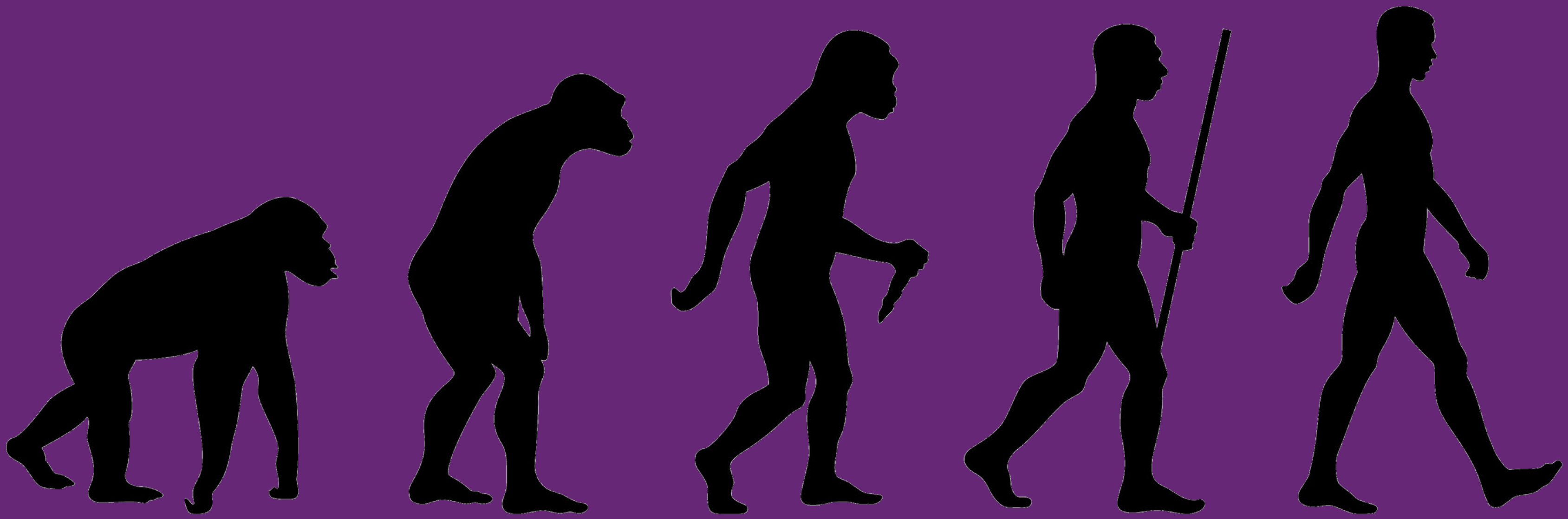


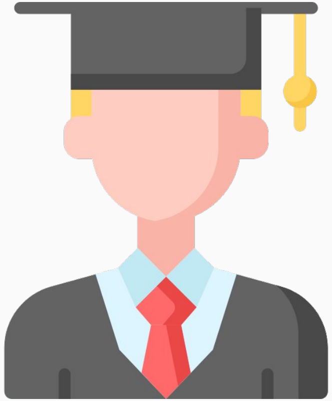
Microservices as an Evolutionary Architecture: Lessons learned

Luram Archanjo

Who am I?

Luram Archanjo

- Software Engineer @Sensedia
- MBA in Java projects
- Java and Microservice enthusiast



Agenda

- Use Case
- Microservices
- Evolutionary Architecture
- Communication Patterns
- Concurrency Pattern
- Questions

