

Mathieu ANCELIN
Alexandre DELEGUE
Jonathan GOLDSMITH



#RaspberryCluster #Zombies

@TrevorReznik @chanksleroux

Mathieu ANCELIN

- Développeur @SERLI
- Scala, Java, web & OSS
 - ReactiveCouchbase, Weld-OSGi, etc ...
- Poitou-Charentes JUG
- Membre de l'expert group MVC 1.0
- @TrevorReznik



Alexandre DELEGUE

- Développeur @ SERLI
- Java
- Scala
- Web
- spring, play, ...
- @chanksleroux



SERLI

- Société de conseil et d'ingénierie du
- 70 personnes
- Java, cloud, mobilité
- Contribution à des projets O_{JUG} sonarsource
- 10% de la force de travail sur l'OSS
- Membre de l'EG JSR-346
- Membre de l'OSGi Alliance
- www.serli.com @SerliFr

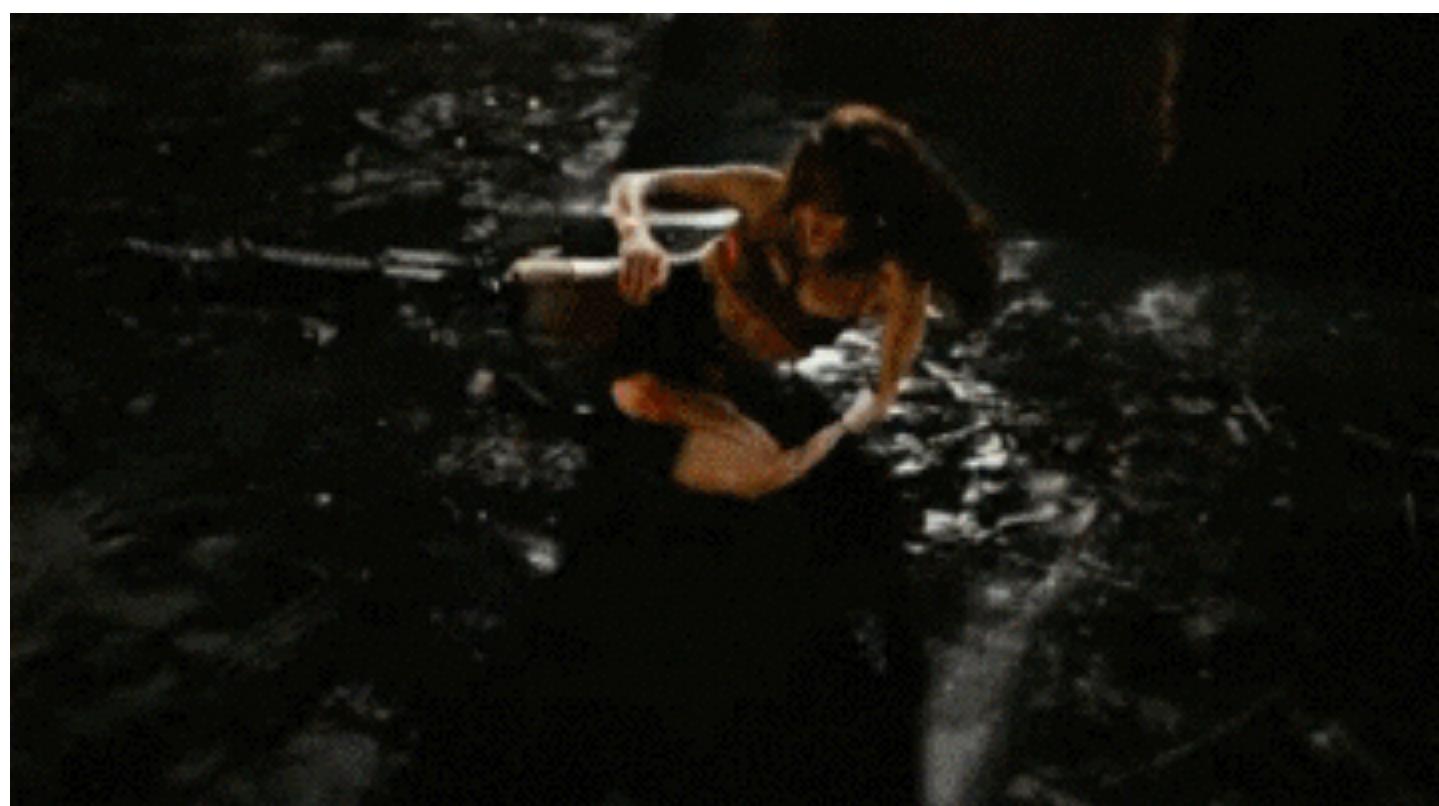


France
DEVOXX
DEV
JUG

Les technologies présentées sont inspirées de technologies réelles

Les applications qui en découlent sont fictives

Toute ressemblance avec des applications existantes n'est que fortuite



www.amazing.com

France

TM
DEVVOXX
□

Amazing Store Hello test 0 items Logout

Search a product Search

Mitraillette années 30 canon court
Mitraillette années 30. S ...
3599.0 €
[Add to cart](#)

