Fabrice Clement

When the architect is waterfalling...

- *Just enough architecture*
- *Emergent architecture*
- *Sacrificial Architecture*
- *Who needs an architect ?*
- *Nobody at Google has the title of Architect*

devops
Terme de recherche

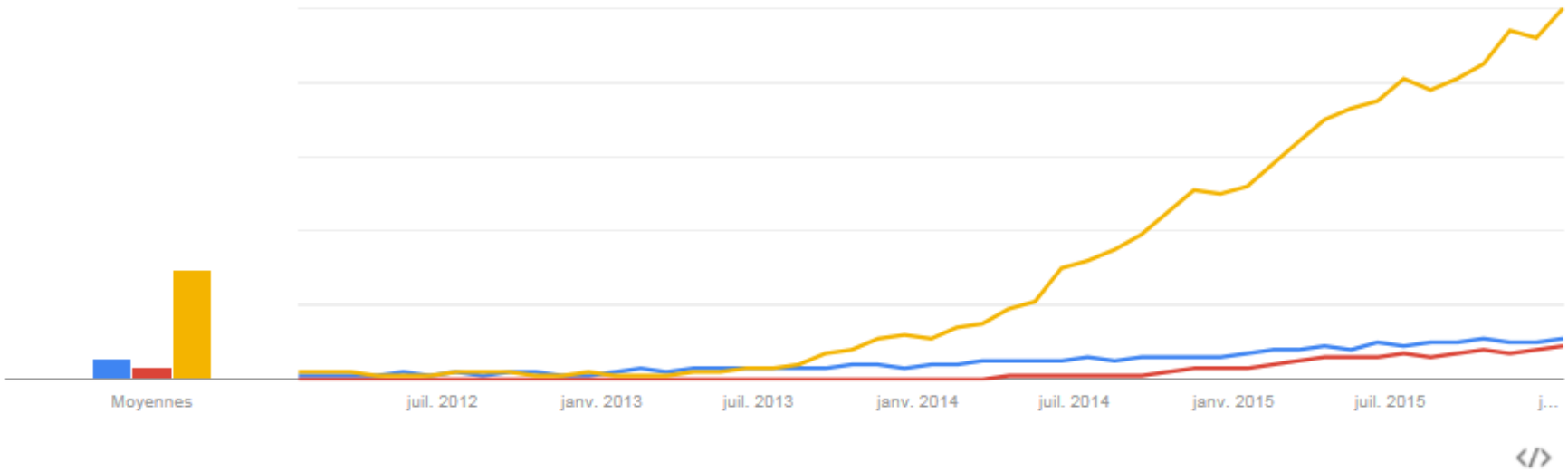
microservices
Terme de recherche

docker
Terme de recherche

