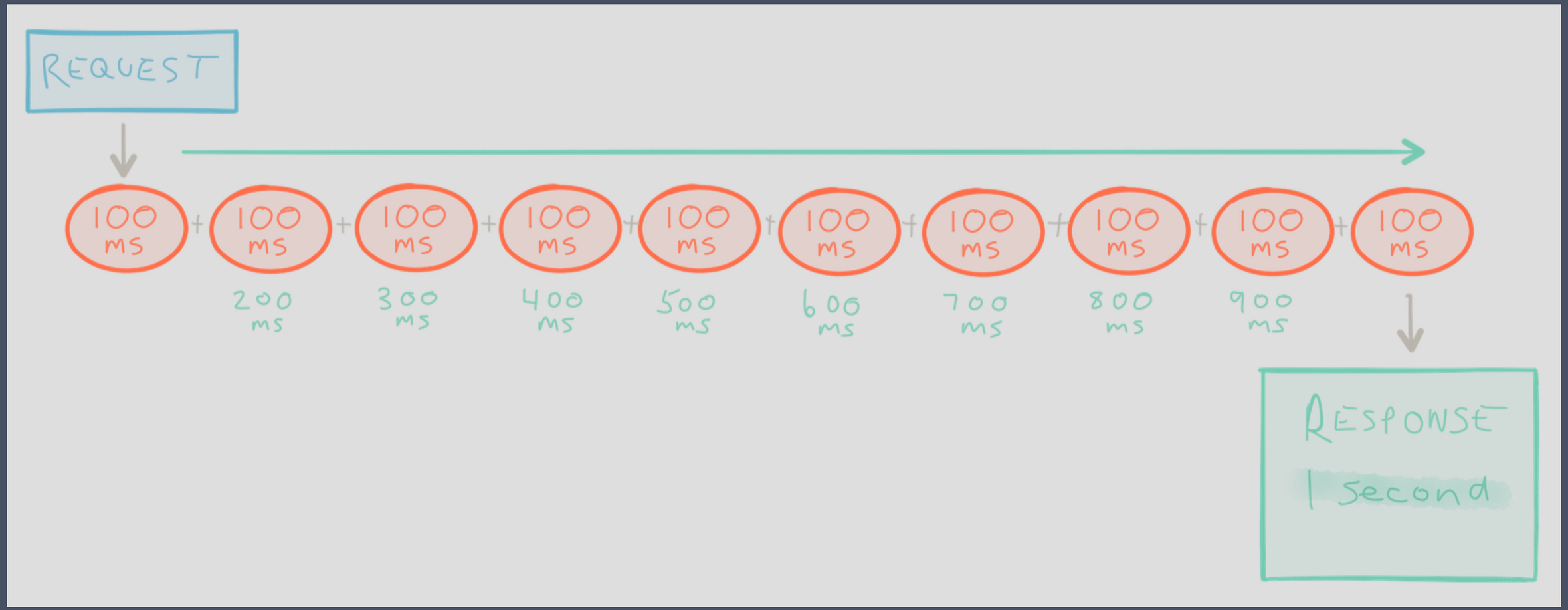
–IS ABOUT **NOT BLOCKING THREADS**

ASYNC COMMUNICATION

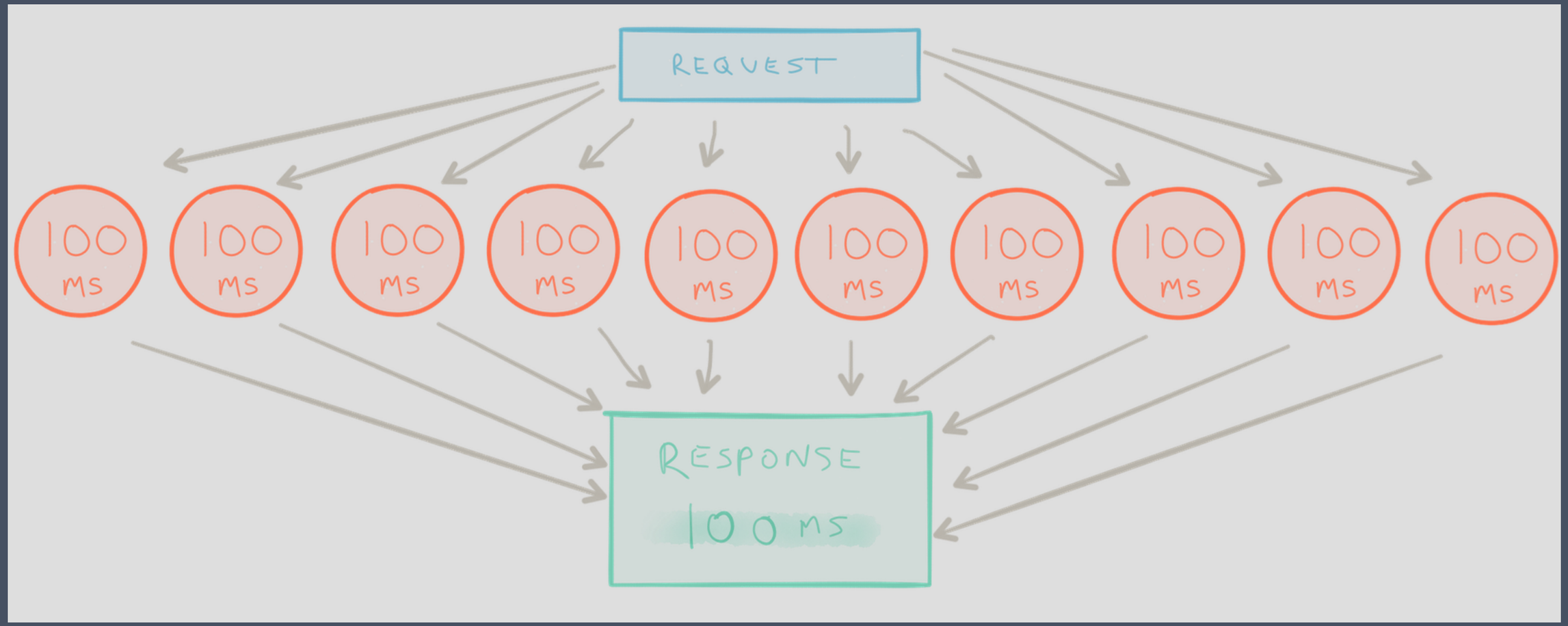
–IS ABOUT **NOT BLOCKING REQUESTS**

ASYNCHRONOUS IO

SYNCHRONOUS DISPATCH



ASYNCHRONOUS DISPATCH



ASYNCHRONOUS COMMUNICATION

ASYNC COMMUNICATION ALLOWS DECOUPLING IN

SPACE

AND

TIME

ASYCHRONOUS MESSAGE-PASSING EMBRACES THE CONSTRAINTS OF DISTRIBUTED SYSTEMS

MESSAGE-PASSING ALLOWS FOR LOCATION TRANSPARENCY

ONE COMMUNICATION ABSTRACTION ACROSS ALL DIMENSIONS OF SCALE

CORE ⇒ SOCKET ⇒

CPU ⇒ CONTAINER ⇒

SERVER ⇒ RACK ⇒

DATA CENTER ⇒ SYSTEM

MICROWHAT?