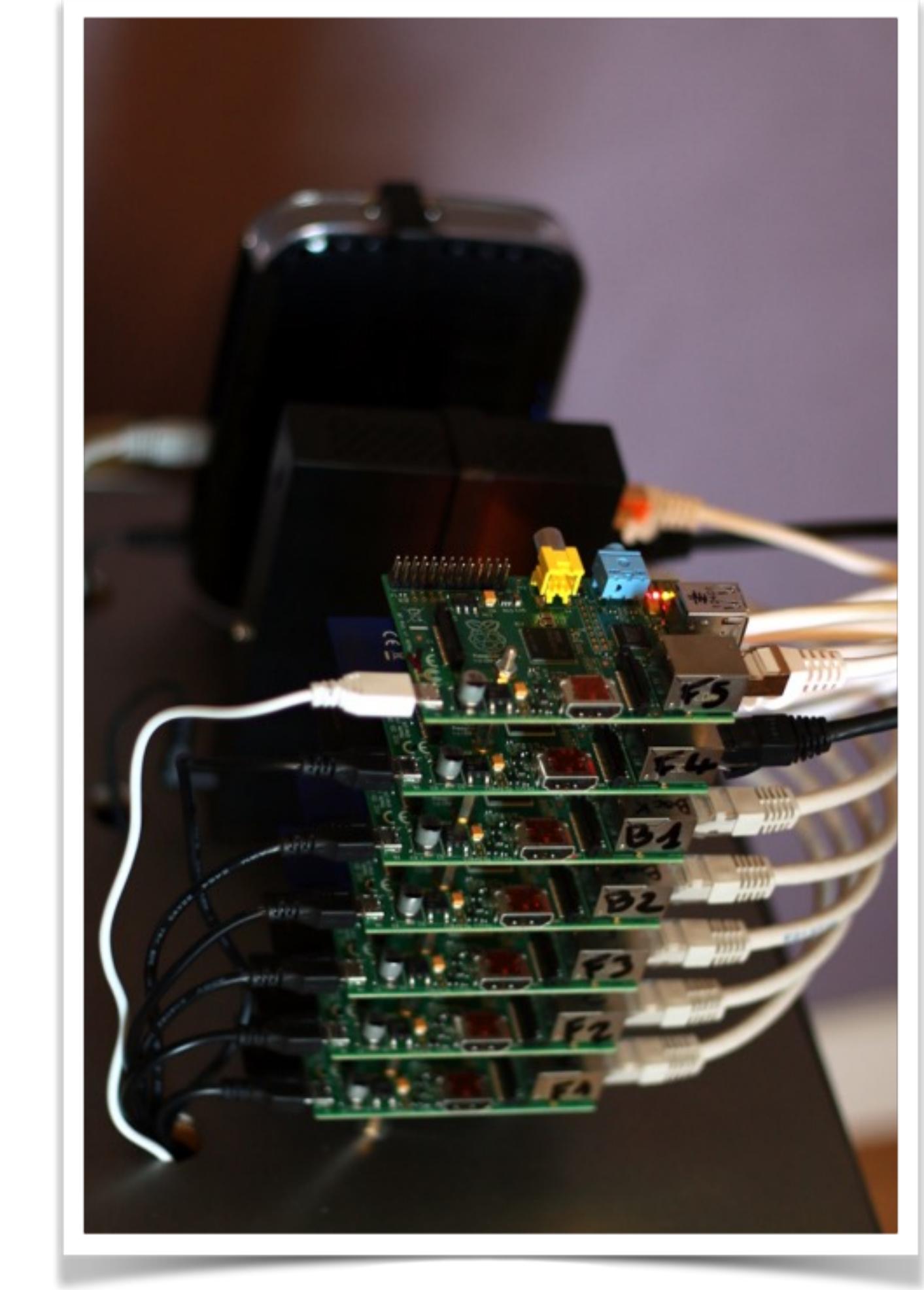
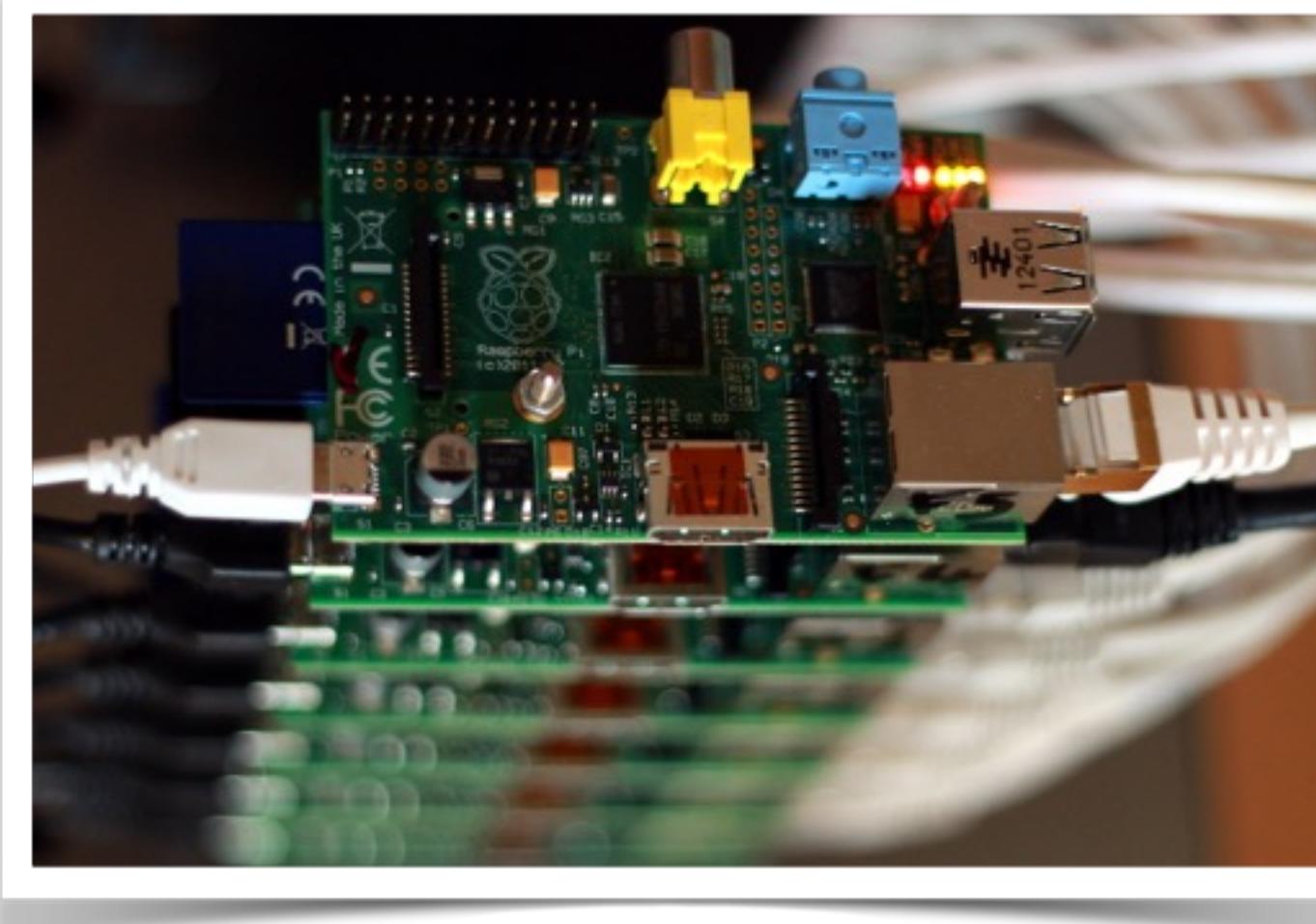
