italiaonline

Microservice: Yes or no?

Marco Ghisellini – Assago, 28/06/2019

I nostri brand

LIBERO VIRGILIO PgCasa.it SUPEREVA DiLei SìViaggia QuiFinanza Buonissimo PagineGialle PagineBianche TuttoCittà

A future forecast ….

Microservice - Definition

Microservices are an architectural approach to building applications.

As an architectural framework, microservices are distributed and loosely coupled,

so one team’s changes won’t break the entire app.

The benefit to using microservices is that development teams are able

to rapidly build new components of apps to meet changing business needs.

From RedHat

Microservice – Very Nerd Definition!!

Building applications as suite of services.

As well as the fact that services are independently deployable and scalable,

each service also provides a firm module boundary,

even allowing different services to be written in different programming languages.

They can also be managed by different teams.

Micro Services - Martin Fowler

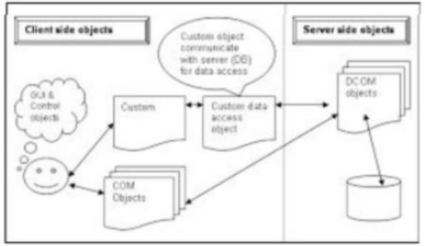
Distributed solution … doesn’t it already exists ??

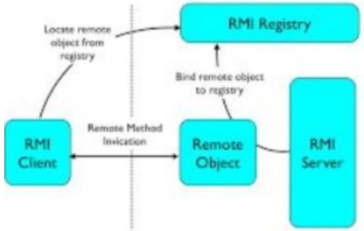
Corba / Dcom / Java Rmi

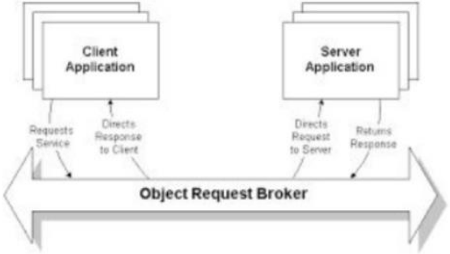
Corba: **C**ommon **O**bject **R**equest **B**roker **A**rchitecture

Dcom: **D**istributed **C**omponent **O**bject **M**odel;

Java Rmi: **R**emote **M**ethod **I**nvokation.





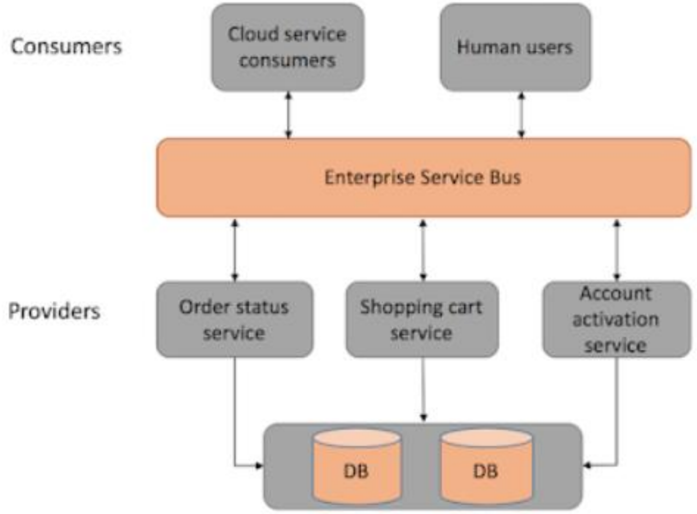


They all work(ed) well, although they shared Object and not Service.