Use case

THE CHALLENGE

🕒 Response time

📶 Throughput

In the beginning

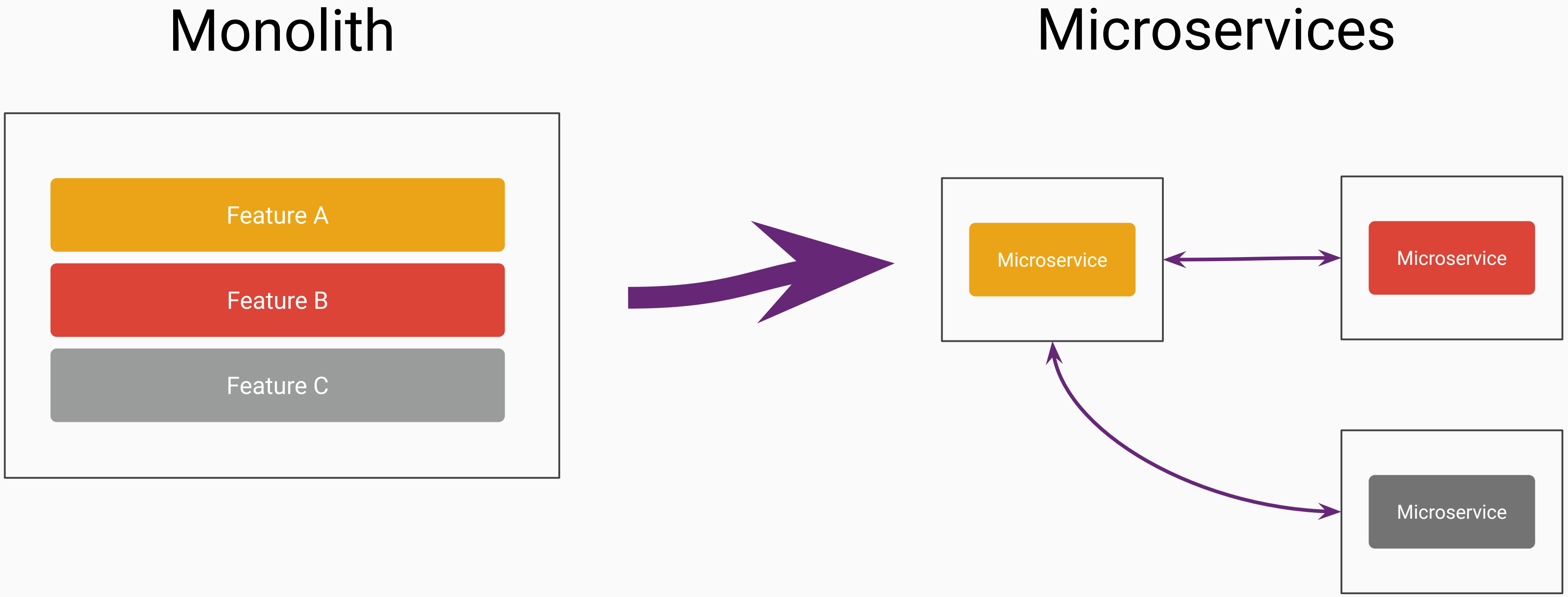
As is

- Single communication pattern
- Java & Spring everywhere
- Synchronous process everywhere
- 50-60% of our commits during the sprint were in the same repository
- Service with different workload requirements (Crypto and Business)