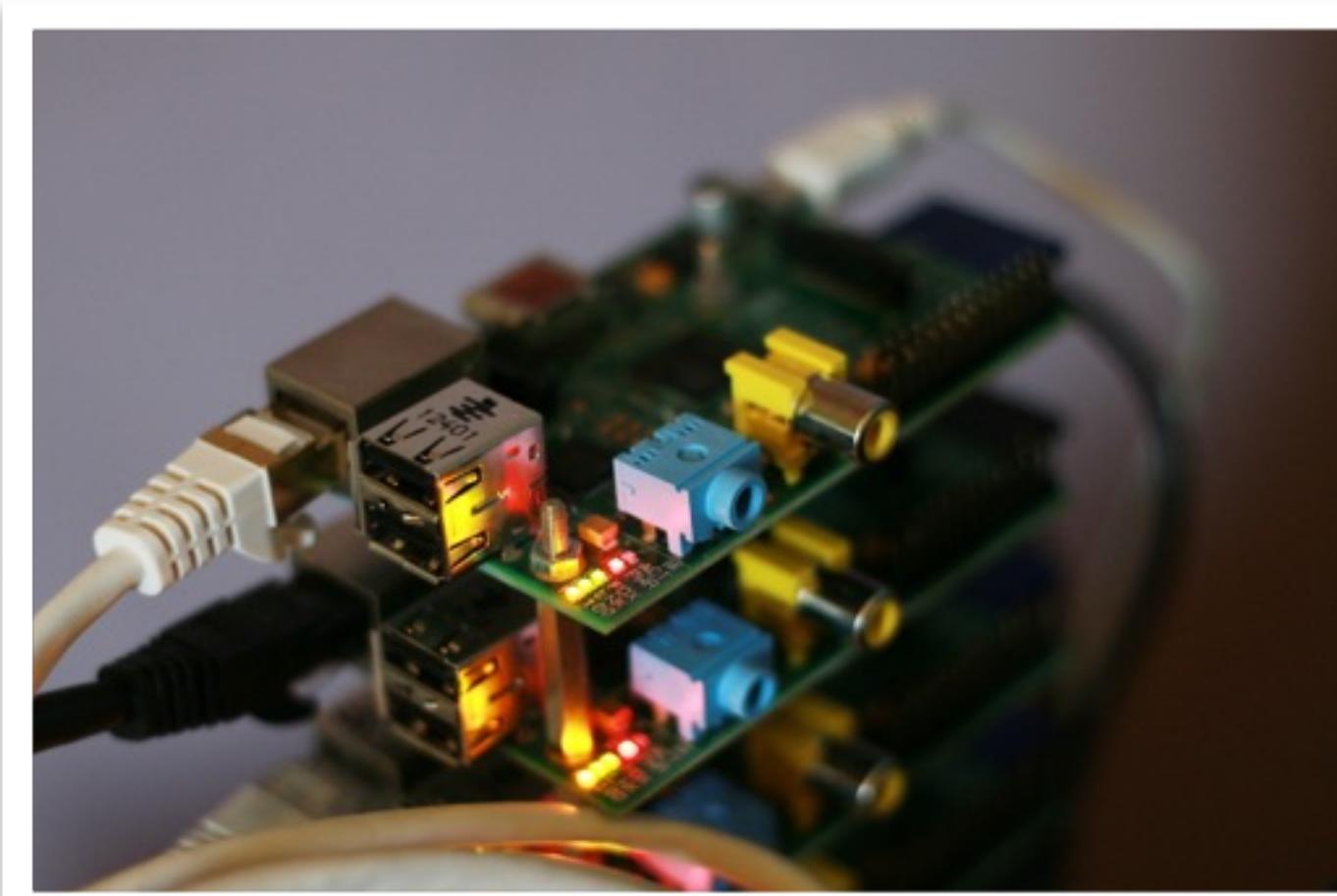
Mitraillette années 30
Mitraillette années 30. P ...
3599.0 €
[Add to cart](#)