THE UNIX PHILOSOPHY:
WRITE PROGRAMS THAT DO ONE THING & DO IT WELL.
WRITE PROGRAMS TO WORK TOGETHER.

– DOUG MCILROY

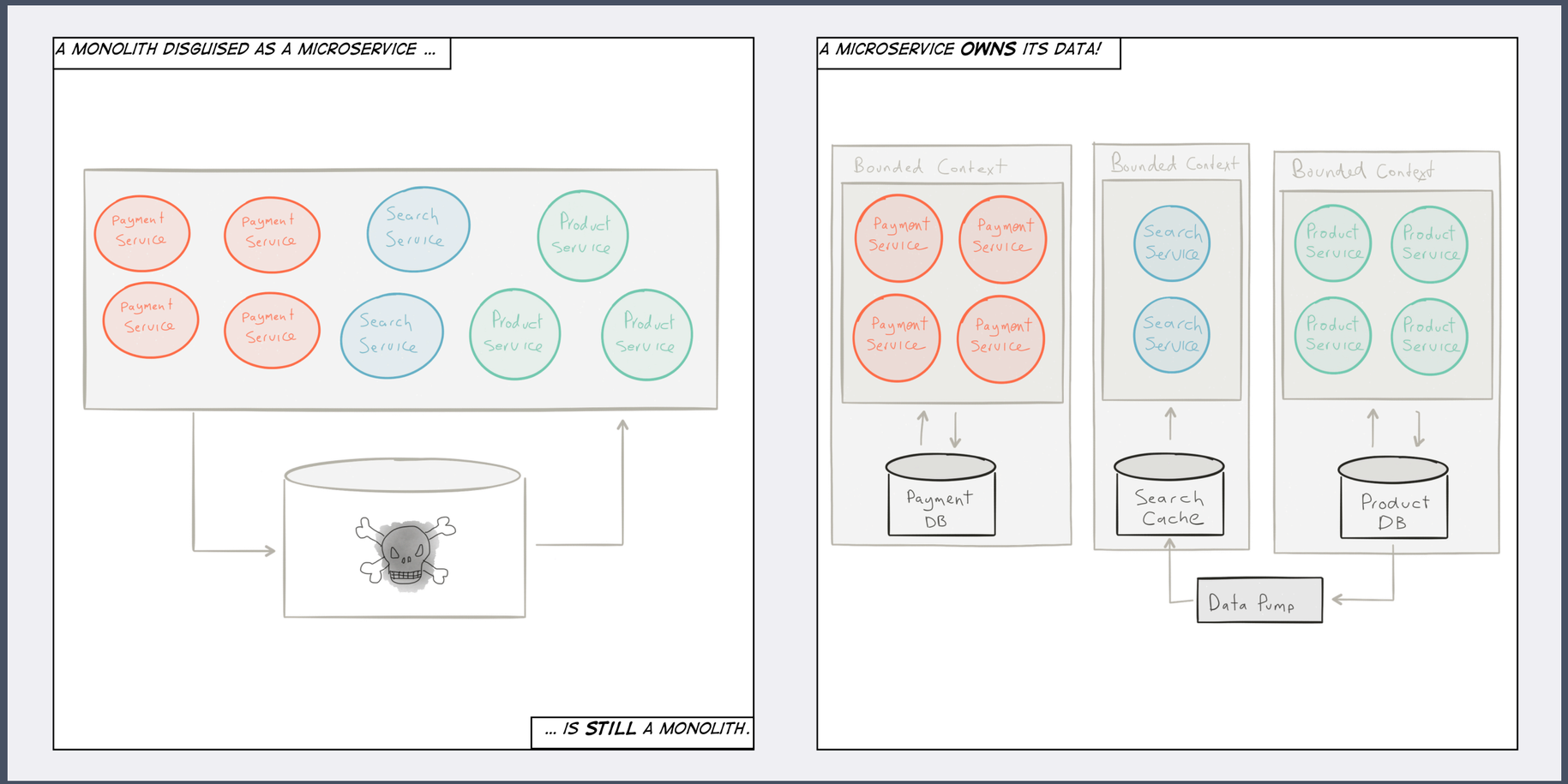
DO ONE THING

AND

DO IT WELL

BUT WHAT ABOUT STATE?

OWN YOUR STATE. EXCLUSIVELY



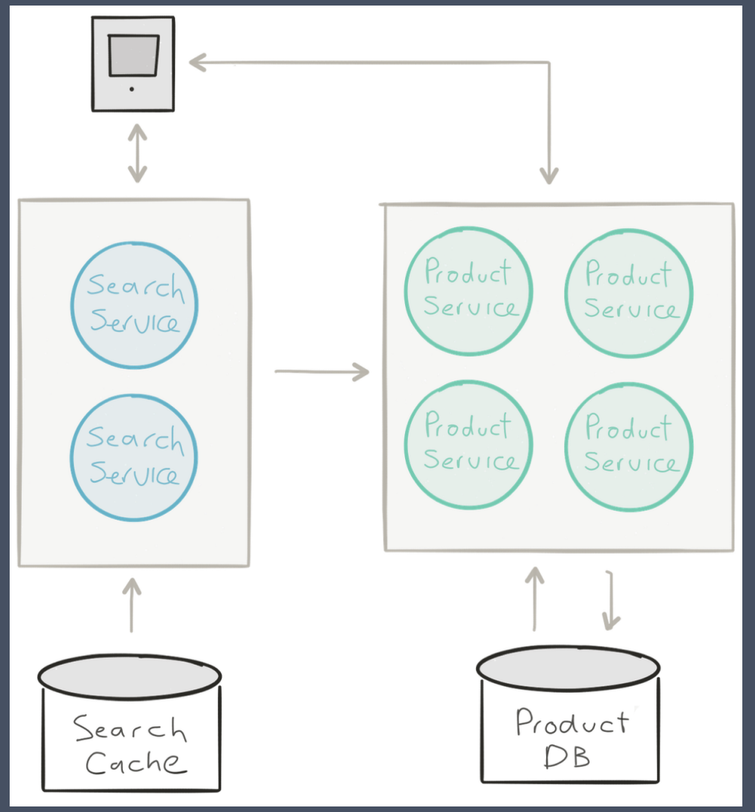
THERE IS NO SUCH THING AS A "STATELESS" ARCHITECTURE

IT'S JUST SOMEONE ELSE'S PROBLEM

THINK IN TERMS OF

CONSISTENCY BOUNDARIES

BOUNDED CONTEXTS



POLYGLOT PERSISTENCE

THE TRUTH IS THE LOG.
THE DATABASE IS A CACHE OF A SUBSET OF THE LOG.