SOA

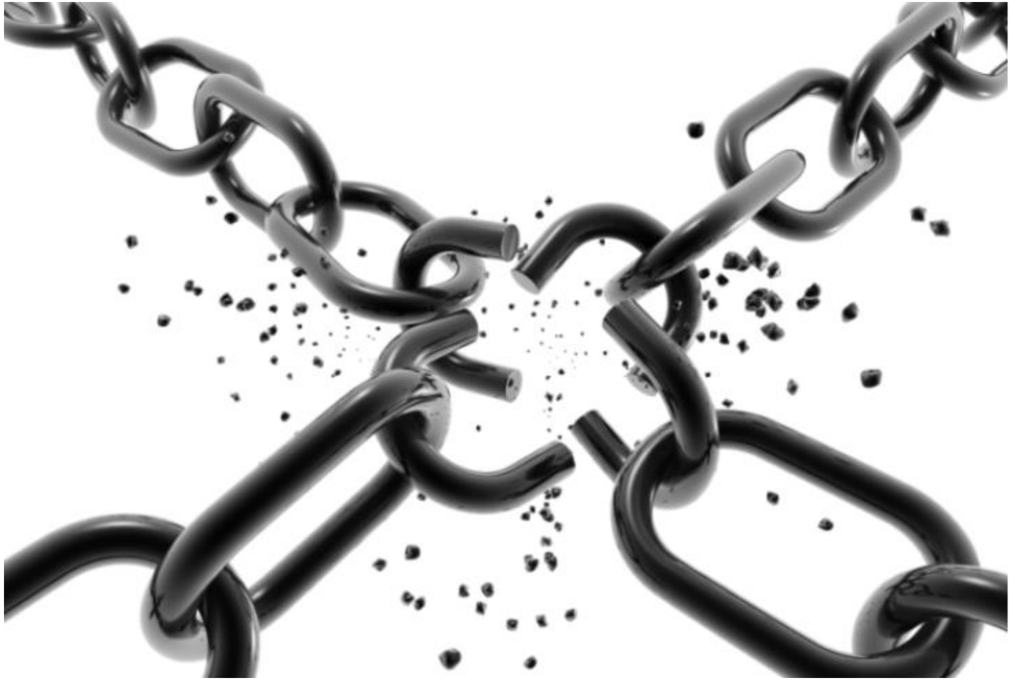
Con: **S**ervice **O**rientied **A**rchitecture.

- Soap/Rest protocol;
- Enterprise Service Bus;
- Shared Services, not an Objects.



…Monolithic application usually developed with Soap language, Xml, Message Queue;