Arbalète et accessoires
Magnifique arbalète pouli ...
200.99 €
[Add to cart](#)

#RaspberryCluster #Zombies

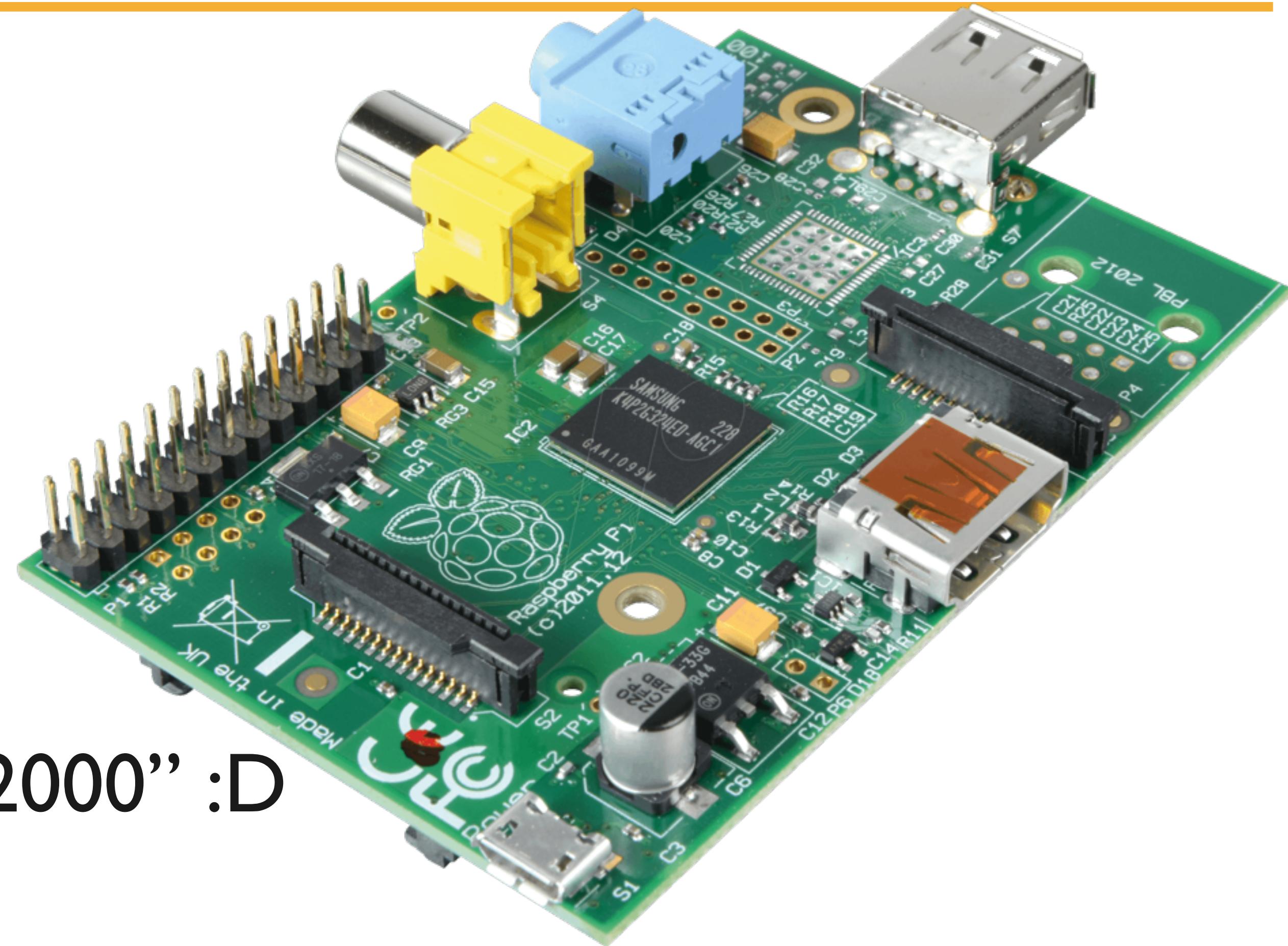
@TrevorReznik @chanksleroux

Mobile infrastructure



Raspberry Pi B+

- CPU 700 MHz
- 512 Mo de RAM
- 2 ports USB
- Carte SD
- Port ethernet
- “Un ordinateur comme en 2000” :D
- Le tout pour ~35\$