– PAT HELLAND

FAVOR

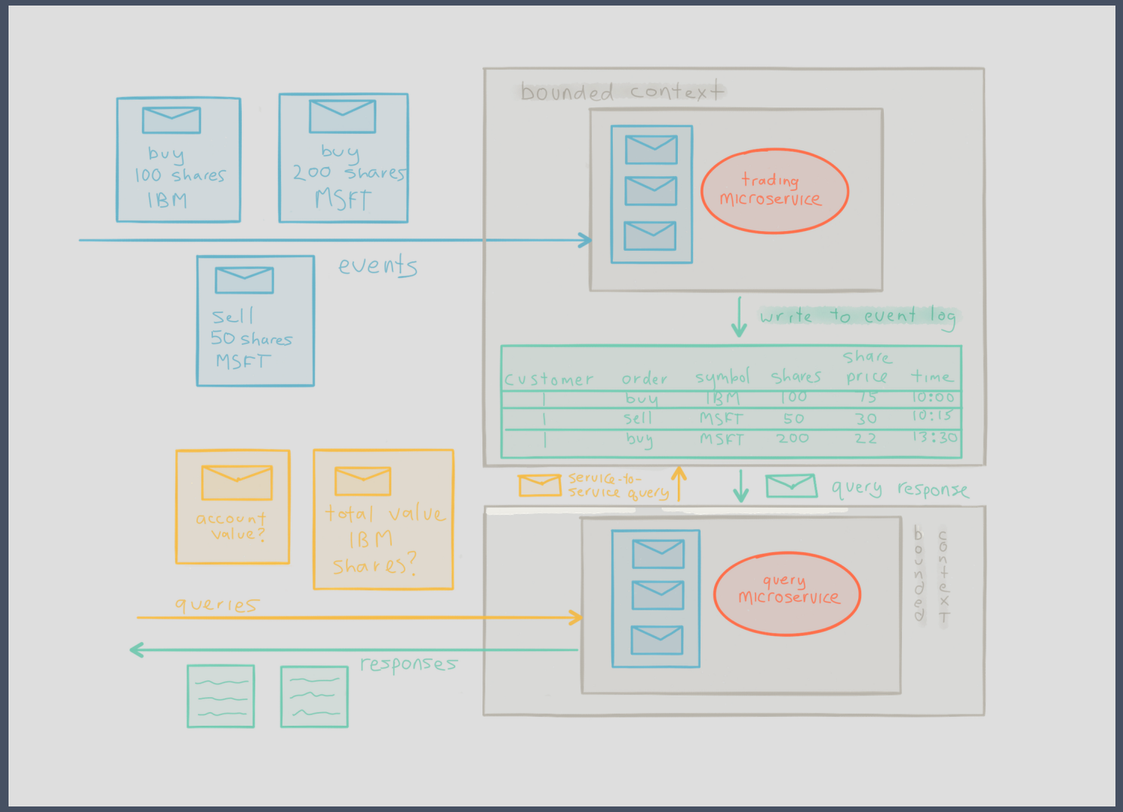
EVENT LOGGING

THE LOG

IS A DATABASE OF THE PAST

NOT JUST A DATABASE OF THE PRESENT

EVENT SOURCING WITH CQRS



EVENT LOGGING AVOIDS THE INFAMOUS

OBJECT-RELATIONAL IMPEDENCE MISMATCH

YOU NEED TO SEPARATE

THE STATELESS PART - THE BEHAVIOR

FROM

THE STATEFUL PART - THE KNOWLEDGE

Life beyond Distributed Transactions: an Apostate's Opinion

Position Paper

Pat Helland

Amazon.Com
705 Fifth Ave South
Seattle, WA 98104
USA
PHelland@Amazon.com

The positions expressed in this paper are personal opinions and do not in any way reflect the positions of my employer Amazon.com.

ABSTRACT

Many decades of work have been invested in the area of distributed transactions including protocols such as 2PC, Paxos, and various approaches to quorum. These protocols provide the application programmer a façade of global serializability. Personally, I have invested a non-trivial portion of my career as a strong advocate for the implementation and use of platforms providing guarantees of global serializability.

My experience over the last decade has led me to liken these platforms to the Maginot Line[1]. In general, application developers simply do not implement large scalable applications assuming distributed transactions. When they attempt to use distributed transactions, the projects founder because the performance costs and fragility make them impractical. Natural selection kicks in… Instead, applications are built using different techniques which do not provide the same transactional guarantees but still meet the needs of their businesses.

This paper explores and names some of the practical approaches used in the implementations of large-scale mission-critical applications in a world which rejects distributed transactions. We discuss the management of fine-grained pieces of application data which may be repartitioned over time as the application grows. We also discuss the design patterns used in sending messages between these repartitionable pieces of data.

The reason for starting this discussion is to raise awareness of new patterns for two reasons. First, it is my belief that this awareness can ease the challenges of people hand-crafting very large scalable applications. Second, by observing the patterns, hopefully the industry can work towards the creation of platforms that make it easier to build these very large applications.

[1] The Maginot Line was a huge fortress that ran the length of the Franco-German border and was constructed at great expense between World War I and World War II. It successfully kept the German army from directly crossing the border between France and Germany. It was quickly bypassed by the Germans in 1940 who invaded through Belgium.

3rd Biennial Conference on Innovative DataSystems Research (CIDR) January 7-10, Asilomar, California USA.

1. INTRODUCTION

Let's examine some goals for this paper, some assumptions that I am making for this discussion, and then some opinions derived from the assumptions. While I am keenly interested in high availability, this paper will ignore that issue and focus on scalability alone. In particular, we focus on the implications that fall out of assuming we cannot have large-scale distributed transactions.

Goals

This paper has three broad goals:

- Discuss Scalable Applications
  Many of the requirements for the design of scalable systems are understood implicitly by many application designers who build large systems.

STAY MOBILE
BUT ADDRESSABLE

BUT I'LL TAKE MY TIME ANYWHERE.
I'M FREE TO SPEAK MY MIND ANYWHERE.
AND I'LL REDEFINE ANYWHERE.
ANYWHERE I ROAM.
WHERE I LAY MY HEAD IS HOME.