Microservice – loosely coupled

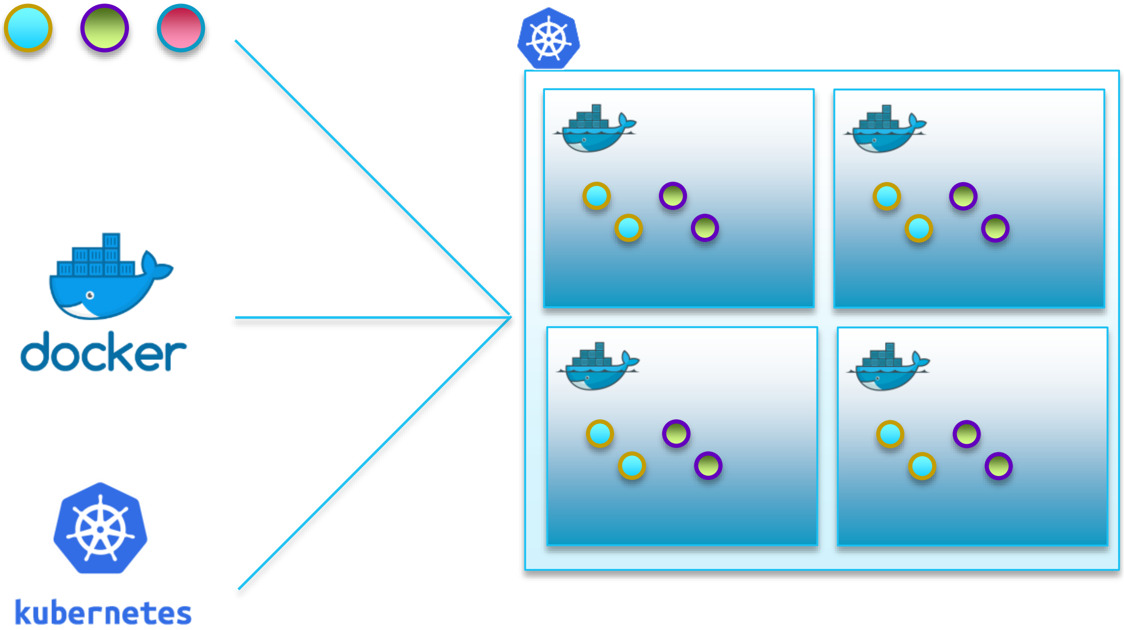


Microservice – losing coupling

Services

Container

Orchestrator



italiaonline

PgCasa.it

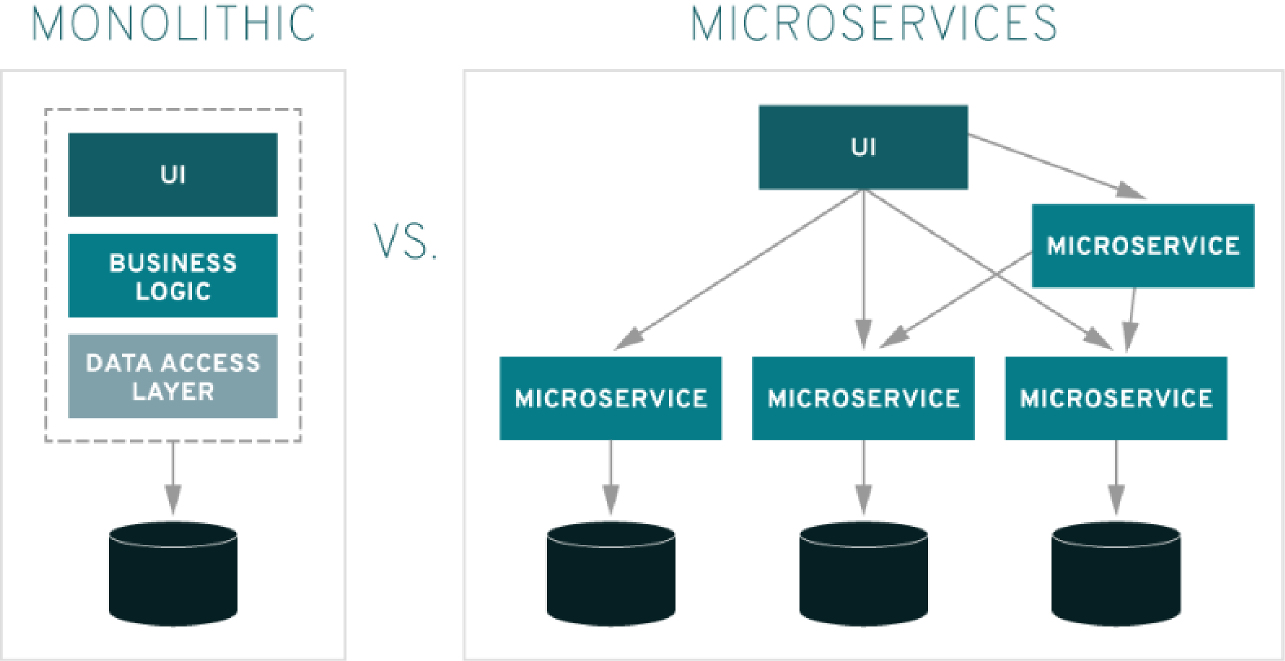
L’home service
su misura

Monoliths vs Microservices

I nostri brand

LIBERO VIRGILIO PgCasa.it SUPEREVA DiLei SìViaggia QuiFinanza Buonissimo Pagine Gialle Pagine Bianche Tutto Città

Monolithic vs Microservice - Schema



From RedHat

Monoliths - Pros

- Natural starting point
  - Easier to get started and deliver value
- Simpler build and deployment
- Simpler scalability
- Simpler security
- Low latency
  - Intra-process communication
- Simpler testing
- Simpler logging and monitoring
- Simpler data and database management
- Simpler transaction management

Monoliths - Cons

- Large code base
- Simple change requires whole app to be redeployed
- Increased complexity as functionality is coupled together
- Single type of database doesn’t meet all requirements
- Tend to get difficult to work with over time
- Huge resource requirements
- Reduced agility over time
- Coarse-grain transactions

Microservices - Pros

- Smaller manageable functional units
  - Multiple smaller code bases
- Single responsibility per service
  - Single service provides single functionality
  - Clearer separation of concerns
- Easier on-boarding process
- Independently scalable services Independently deployable
- Polyglot technologies & frameworks as applicable
- Frequent functionality releases

Microservices - Cons

- Distributed System Architecture
  - Design, Development, Deployment, CAP theorem
- Handle Increased orchestration
- Troubleshooting challenges
- Data consistency issues
  - Eventual consistency
  - Compensatory & reconciliatory procedures
- Increased latency due to remote calls
- Distributed Configuration Management
- Organizational Maturity
- Architectural complexity

Microservices – Problem Solution

- DevOps
- Use supporting platform
  - Cloud
  - Containers
- Architect
  - Low coupling, High Cohesion *

* Classes with high specializations without a strong dependencies each others.