Mobile infrastructure



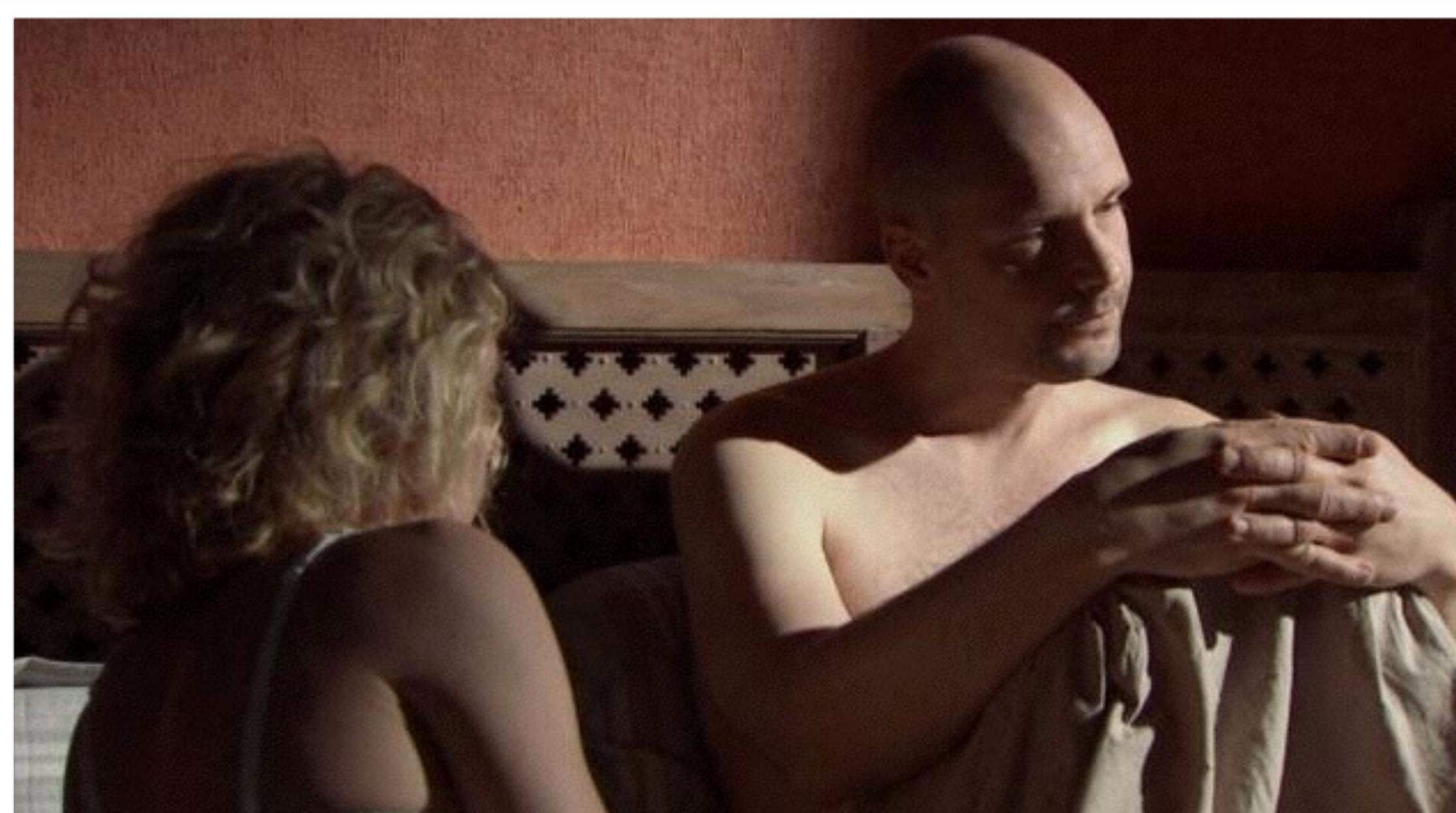
Démo



Résister à la charge