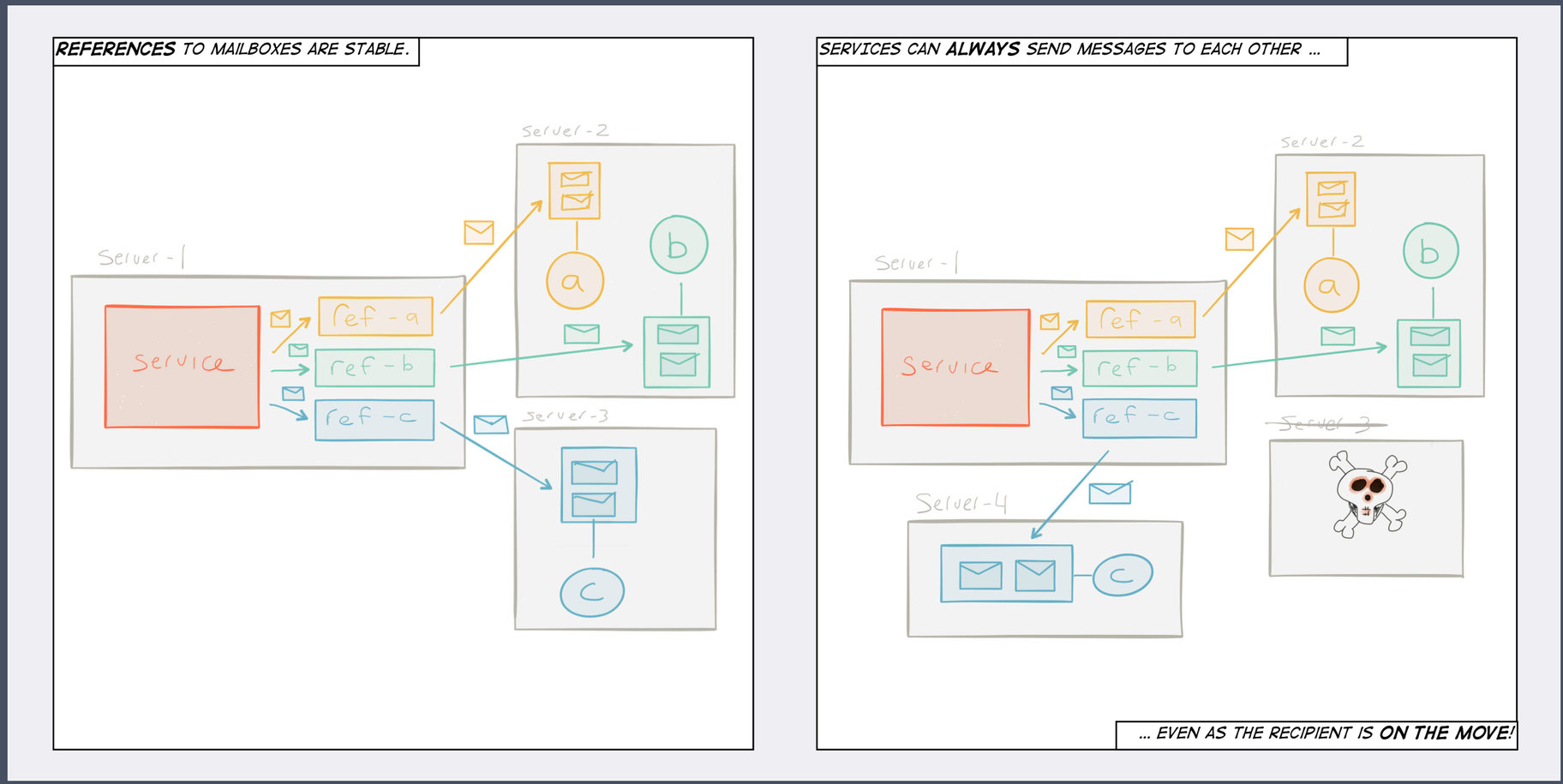
- WHEREVER I MAY ROAM BY LARS ULRICH, JAMES HETFIELD (METALLICA)

WHY WOULD I NEED

VIRTUAL ADRESSING?

1. LOAD-BALANCING BETWEEN STATELESS SERVICES
2. STATE REPLICATION OF STATEFUL SERVICES
3. RELOCATION OF A STATEFUL SERVICE

REFERENCES SHOULD ALWAYS WORK



...AND NOW WHAT?


fresh tweets
@tweets_so_fresh
Follow
when u have converted everything to
#microservices
RETWEETS
132
LIKES
174
9:34 PM - 15 May 2016

ONE ACTOR IS NO ACTOR.
ACTORS COME IN SYSTEMS.