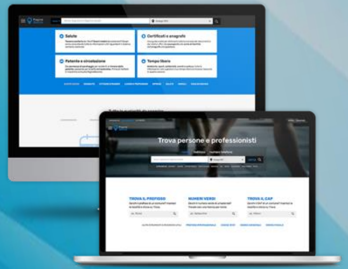
italiaonline

Microservices

Quick Start (only an overview)

Pagine Bianche

Un tesoro di
10 milioni di abbonati

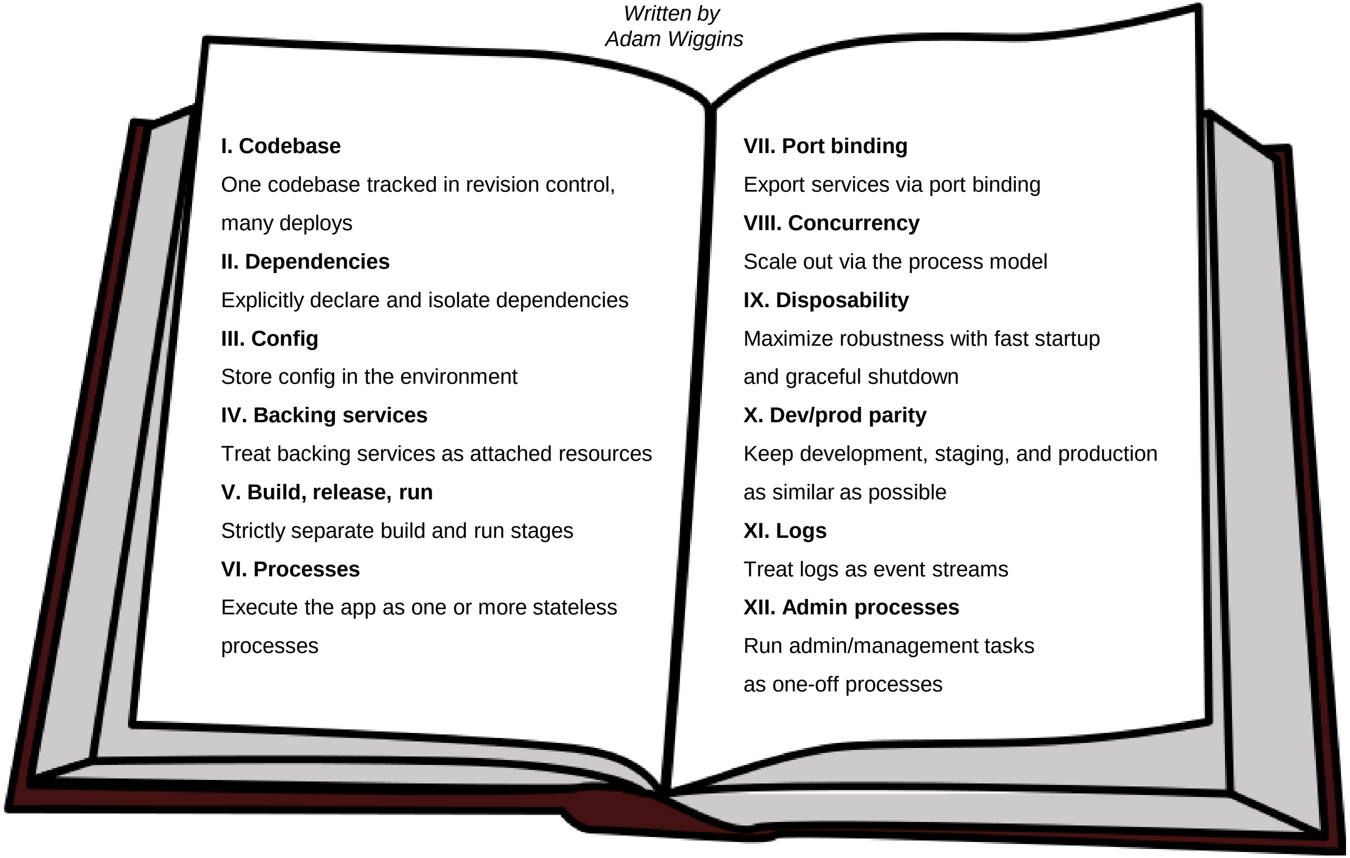


I nostri brand

LIBERO VIRGILIO PgCasa.it SUPEREVA DiLei SìViaggia QuiFinanza Buonissimo Pagine Gialle Pagine Bianche Tutto Città