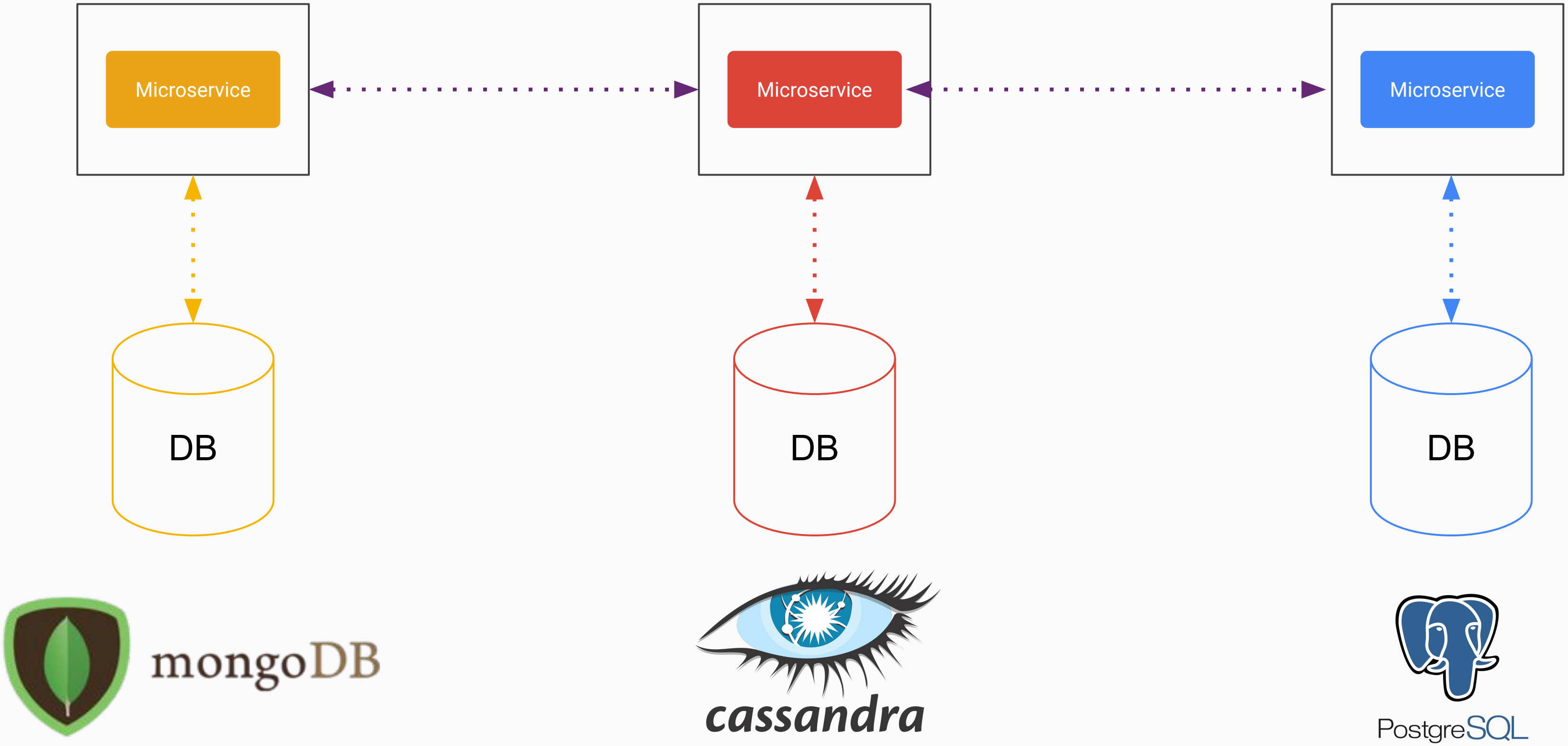
We were in the wrong way to adopt microservices architecture!

Let's recap what is the microservices architecture!