+ Ajouter un terme

Évolution de l'intérêt pour cette recherche

Comparer à la catégorie | Titres des actualités | Prévisions



“Systems tend to expand to fill the known universe” ***(1)***

hello-world.tgz **200 Go**

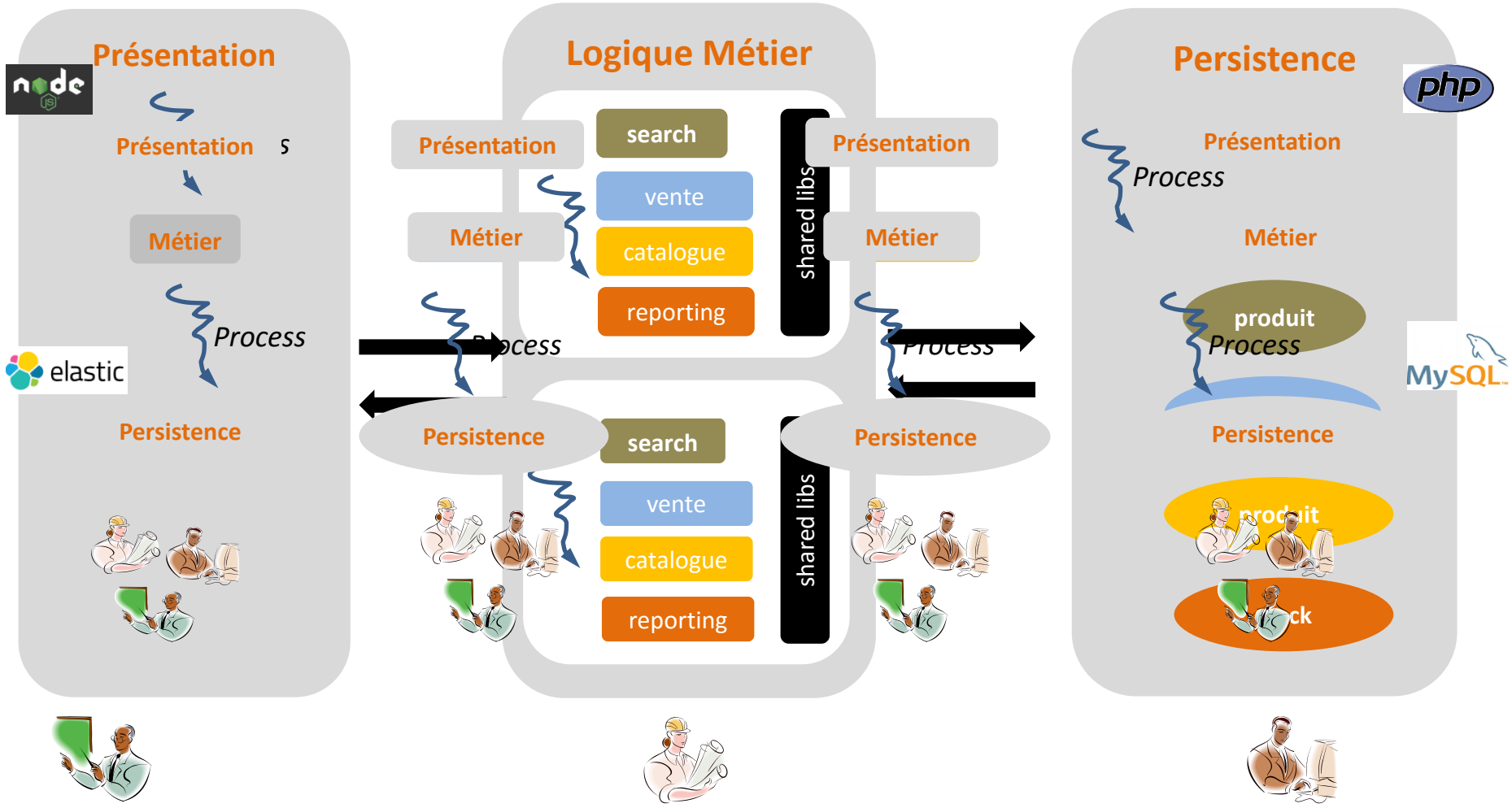
« Tout logiciel gagne des fonctionnalités

jusqu'à être capable de lire des mails.