The Twelve-Factor App

The Twelve-Factor App - List



Written by
Adam Wiggins

I. Codebase

One codebase tracked in revision control, many deploys

II. Dependencies

Explicitly declare and isolate dependencies

III. Config

Store config in the environment

IV. Backing services

Treat backing services as attached resources

V. Build, release, run

Strictly separate build and run stages

VI. Processes

Execute the app as one or more stateless processes

VII. Port binding

Export services via port binding

VIII. Concurrency

Scale out via the process model

IX. Disposability

Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity

Keep development, staging, and production as similar as possible

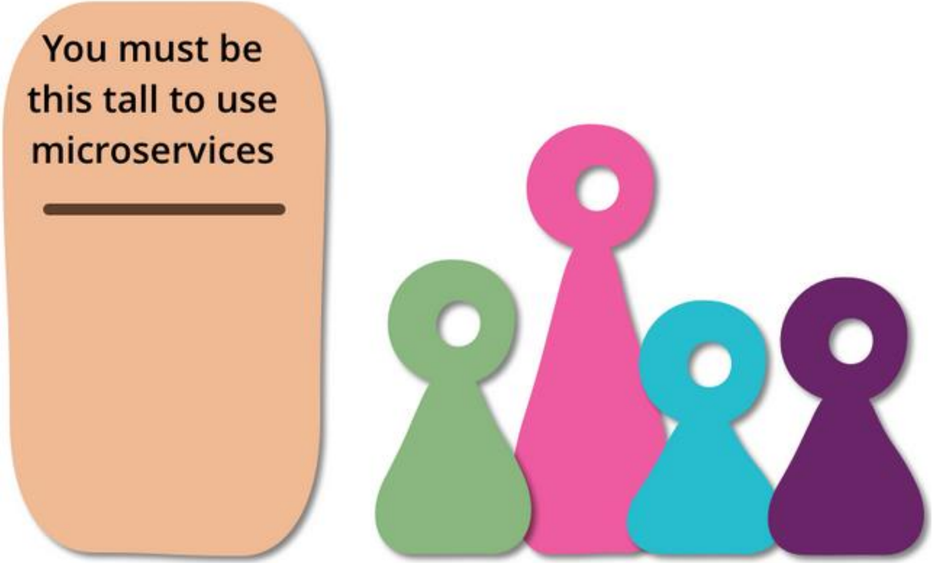
XI. Logs

Treat logs as event streams

XII. Admin processes

Run admin/management tasks as one-off processes

Should I use or should I stay away?



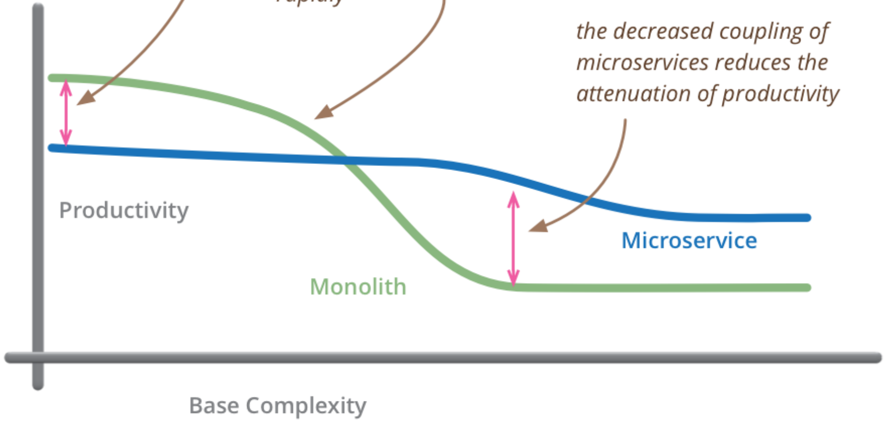
https://martinfowler.com/bliki/MicroservicePrerequisites.html

Learning curve

for less-complex systems, the extra baggage required to manage microservices reduces productivity

as complexity kicks in, productivity starts falling rapidly

the decreased coupling of microservices reduces the attenuation of productivity



but remember the skill of the team will outweigh any monolith/microservice choice

https://www.martinfowler.com/bliki/MicroservicePremium.html

Use it when …

- **DevOps**
  Organizations with strong DevOps culture.