- CARL HEWITT

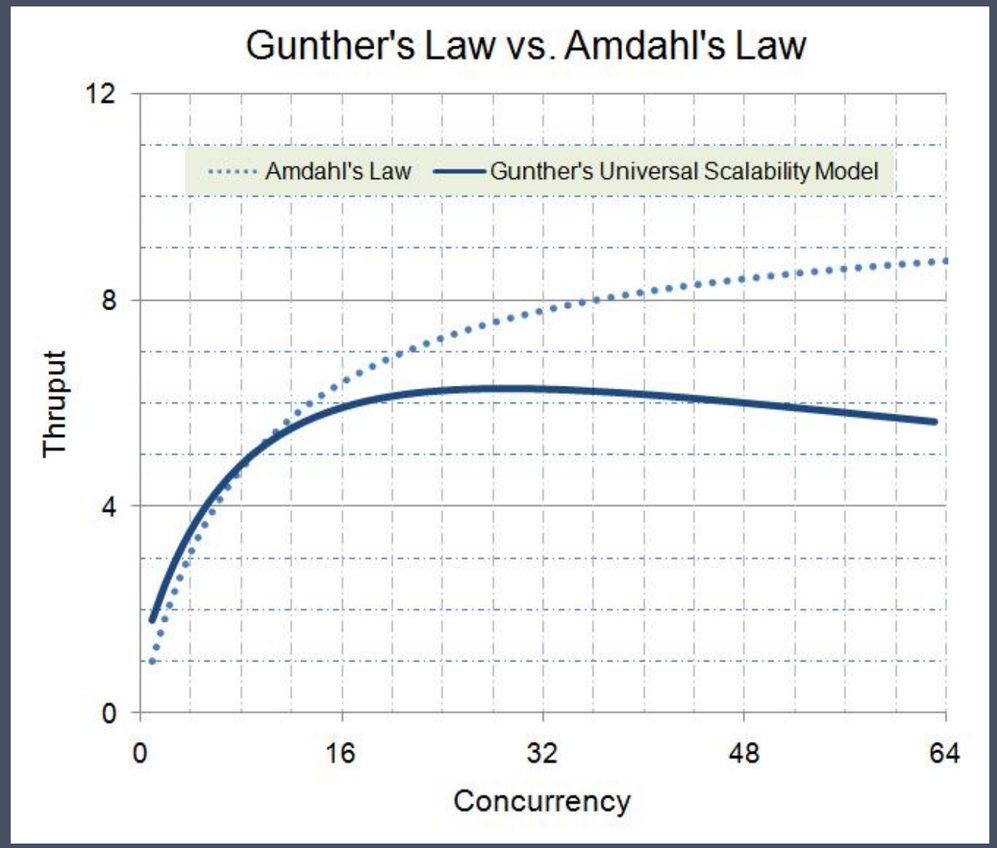
MICROSERVICES
COME IN SYSTEMS

SYSTEMS NEED TO EXPLOIT REALITY

INFORMATION HAS LATENCY

WE ARE ALWAYS LOOKING INTO THE PAST

THE COST OF MAINTAINING THE ILLUSION OF AN ABSOLUTE NOW




AS LATENCY GETS HIGHER. THE
ILLUSION CRACKS EVEN MORE

IN A **DISTRIBUTED** SYSTEM, YOU CAN KNOW

WHERE

THE WORK IS DONE

OR YOU CAN KNOW **WHEN**

THE WORK IS DONE

BUT YOU **CAN'T KNOW BOTH**

– PAT HELLAND

PERFECT STORM

ISLAND

INSIDE DATA: OUR CURRENT PRESENT
OUTSIDE DATA: BLAST FROM THE PAST
BETWEEN SERVICES: HOPE FOR THE FUTURE

– **PAT HELLAND** (DATA ON THE INSIDE VS DATA ON THE OUTSIDE)

STRIVE TO MINIMIZE

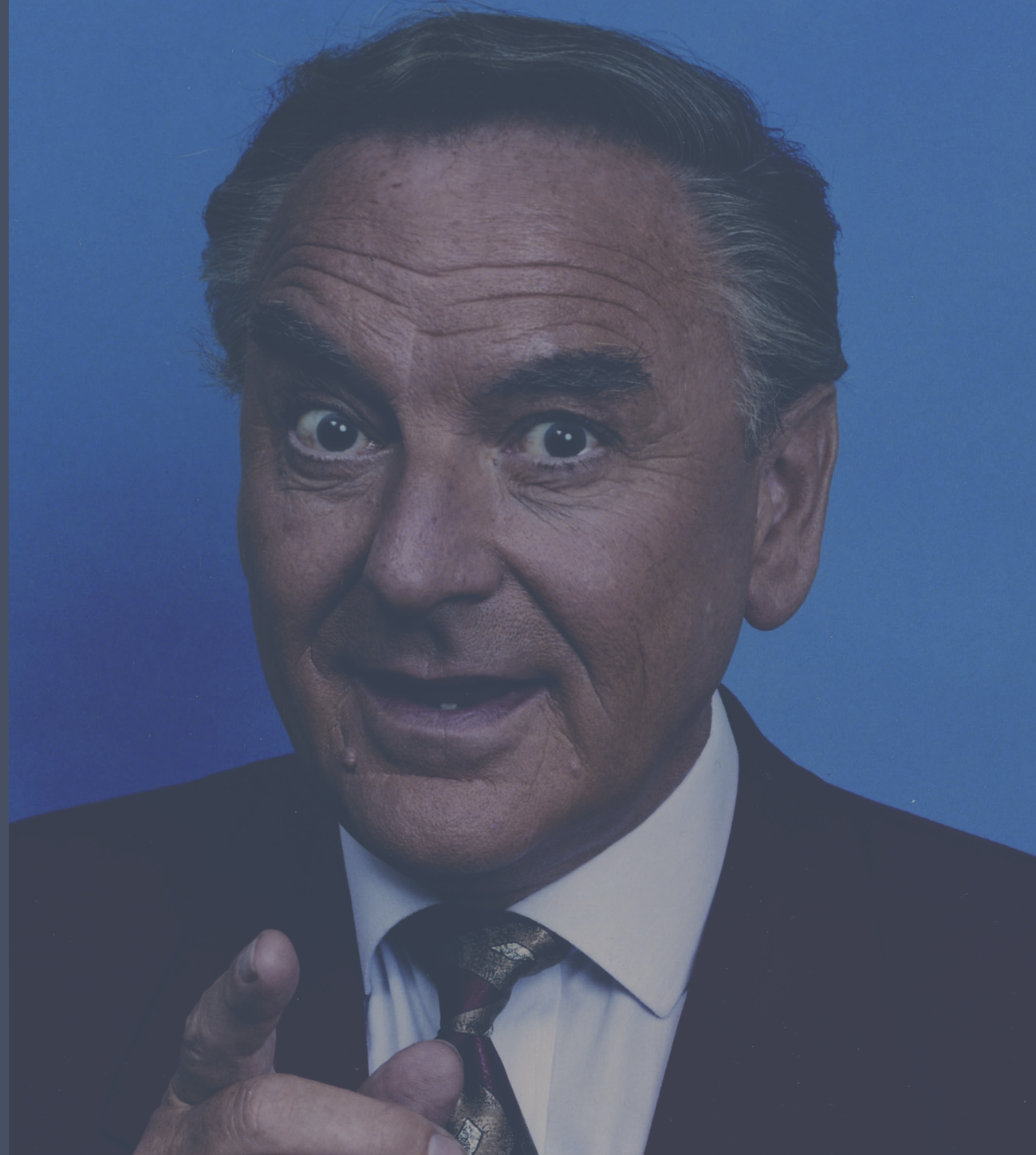
COUPLING & COMMUNICATION

WORDS ARE VERY
UNECESSARY.
THEY CAN ONLY DO HARM.
ENJOY THE SILENCE.

- ENJOY THE SILENCE BY MARTIN GORE
(DEPECHE MODE)

SILENCE IS NOT ONLY GOLDEN. IT IS SELDOM MISQUOTED.