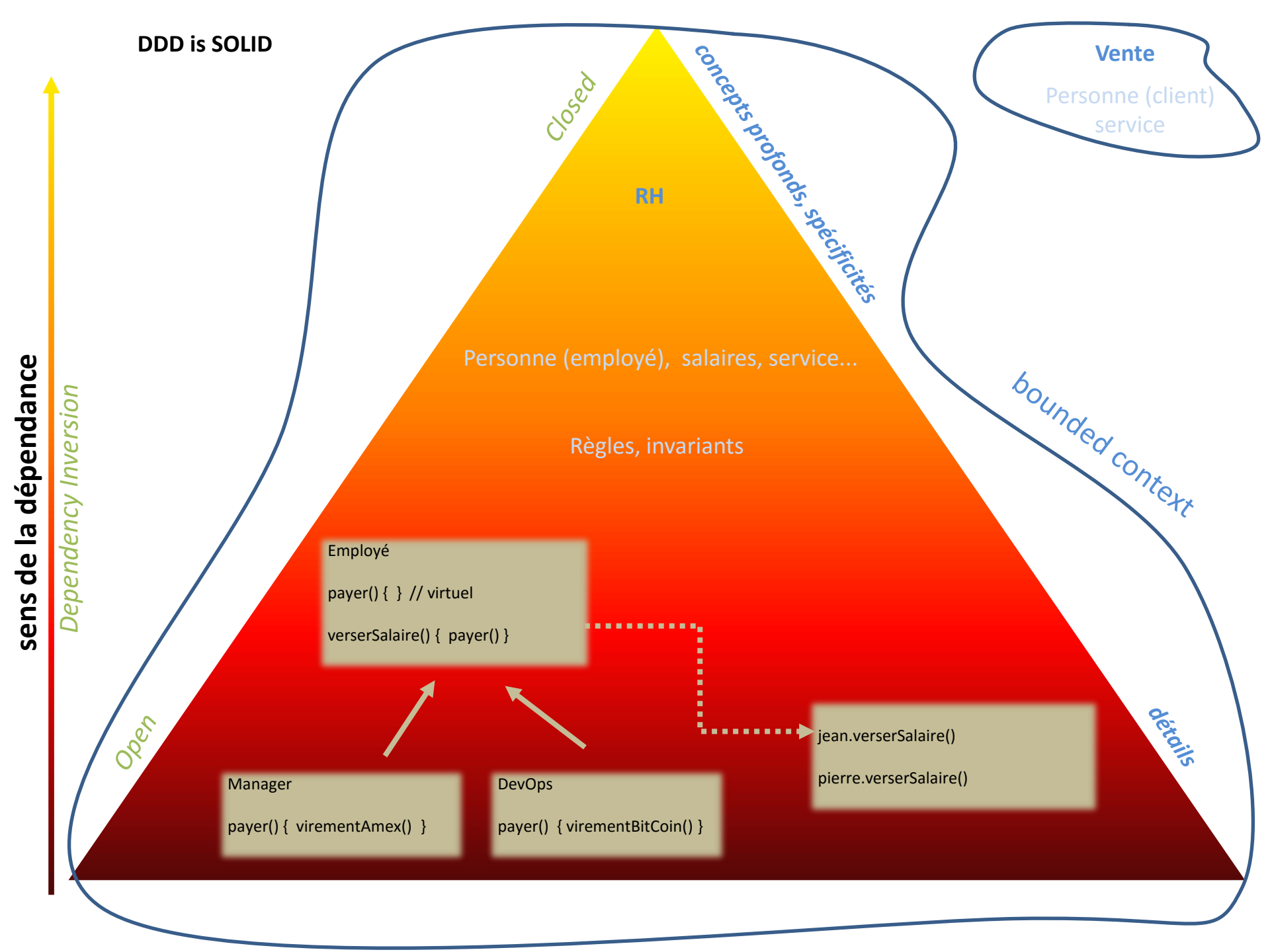
Ceux qui n'en sont pas capables sont remplacés par d'autres qui le sont » ***(2)***

1 : John Gall
2 : Jamie Zawinski

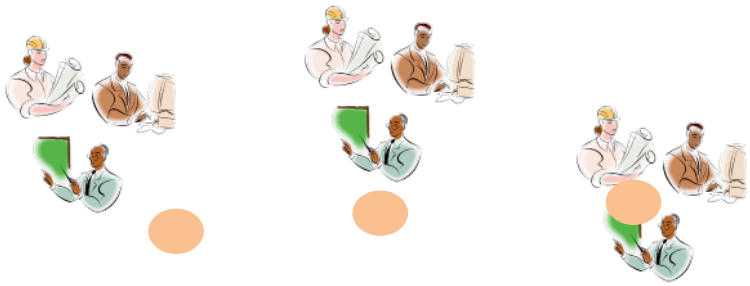
Architecture Micro-services : priorité aux Couches Techniques