- **Splitting in small pieces**
  Microservices allow to break a very big monolith in small pieces.

- **Agility**
  you can update, test and deploy each micro service in an independent way.

- **Independent Scalability**
  The services can scale at different rates.

- **Isolated Failure**
  Microservices can build appropriate failover mechanisms.

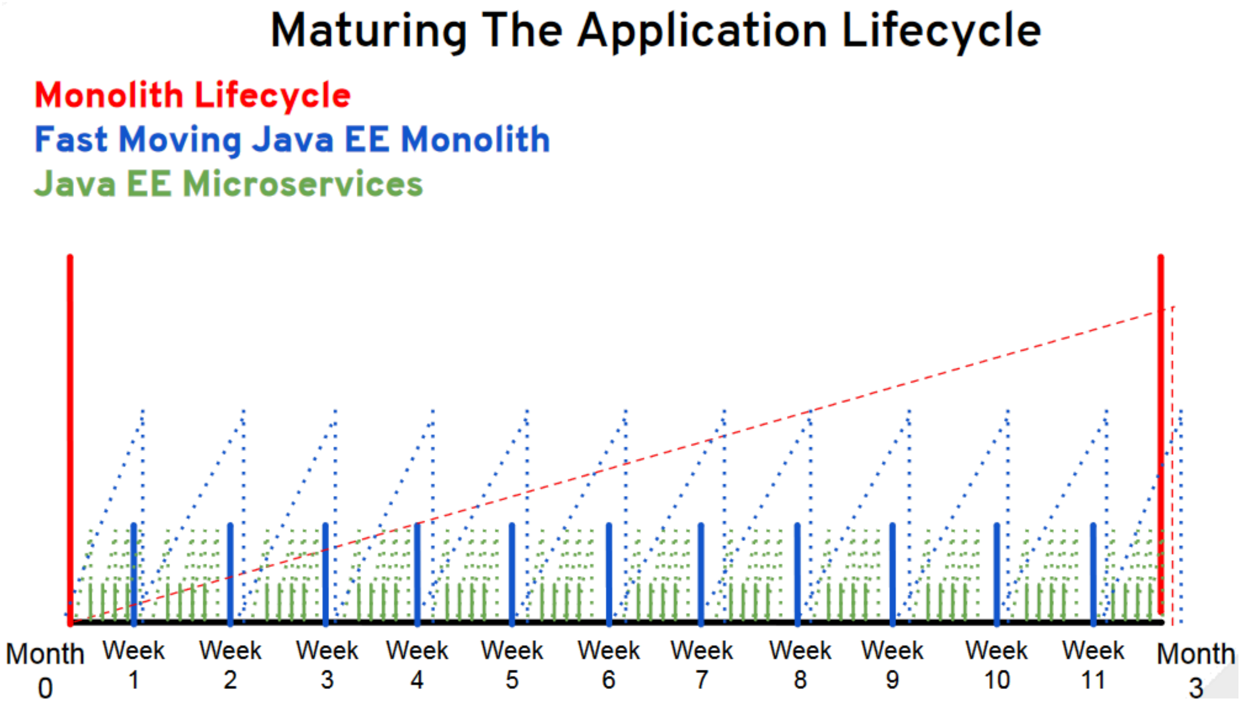
- **External Dependencies**
  Simplify Interactions with External Dependencies: Façade Pattern.

- **The Freedom to Choose the Right Tech for the Job**
  Different languages for different purposes.

- **CI/CD**
  Team has a mature CI/CD pipelines.

- Multiple Rates of Change …

Delivery Frequency



Maturing The Application Lifecycle

Ugo Landini, Solution Architect di Red Hat

italiaonline

Successful Cases

Tutto Città

Tutte le strade portano qui

I nostri brand

LIBERO VIRGILIO PgCasa.it SUPEREVA DiLei SìViaggia QuiFinanza Buonissimo Pagine Gialle Pagine Bianche Tutto Città

Netflix

NETFLIX

In 2009, when Netflix started to migrate a monolithic infrastructure to a microservices one,
the term ‘microservices’ didn’t even exist anywhere.