Moving to Microservices



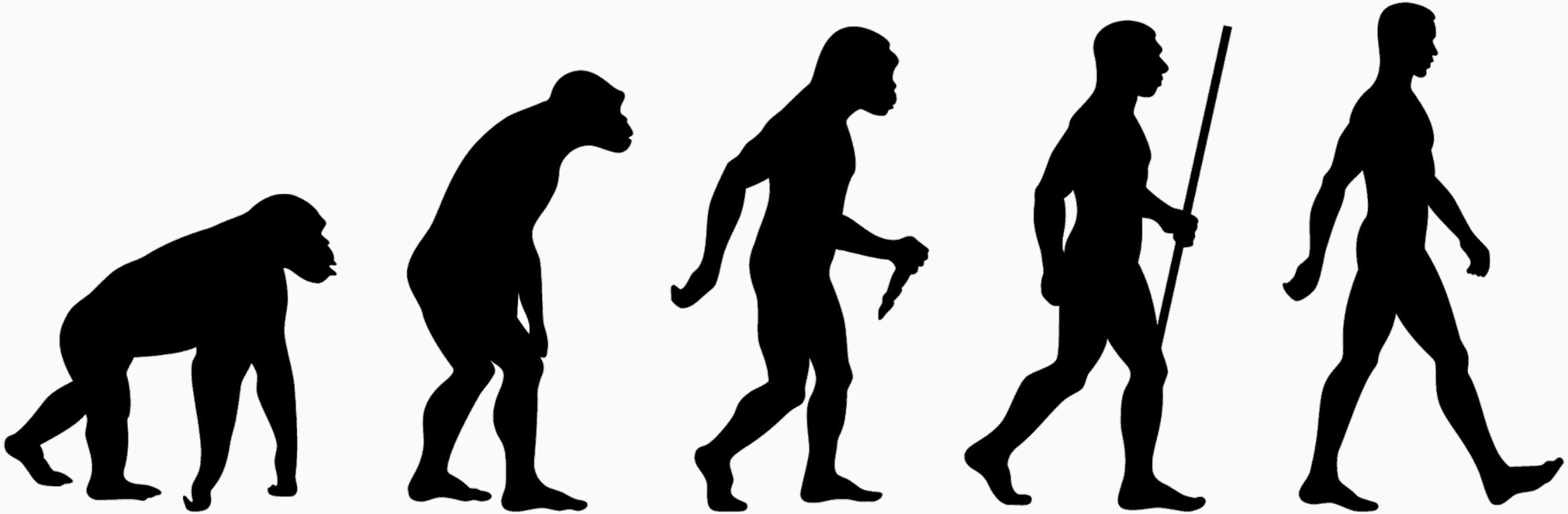
Technological Heterogeneity



Evolutionary Architecture

Evolutionary Architecture

An evolutionary architecture supports guided, incremental change as a first principle across multiple dimensions.