- BOB MONKHOUSE

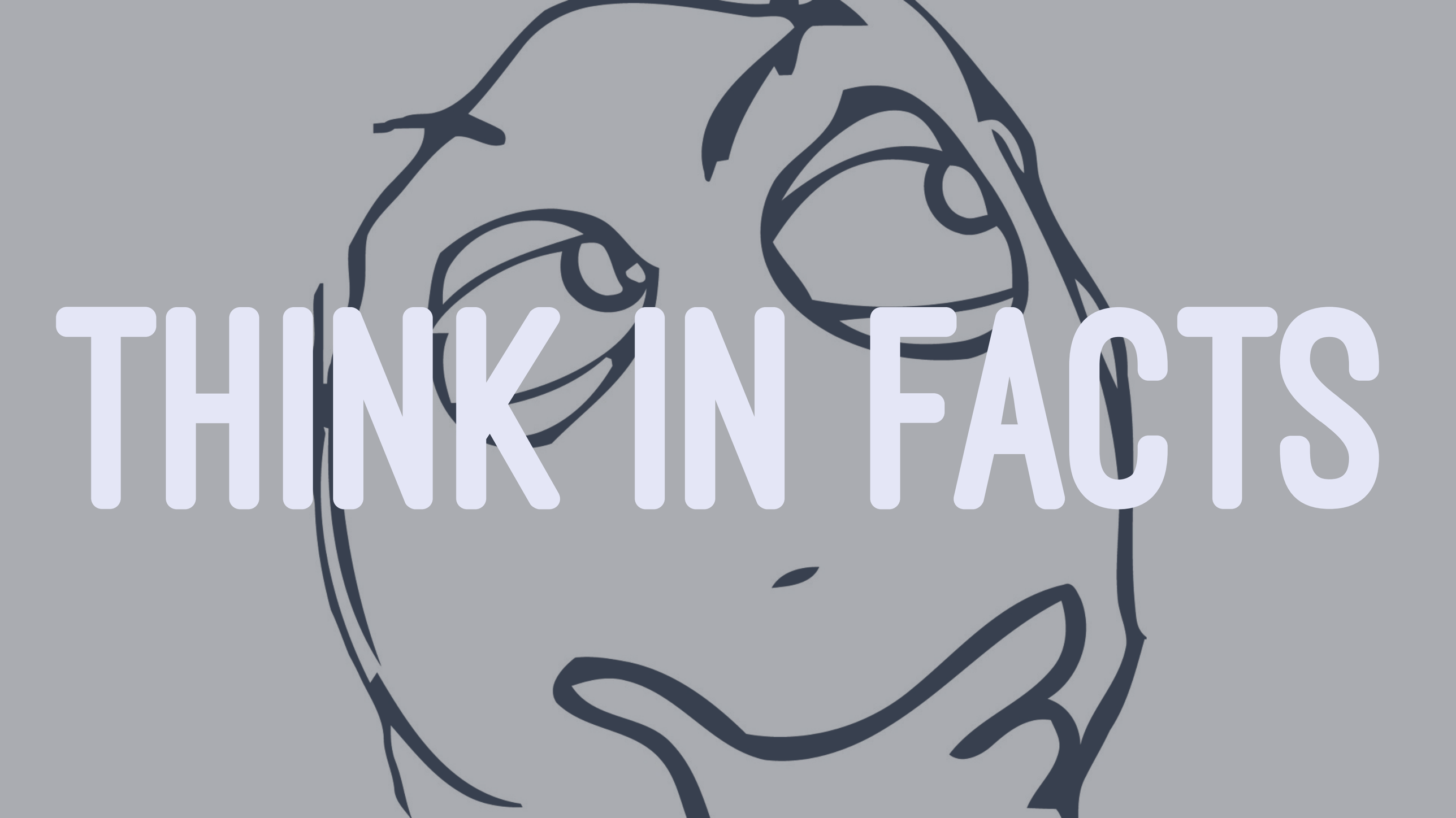


THE CONTENTS OF
A MESSAGE ARE
ALWAYS FROM THE PAST!
THEY ARE NEVER "NOW."

– PAT HELLAND

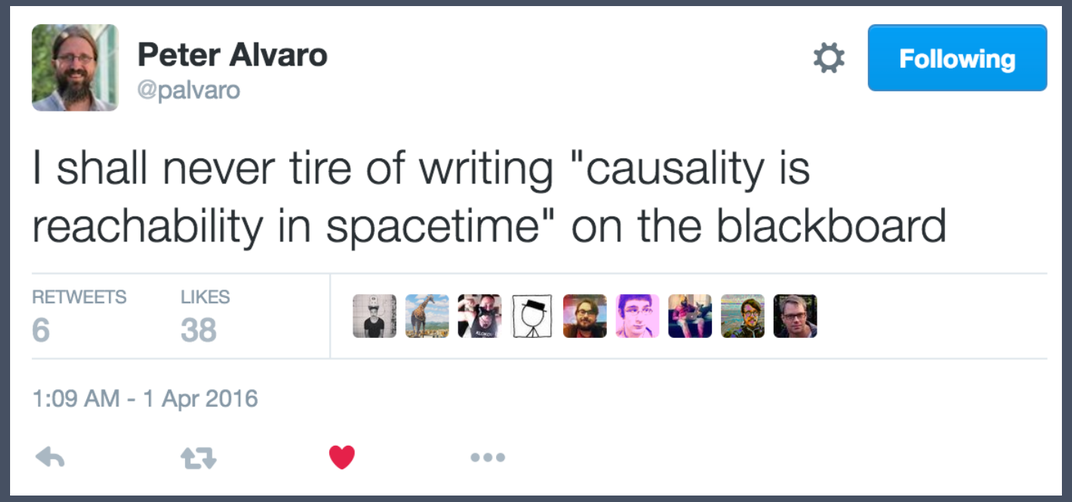
WE HAVE TO RELY ON
EVENTUAL CONSISTENCY
BUT DON'T BE SURPRISED
IT'S HOW THE WORLD WORKS

NO ONE WANTS
EVENTUAL CONSISTENCY
IT IS A NECESSARY EVIL


THINK IN FACTS

CAUSALITY

IS A PATH TO KNOWLEDGE



TO CONDENSE FACT FROM THE VAPOR OF NUANCE

– NEAL STEPHENSON, SNOW CRASH

WE NEED TO

CONTAIN MUTABLE STATE &

PUBLISH FACTS

HOW CAN WE MANAGE PROTOCOL EVOLUTION?

BE CONSERVATIVE
IN WHAT YOU DO.
BE LIBERAL IN WHAT YOU
ACCEPT FROM OTHERS.