Working on a monolithic architecture was proving to be difficult for the company with every passing day and the service would have outages whenever Amazon’s servers went down.

After moving to microservices, Netflix’s engineers could deploy **thousands** of different code sections **every day**.

Forced to write their own entire platform on the cloud, the company has been pretty open about what they learned with the move, and they have even managed to open source many of the components and tools to help the community.

Though Netflix hasn’t put up their entire platform code on Github, which could also help new companies.
Overall, moving to microservices was incredibly beneficial for Netflix, and it has led to **decrease their application’s downtime** to a large extent.

Amazon

Back when Amazon was operating on a monolithic architecture, it was difficult for the company to predict and manage the fluctuating website traffic.

In fact, the company was losing a lot of money as most of the server capacity was being wasted in downtime.

Back in 2001, Amazon's application was one big monolith.

Even though it was divided into different tiers and those tiers had different components, they were tightly coupled with each other, and they behaved like a monolith.

The main focus of the developers was to simplify the entire process, and for that, they pulled out functional units from the code and wrapped them in a web service interface.

For instance, there was a **separate microservice was calculated the total tax at check out**.

The company's move to Amazon Web Services (AWS) cloud for microservices helped them **scale up or down according to the traffic**, handle outages better, and save costs as well.

Since microservices allows to deploy code continuously, engineers at Amazon now deploy **code every 11.7 seconds**.

Uber

Just like any other startup, Uber too started with a monolithic architecture for their application.

At that point, it seemed cleaner to go with a monolithic core since the company was just operating in San Francisco and only offered the UberBLACK option to users.

But as the ride-sharing startup grew multi-fold, they decided to follow the path of other companies like Amazon, Netflix, and Twitter and moved to microservices.

The biggest advantage of migration was, of course, the fact that each microservice can have **its own language and framework**.

Now, with more than 1300 microservices, Uber focuses on applying microservices patterns that can **improve scalability and reliability of the application**.

With so many microservices, a big focus is also on identifying the ones that are old and not in use anymore.

That is why the team always ensures to **decommission the old ones** regularly.

References

Article

- https://www.slideshare.net/EranStiller/to-microservice-or-not-to-microservice
- https://www.slideshare.net/eduardtomas/microservices-yes-or-not
- https://martinfowler.com/bliki/MicroservicePrerequisites.html
- https://12factor.net/
- https://content.pivotal.io/blog/should-that-be-a-microservice-keep-these-six-factors-in-mind

Let’s get some code!

- https://techannotation.wordpress.com/2019/01/04/hand-on-kubernetes-world-installation/
- https://techannotation.wordpress.com/2019/01/04/hand-on-kubernetes-world-booking-example/
- https://techannotation.wordpress.com/2019/01/17/kubernetes-gets-married-to-istio/
- https://techannotation.wordpress.com/2019/01/24/istio-telemetry-applications-installed/
- https://techannotation.wordpress.com/2019/05/13/microservice-every-service-with-its-own-database/
- https://techannotation.wordpress.com/2019/06/03/docker-swarm-containers-distribution/

italiaonline

Any Question?

marco.ghisellini@Italiaonline.it

I nostri brand

LIBERO V:RGILIO PgCasa.it SUPEREVA Di·Lei SìViaggia QuiFinanza Buonissimo Pagine Gialle Pagine Bianche Tutto Città

Di·Lei

Digitale, femminile, singolare

italiaonline
Nerd Happy Hour

Giovedì 4 luglio dalle 16:00 alle 17:00 *, in contemporanea dalle sedi di Pisa e Milano (e dalle altri sedi che vorranno partecipare) , un nuovo format per il nostro incontro settimanale*

PANEL- Linguaggi di programmazione in azienda: from start to end

"Una tavola rotonda, moderata, **aperta a tutti** in cui, grazie anche al supporto di alcuni nostri colleghi,
faremo una overview dei linguaggi di programmazione maggiormente utilizzati in azienda affrontandone pregi, difetti e punti di forza"

Moderatore: Francesca Bonfanti
Relatori: Antonino Mistretta, Marco Ghisellini, Riccardo Marangone

I nostri brand

LIBERO V:RGILIO PgCasa.it SUPEREVA DiLei SiViaggia QuiFinanza Buonissimo Pagine Gialle Pagine Bianche Tutto Città