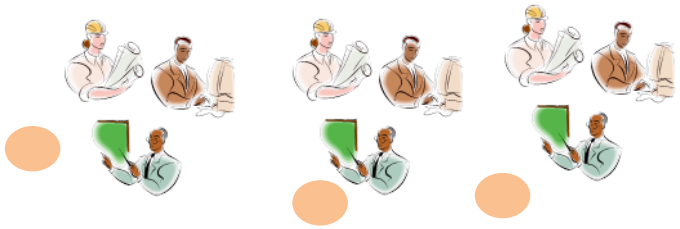


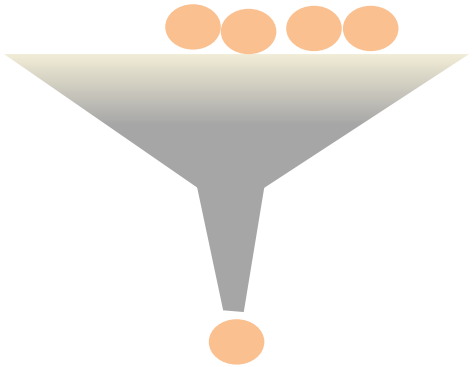
Modèle (Granularité)	Drivers	Prérequis	Conséquences
Monolithe (mono grained)	Performance Vision produit v1	Mono techno Charge compatible	Intégration simple Résilience = Robustesse Simplicité opérationnelle Concurrence simple
SOA (coarse grained)			
Microservices (Fine Grained)	PolyTechnos Evolutions autonomes Vision service Scaling (online) users	Maitrise des systèmes distribués Modularité fonctionnelle	Complexité opérationnelle Intégration / Test plus complexe Résilience = Anti-Fragile

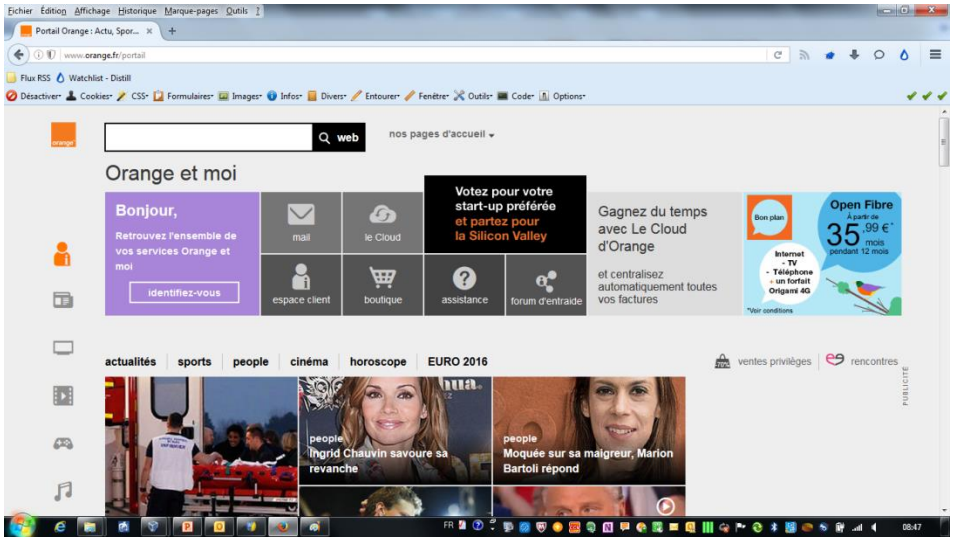

DDD is SOLID
sens de la dépendance
Dependency Inversion
Closed
Open
concepts profonds, spécificités
détails
bounded context
Vente
Personne (client)
service
RH
Personne (employé), salaires, service...
Règles, invariants
Employé
payer() { } // virtuel
verserSalaire() { payer() }
Manager
payer() { virementAmex() }
DevOps
payer() { virementBitCoin() }
jean.verserSalaire()
pierre.verserSalaire()

Parallèle « changes »