Résister aux pannes

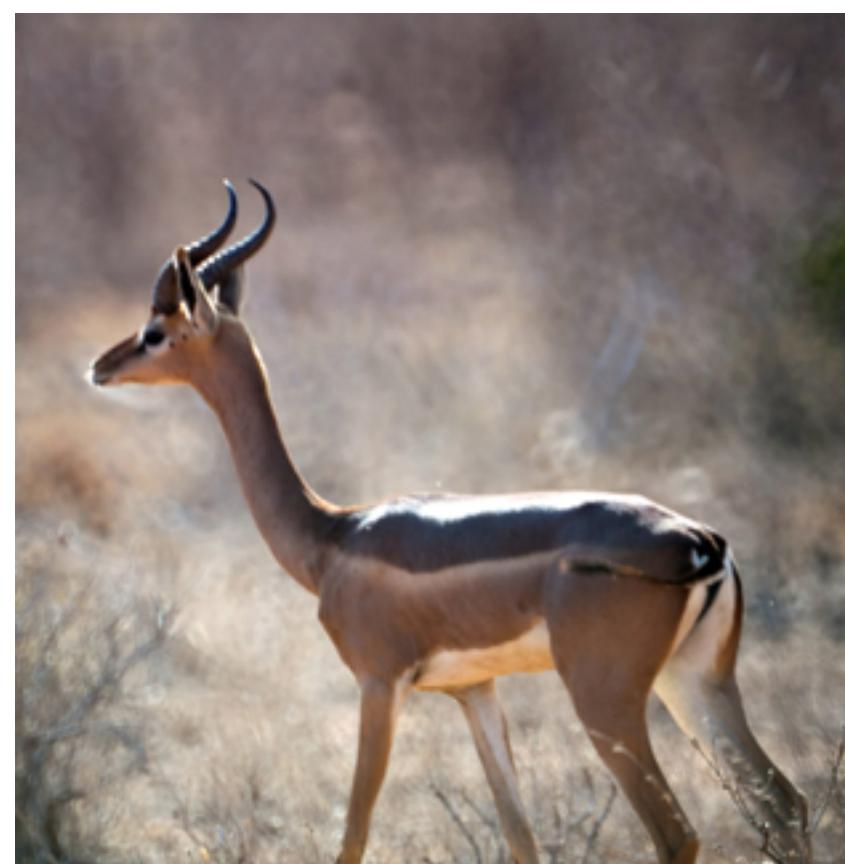
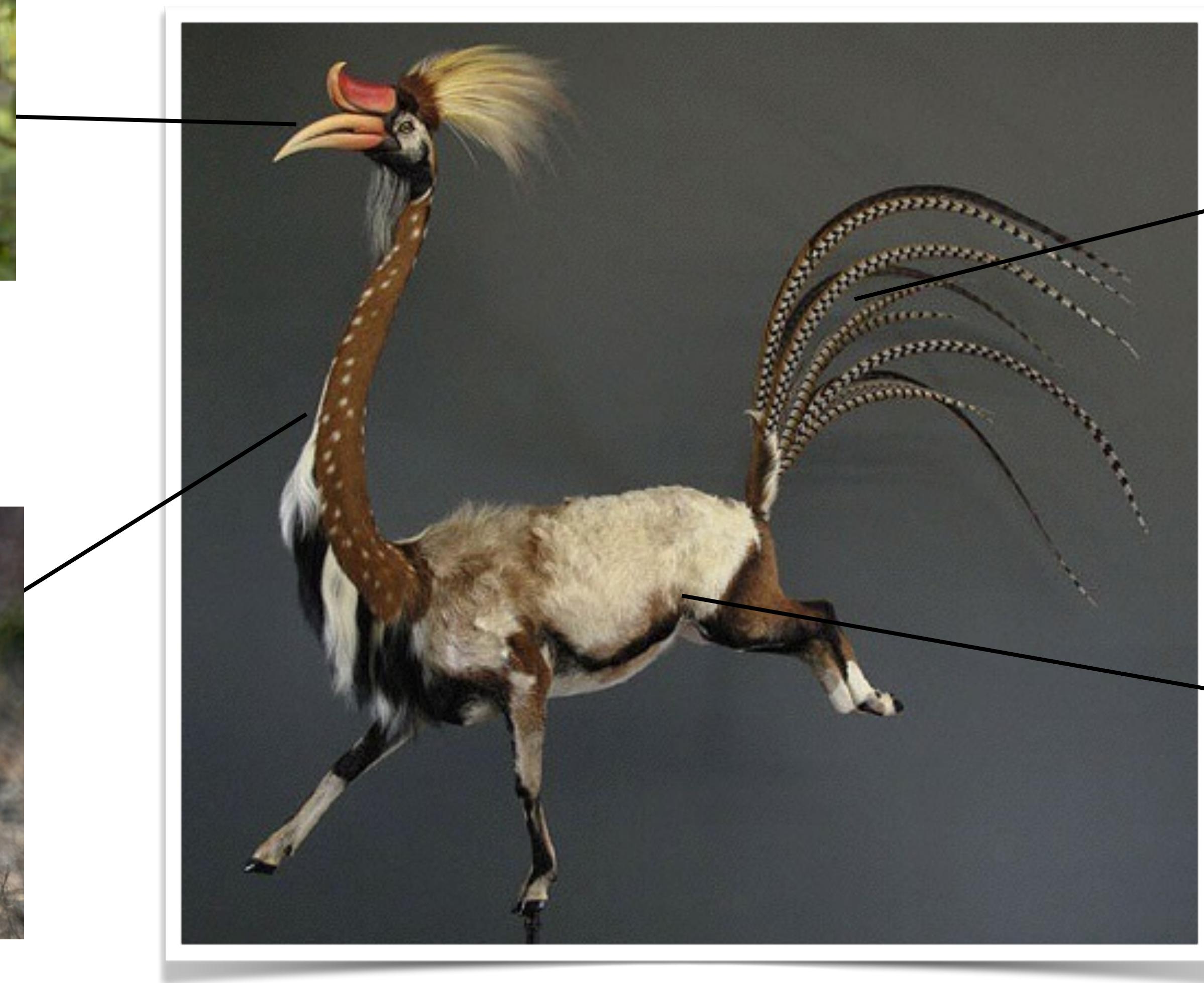