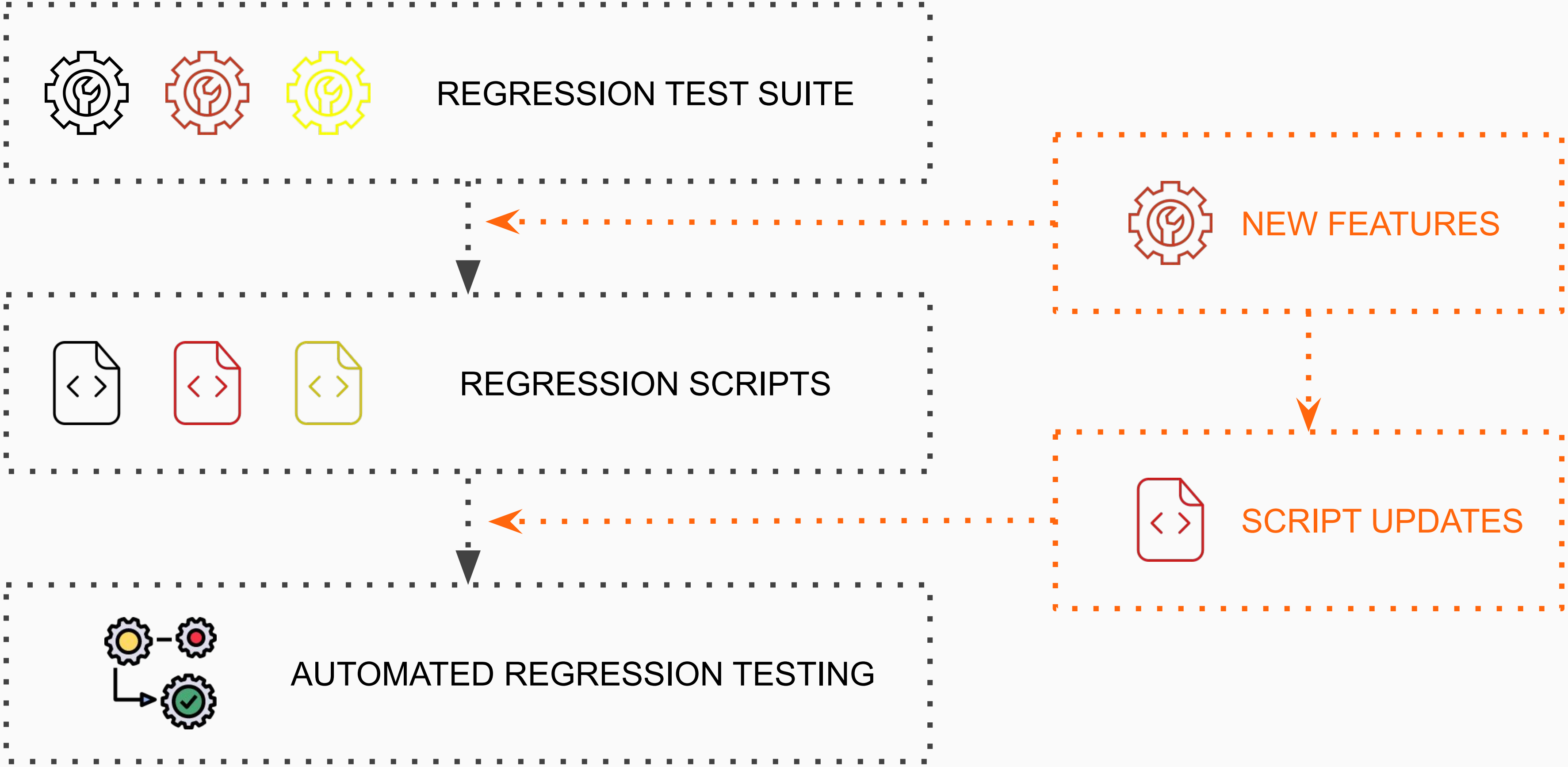


Source: https://www.thoughtworks.com/books/building-evolutionary-architectures

Before make a architectural decision, we need to be confident! How?

Regression Tests

Regression Tests



REGRESSION TEST SUITE

NEW FEATURES

REGRESSION SCRIPTS

SCRIPT UPDATES

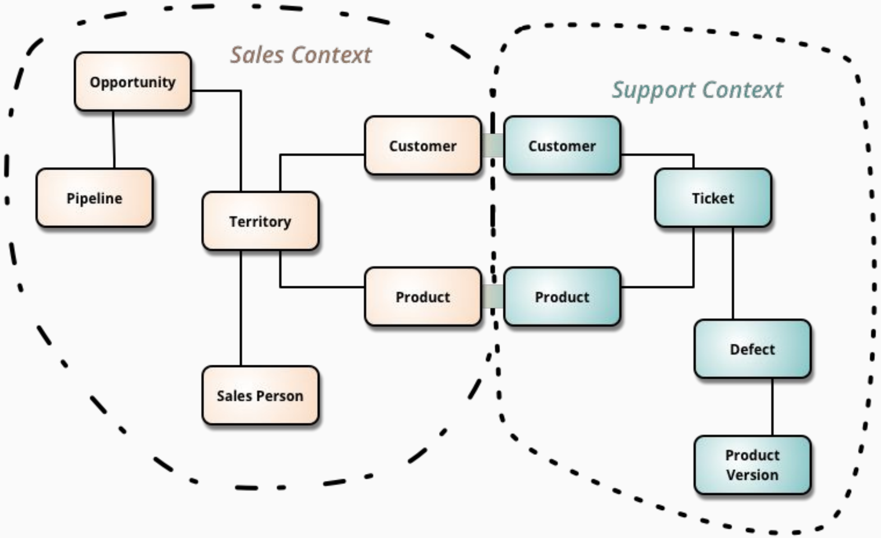
AUTOMATED REGRESSION TESTING

We have tests coverage and now?

Domain Driven Design (DDD)

Domain Driven Design - Bounded Context

Bounded Context is a **central** pattern in Domain-Driven Design. It is the focus of DDD's strategic design section which is all about dealing with **large models** and **teams**. DDD deals with large models by dividing them into different Bounded Contexts and being **explicit** about their interrelationships.



Source: https://martinfowler.com/bliki/BoundedContext.html

We separated the responsibilities and now?

Communication Patterns

REST API
SPACE RANGER
LIGHTYEAR
REST API, EVERYWHERE

Advanced Message Queuing Protocol (AMQP)

Advanced Message Queuing Protocol (AMQP)

- Introducing Message-Driven Programming with Low Level Events
- Fire & Forget principle
- Durable Messages (avoid Circuit Breakers & Retries)
- Better Scale strategies & Application Decoupling
- Fits well for 3rd partners integrations

Advanced Message Queuing Protocol (AMQP) - Thoughts

- Keep track of events otherwise you will have problems
- We used retries to put message in the queue
- Take care of the data into the message (security concerns)