Consistency Analysis in Bloom: a CALM and Collected Approach

Peter Alvaro Neil Conway Joseph M. Hellerstein
William R. Marczak

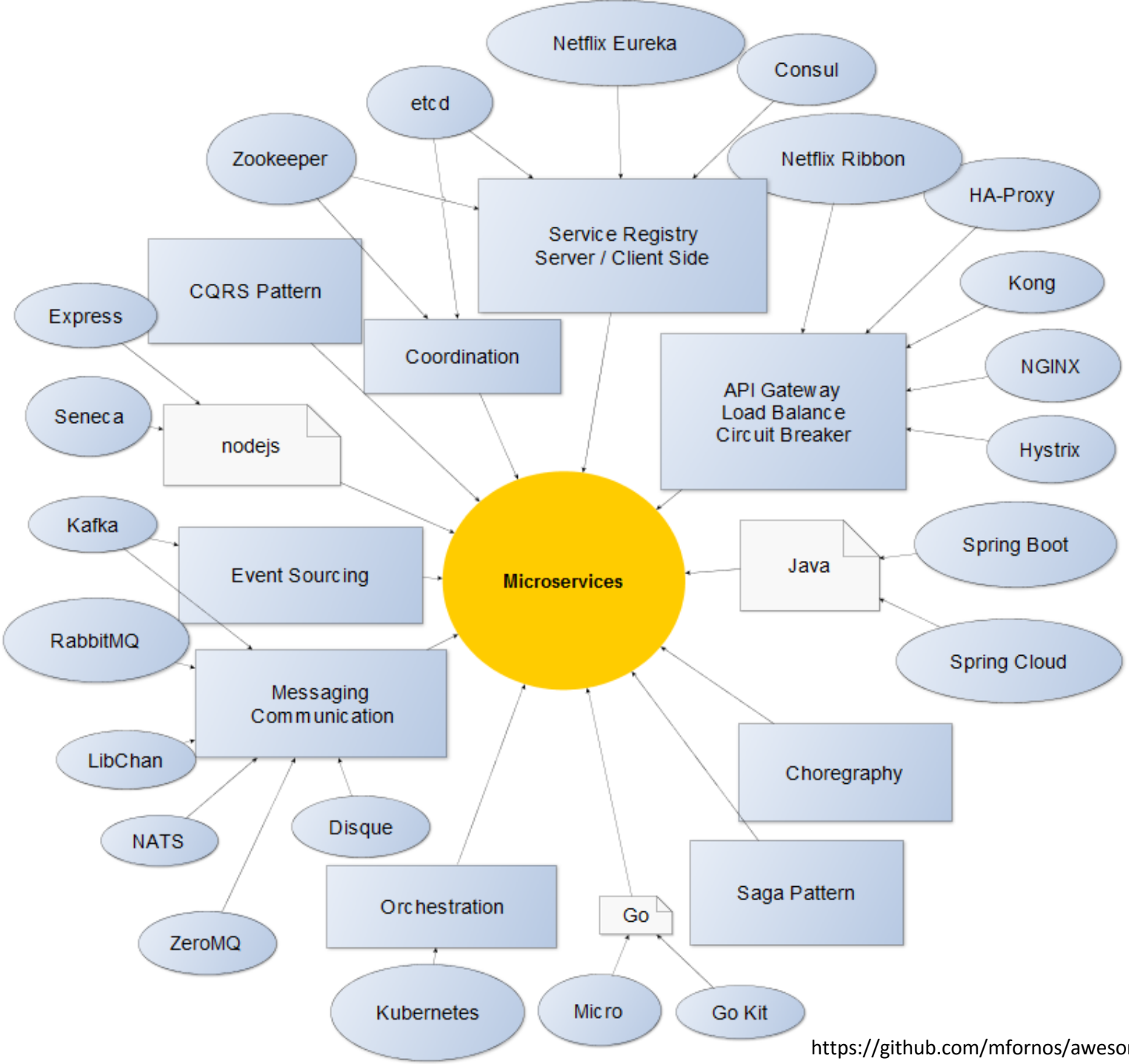
ABSTRACT

Distributed programming has become a topic of widespread interest, and many programmers now wrestle with tradeoffs between data shield programmers from much of the complexity of distributed programming. However, there is a widespread belief that the costs of these mechanisms are too high in many important scenarios where

2. CONSISTENCY AND LOGICAL MONOTONICITY (CALM)

In this section we present a strong connection between distributed consistency and logical monotonicity. This discussion informs the language and analysis tools we develop in subsequent sections.