Absorber une hausse de trafic



Approche classique

Monolithique



Latence



Bloquant



Charge



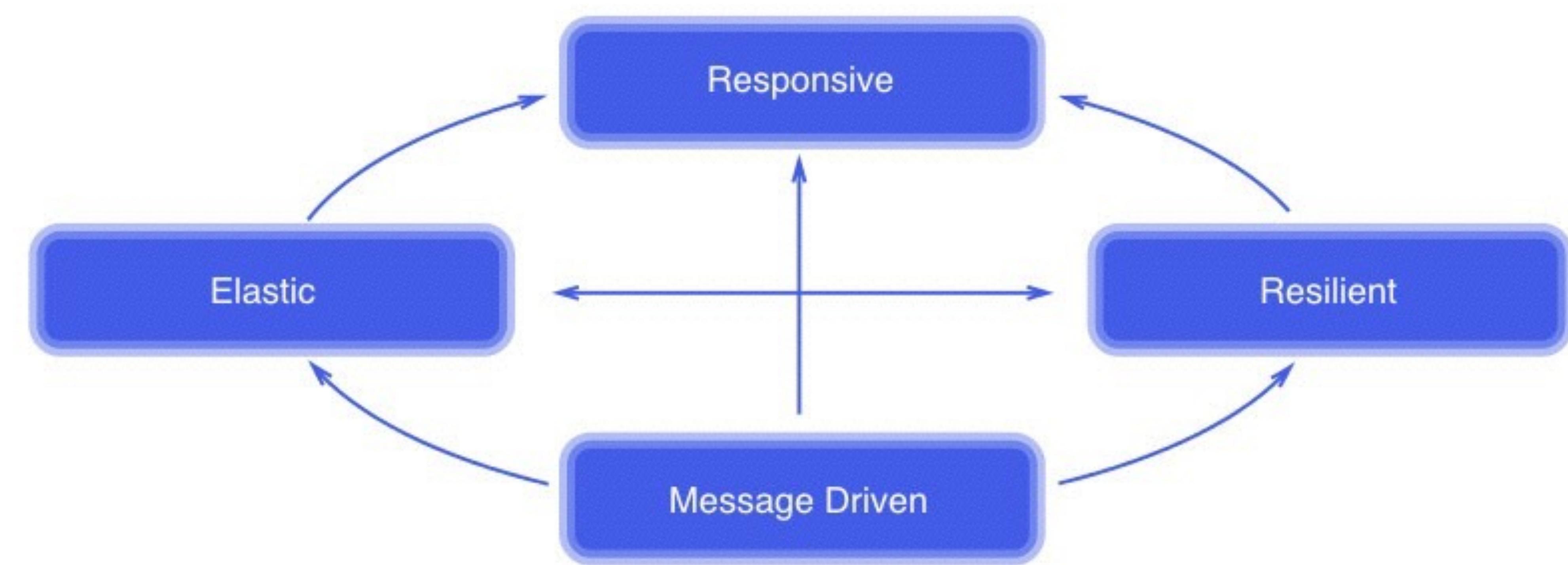
Scalabilité



Reactive manifesto



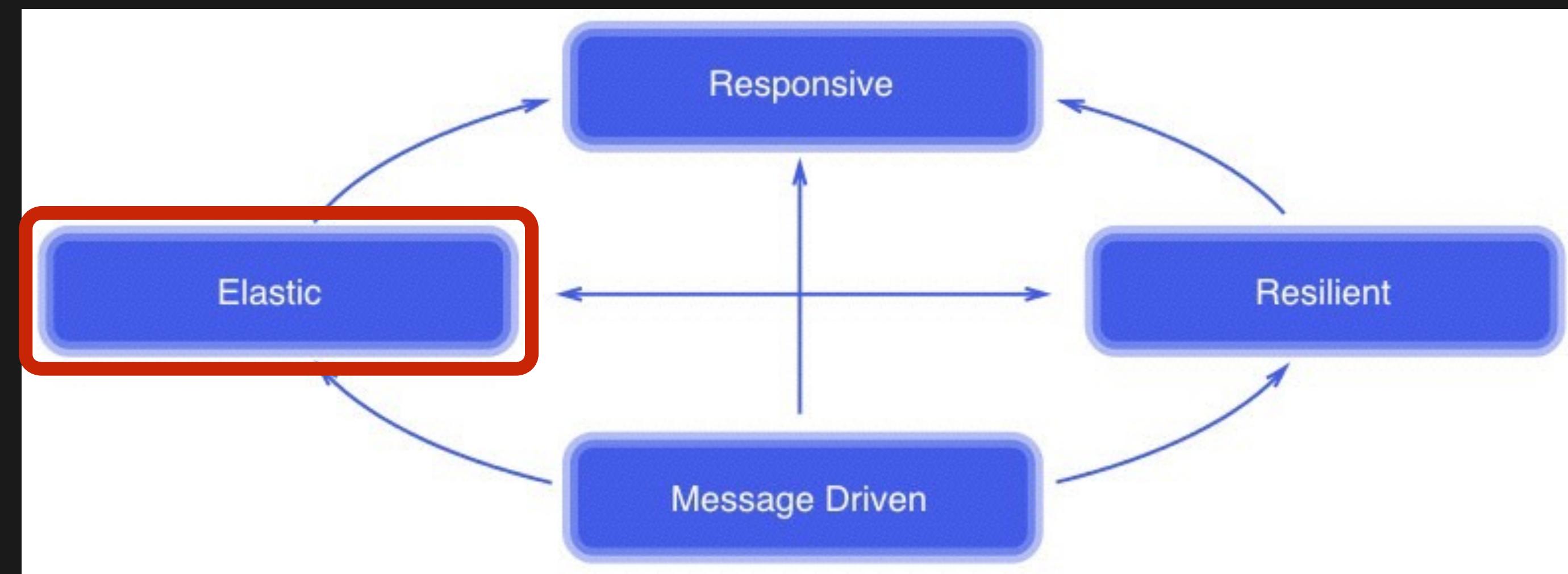
Reactive Manifesto



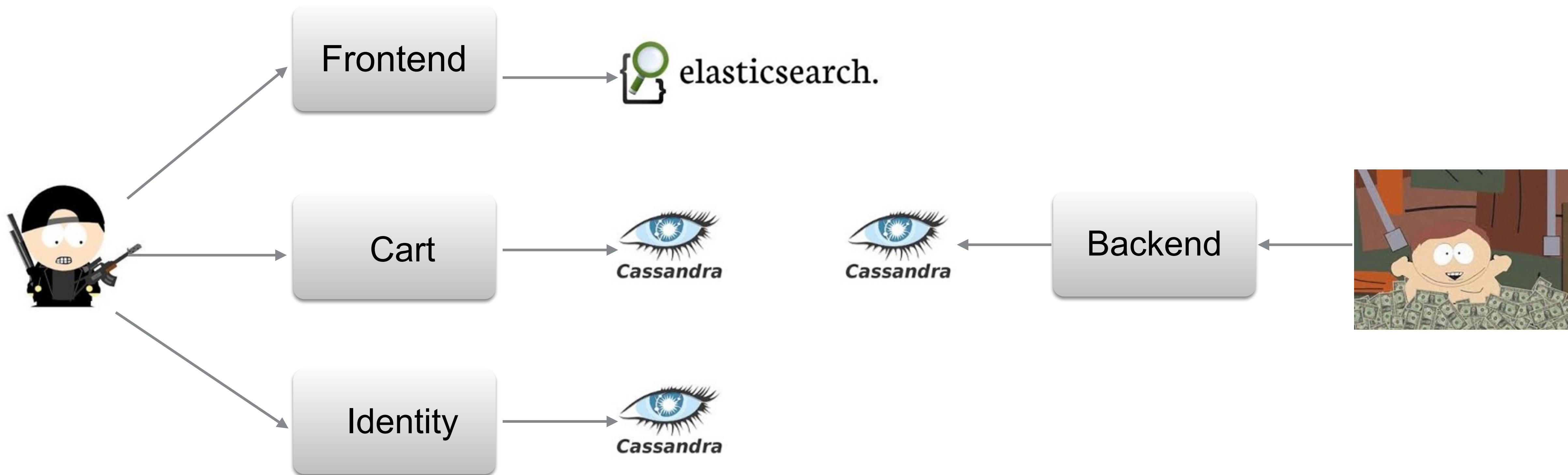
Reactive manifesto

- **responsive** : le système répond dans un temps acceptable et linéaire
- **resilient** : le système répond même en cas de panne.
- **elastic** : le système répond malgré une charge variable. On doit pouvoir augmenter ou en diminuer les ressources allouées aux services.
- **message driven** : le système repose sur des envois de messages asynchrones. Ceci assure un couplage lâche entre les composants de l'application

Elastic



Overview



Datas



elasticsearch.