RabbitMQ

GRPC

gRPC

- HTTP Connections are stateful (avoid open & close)
- Static Typed & Well Defined Contracts
- Reduce serialization complexity & improve efficiency
- Easy Peasy to integrate (there is code generation based in .proto)
- Traffic is binary reduce bandwidth usage

gRPC - Thoughts

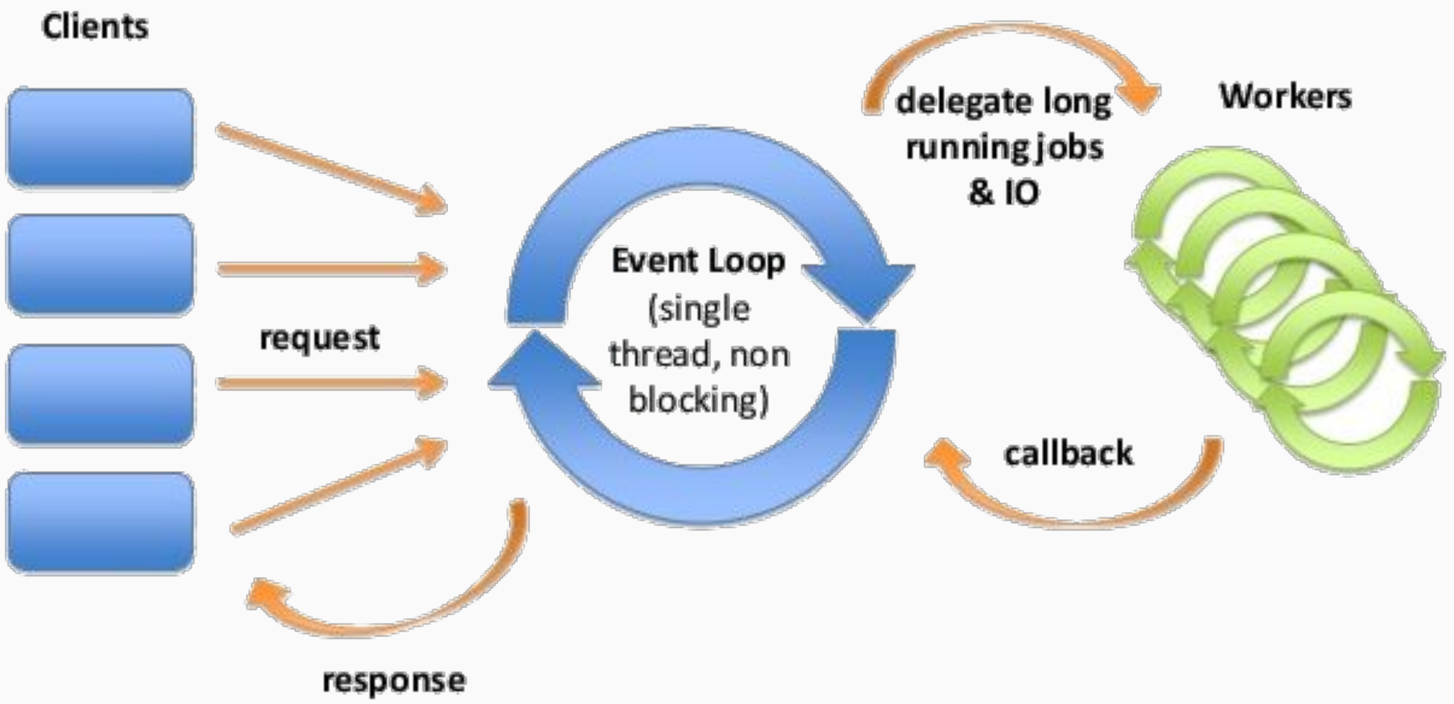
- Load Balancing and Client Load Balancing
- We used in the high throughput solution
- By definition in general we use for internal communication (a.k.a East-West)
- Not for humans, designed by machine communications

We have tests, responsibilities and communication!

It is possible to improve more?