– JON POSTEL

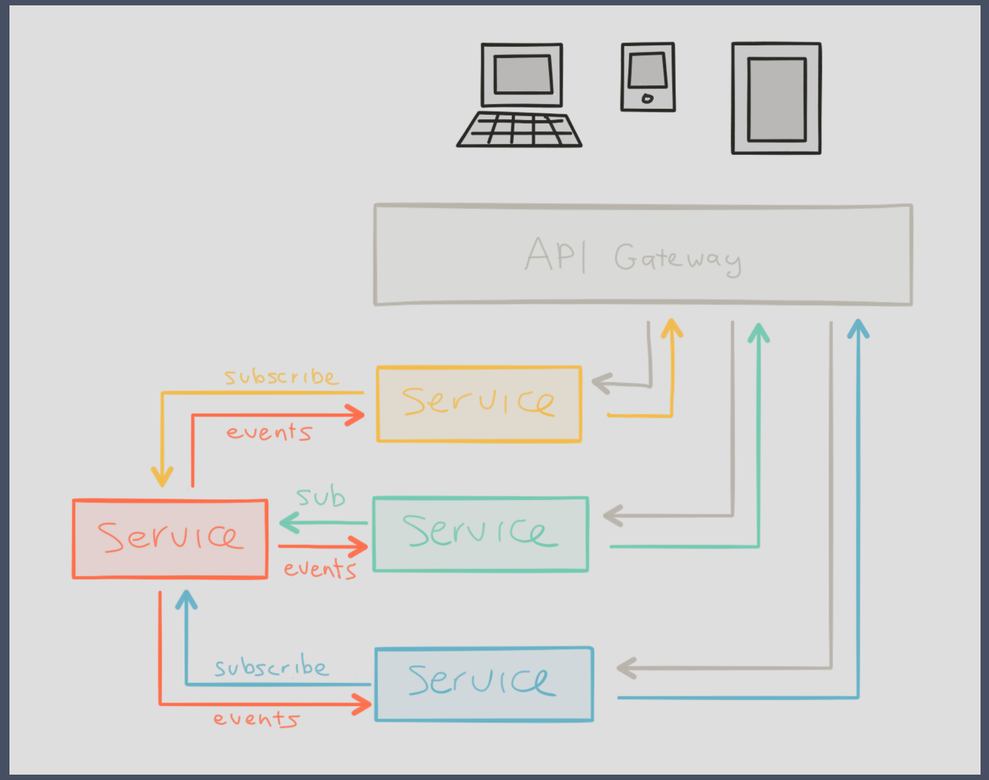
CONSIDER USING AN

ANTI-CORRUPTION LAYER

(CLASSIC DDD PATTERN)

CONSIDER USING AN API GATEWAY

TO SIMPLIFY CLIENT COMMUNICATION



HOW CAN WE MANAGE THE COMPLEXITY OF COMMUNICATION?

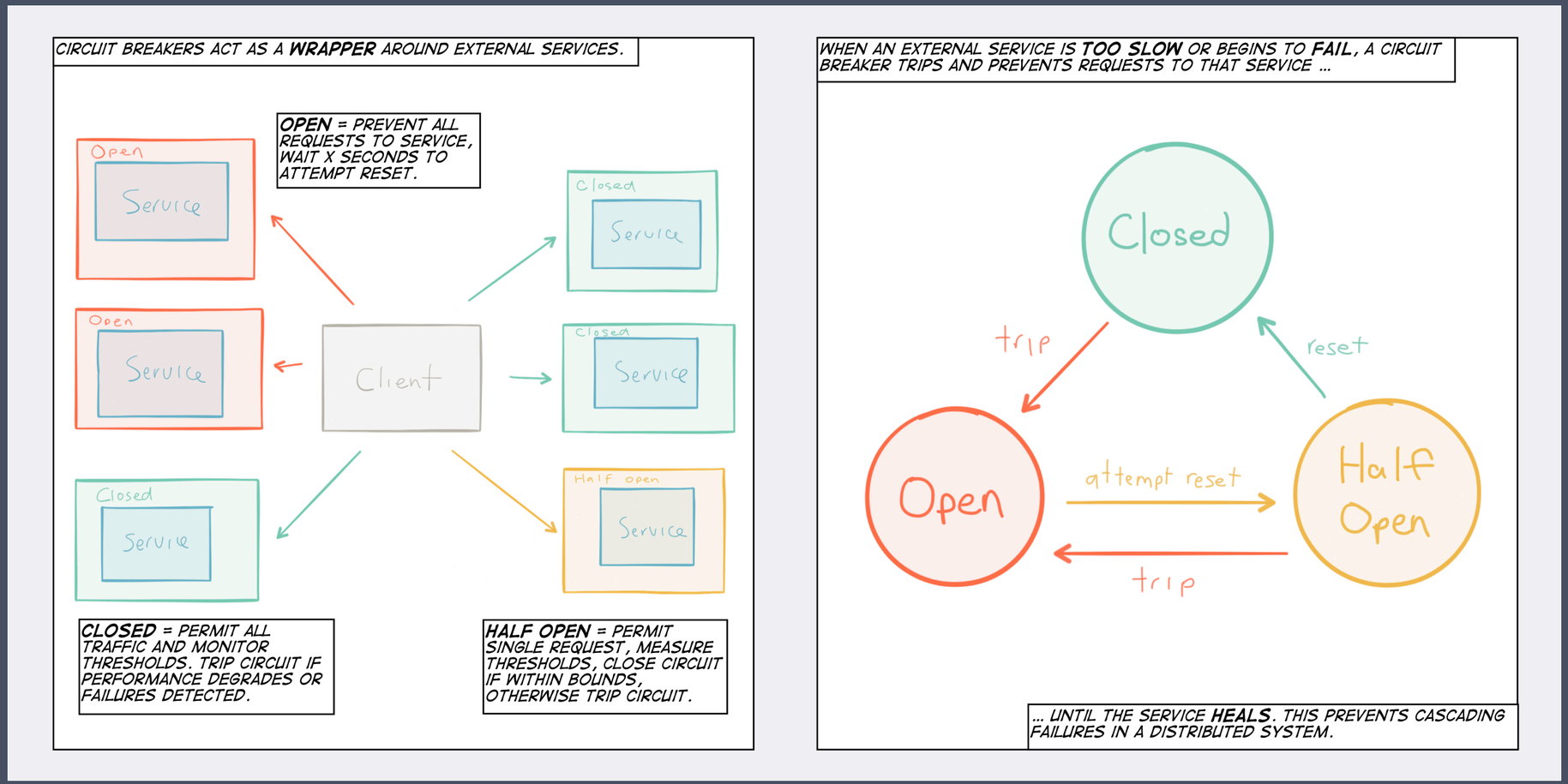
INTEGRATION

WITH OTHER SYSTEMS

IS A RISKY BUSINESS

SYNCHRONOUS COMMUNICATION PUTS YOU AT THE MERCY OF THE OTHER SYSTEM

USE CIRCUIT BREAKERS



ALWAYS APPLY BACK-PRESSURE

OR SOMEONE WILL GET HURT

A FAST SYSTEM
SHOULD NOT OVERLOAD
A SLOW SYSTEM

BUT

WHAT ABOUT

TRANSACTIONS?