A key problem in distributed programming is reasoning about con-



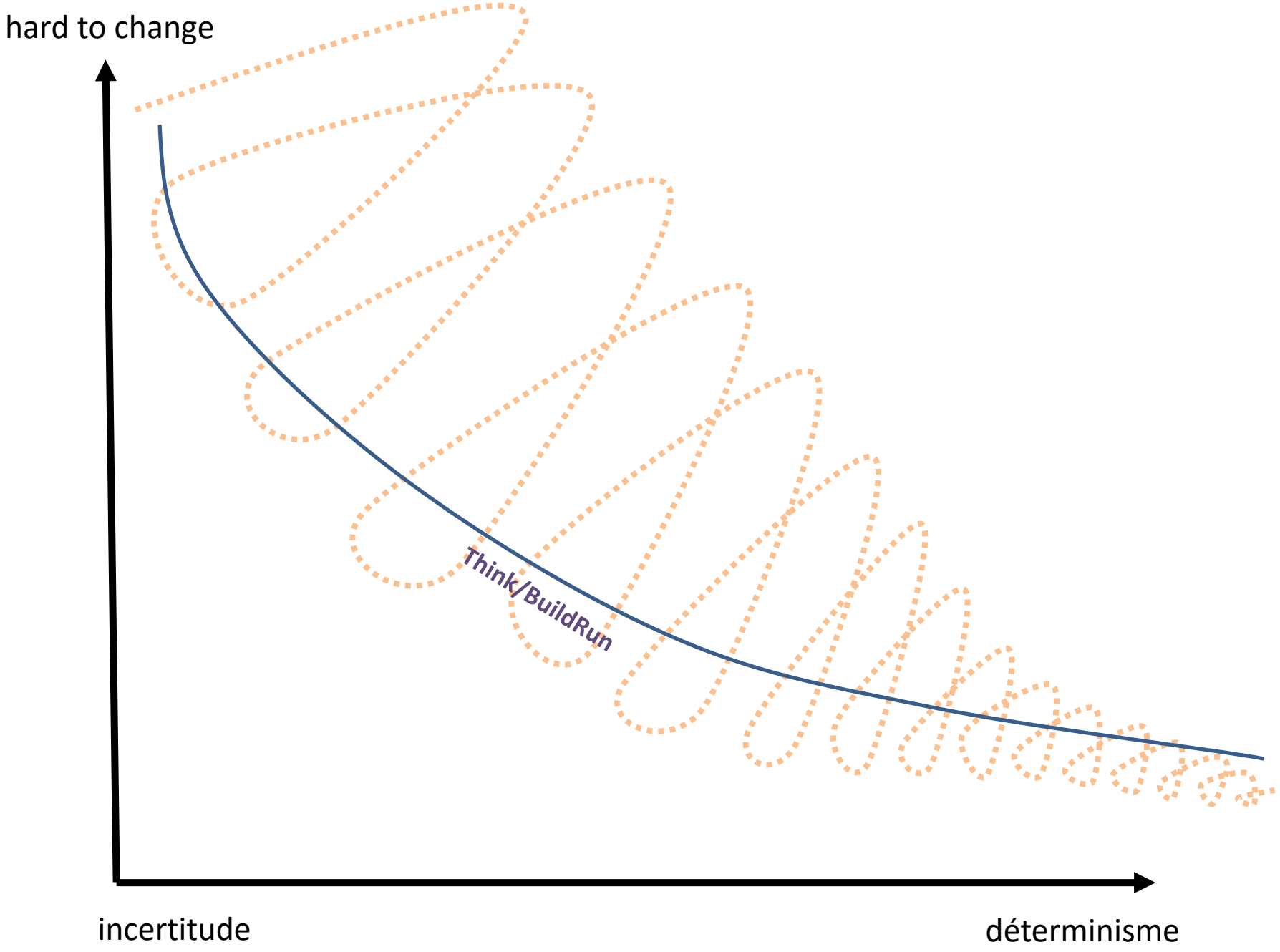
https://github.com/mfornos/awesome-microservices

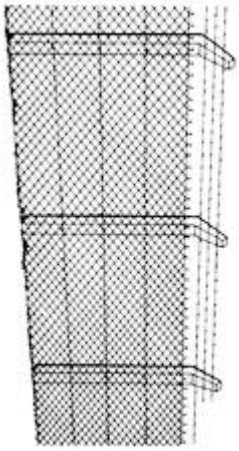
```
package main

import (
   "fmt"
   "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
   fmt.Printf(w, « Go microservice online  »)
}

func main() {
   http.HandleFunc("/", handler)
   http.ListenAndServe(":8080", nil)
}
```


hard to change
Think/BuildRun
incertitude
déterminisme




Alterner le lead
Cultiver le respect
Partager l’engagement


Just married!