Reactor Pattern & Reactive Programming

Reactor Pattern & Reactive Programming



Reactor Pattern & Reactive Programming

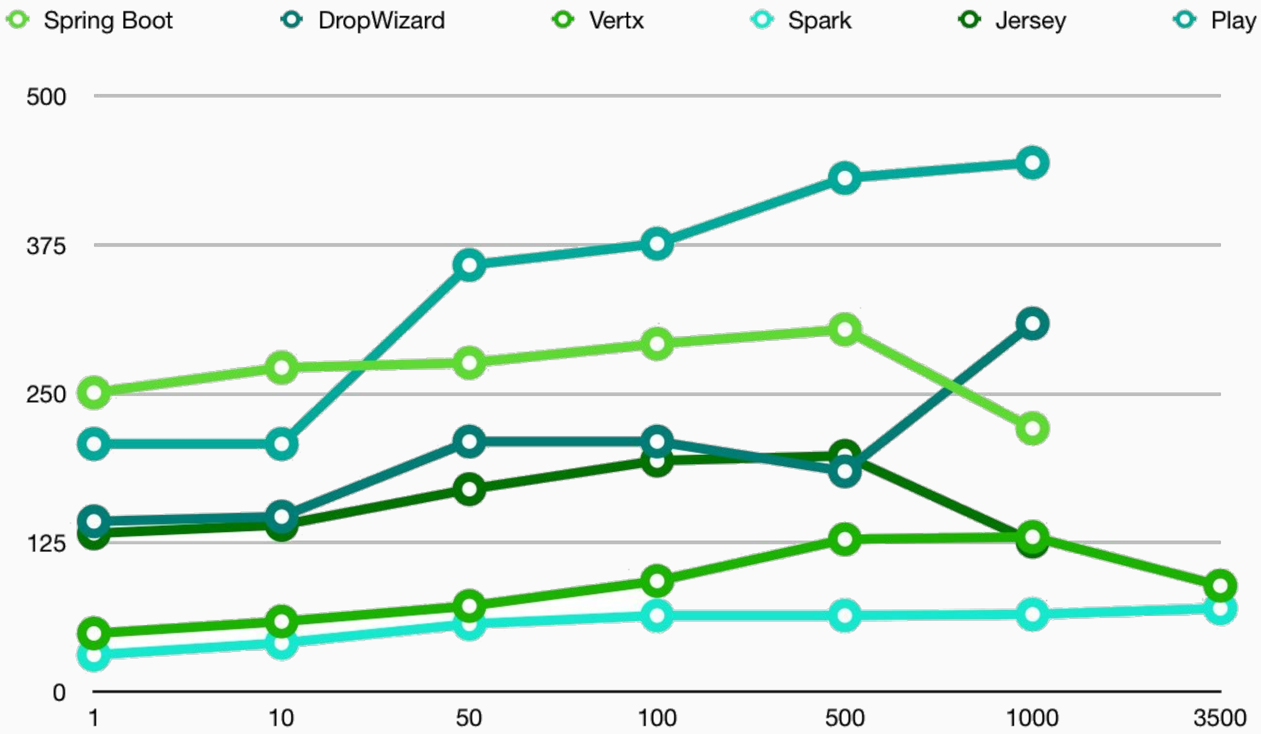
Reactive programming is an asynchronous programming paradigm concerned with data streams and the propagation of change.

- Easier to read, maintain and evolute
- Low memory footprint and high throughput
- Scalability

VERT.X

Source: https://en.wikipedia.org/wiki/Reactive_programming

Reactor Pattern & Reactive Programming



Source: https://craftsmen.nl/memory-usage-6-popular-rest-server-frameworks-compared/

Reactive Programming - Thoughts

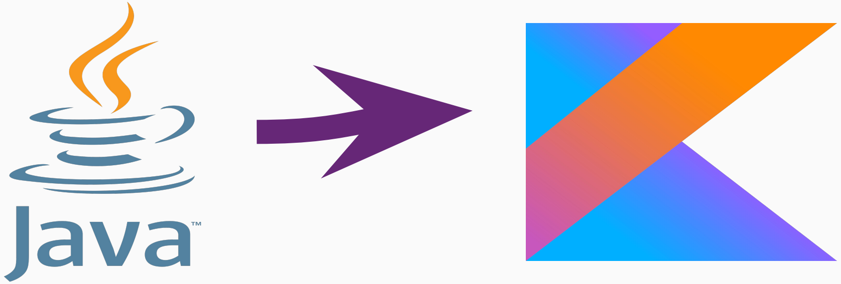
- Never blocks the event loop
- Callback hell
- Don’t look memory, look the event loop size

Language

Language

Kotlin is fully compatible with all Java-based frameworks, which lets you stay on your familiar technology stack while reaping the benefits of a more modern language.