IN GENERAL.
APPLICATION DEVELOPERS
SIMPLY DO NOT IMPLEMENT
LARGE SCALABLE APPLICATIONS
ASSUMING DISTRIBUTED
TRANSACTIONS.

– PAT HELLAND

"TWO-PHASE COMMIT IS THE ANTI-AVAILABILITY PROTOCOL."

- PAT HELLAND

GUESS.
APOLOGIZE.
COMPENSATE.

IT'S HOW THE WORLD WORKS

BUT... I REALLY NEED TRANSACTIONS

SAGA PATTERN

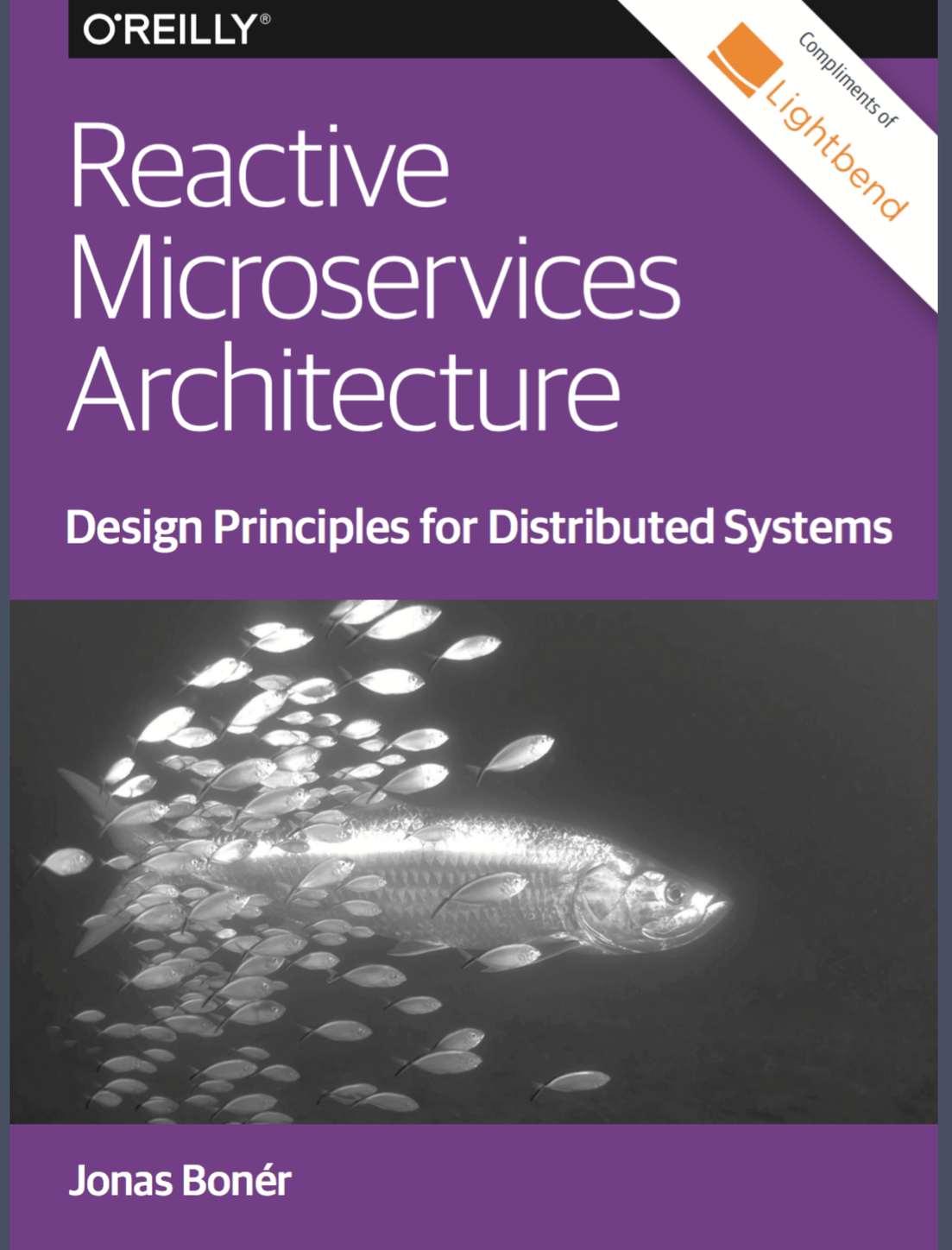
IN SUMMARY

EMBRACE REALITY AND ITS CONSTRAINTS SHALL SET YOU FREE

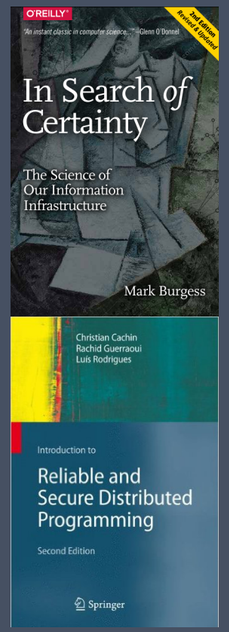
(OR SOMETHING LIKE THAT)

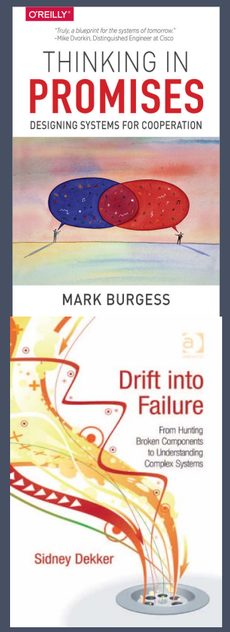
LEARN MORE

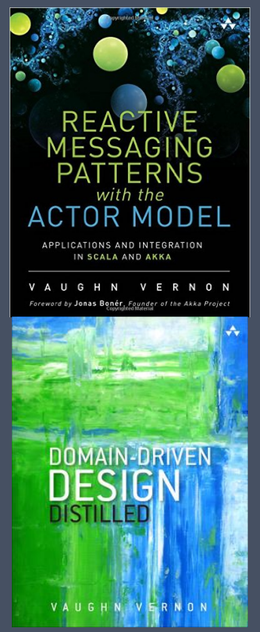
HTTP://BIT.LY/REACTIVE-MICROSERVICES-ARCHITECTURE



NEVER. STOP. READING.







SOME BOOKS TO GET YOU STARTED

THANK YOU