- 40% less code = Easier to read, maintain and evolute
- Fail fast principle = Time to Market



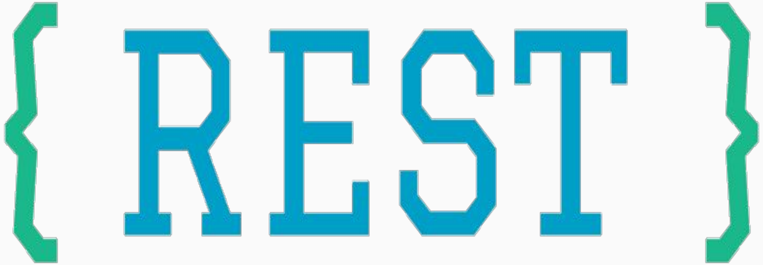
Source: https://kotlinlang.org/docs/reference/server-overview.html

Language - Thoughts

- Expertise in Java Frameworks
- Learning curve, sprint in progress

Before & After

Before

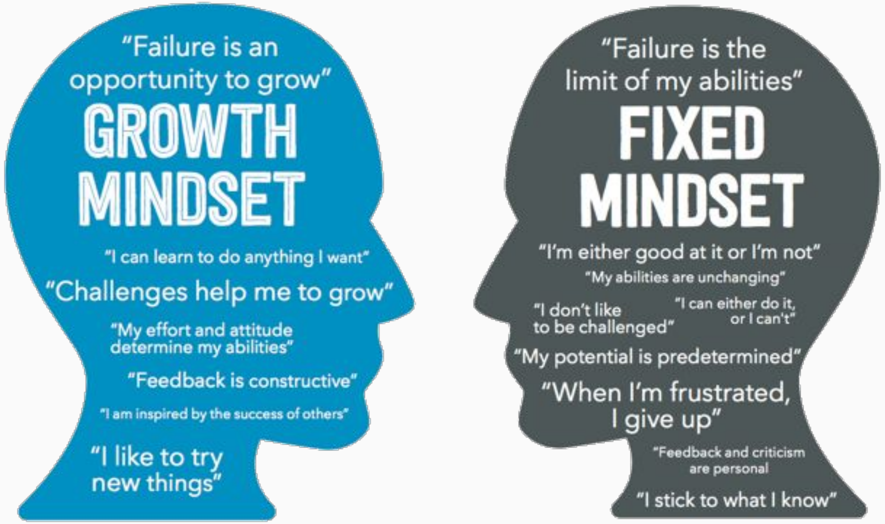


After

OPENSHIFT
sensedia
API PLATFORM
{ REST }
spring®
VERT.X
GRPC
mongoDB
RabbitMQ
Java™
redis
μ
MICRONAUT
ORACLE®

Team Mindset

In a fixed mindset, people believe their **basic qualities**, like their intelligence or talent, are simply **fixed traits**. Contrarily, in a growth mindset, people believe that their most basic abilities can be **developed** through dedication and hard work.


"Failure is an opportunity to grow"
GROWTH MINDSET
"I can learn to do anything I want"
"Challenges help me to grow"
"My effort and attitude determine my abilities"
"Feedback is constructive"
"I am inspired by the success of others"
"I like to try new things"
"Failure is the limit of my abilities"
FIXED MINDSET
"I'm either good at it or I'm not"
"My abilities are unchanging"
"I don't like to be challenged"
"I can either do it, or I can't"
"My potential is predetermined"
"When I'm frustrated, I give up"
"Feedback and criticism are personal
"I stick to what I know"

Source: https://mindsetonline.com/whatisit/about/

Thanks a million!

Questions?

/larchanjo

/luram-archanjo