Mini / Micro Services

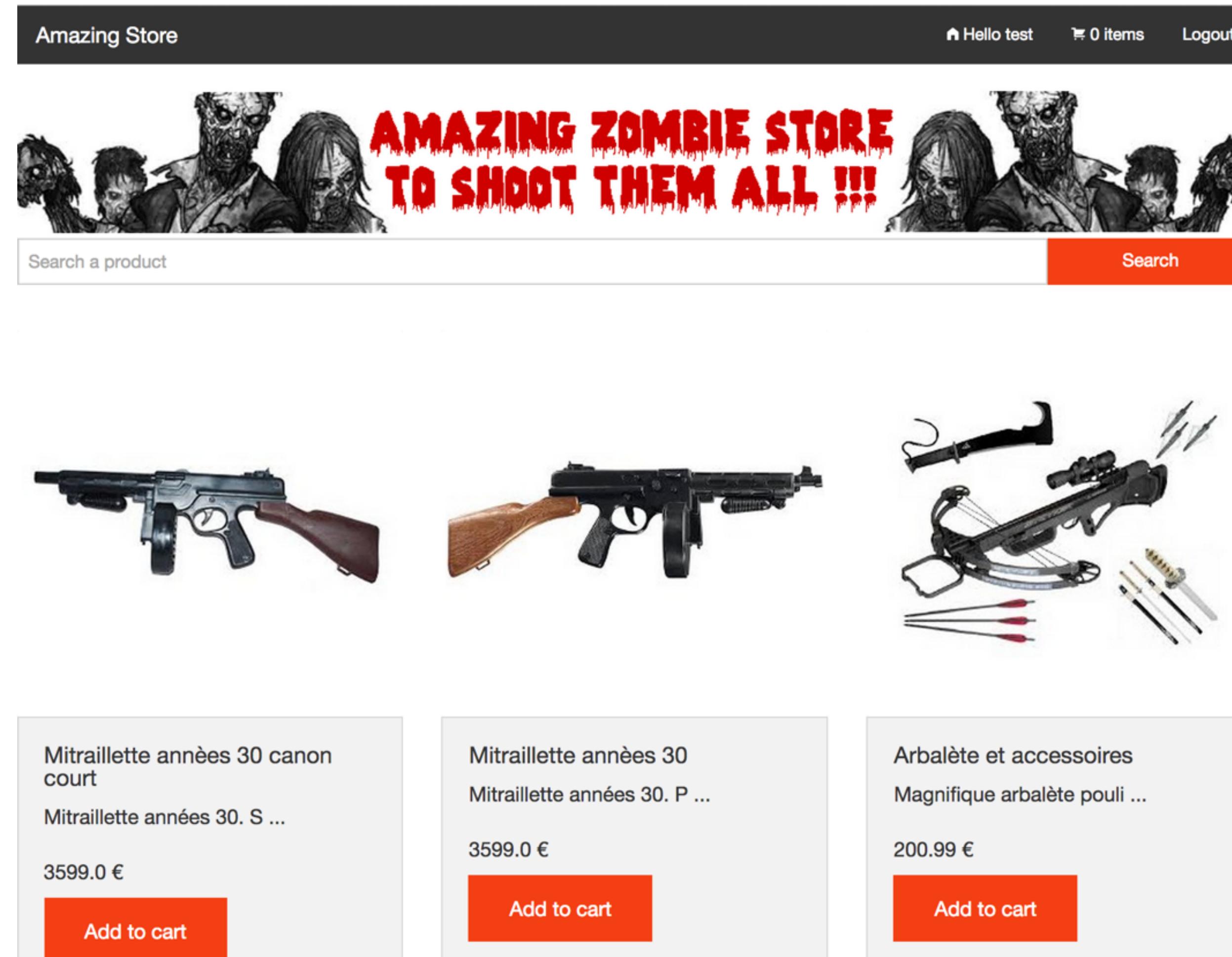
- Une application par domaine fonctionnel
 - store-frontend : présentation du contenu
 - store-identity : authentification / gestion de compte
 - store-cart : panier
 - store-backend : administration du site

Stateless

- Chaque application est stateless
 - aucune donnée n'est stockée dans l'application (pas de cache, pas de fichier ...)
- Chaque application peut être clonée



Frontend

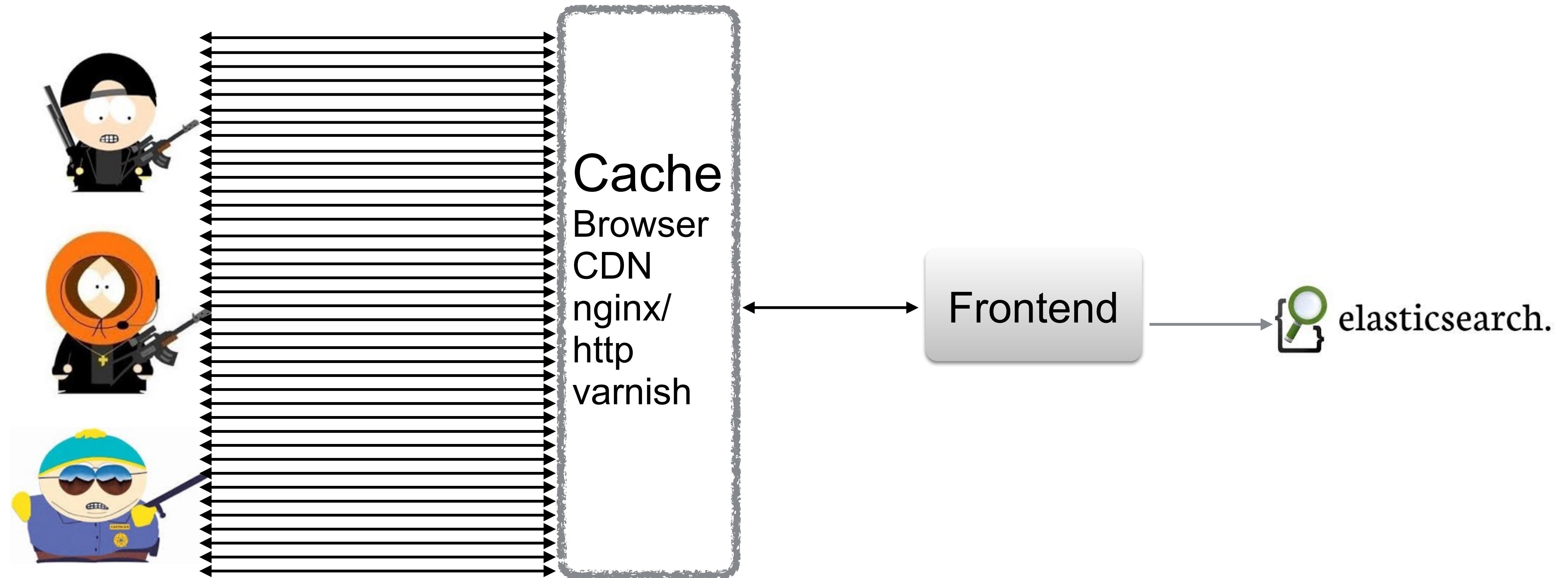


- 100 % html
- Indexation par les moteurs de recherche
- stateless
- une url == un contenu

Limiter la charge



Cache



Optimisations

- Elasticsearch comme cache html pour le contenu
- Recherche plein texte



Base de données ?

Cache
+
recherche full text

Modèle de données

```
{  
    id: "04abe480-2521-11e4-acde-f7b0d99b8321",  
    label: "Product number 1",  
    description: "A description ...",  
    image: "image.jpeg",  
    price: 1.5,  
    fragments: [  
        {  
            type: "search",  
            html: "<div>...</div>"  
        }, {  
            type: "cart",  
            html: "<tr>...</tr>"  
        }  
    ]  
}
```

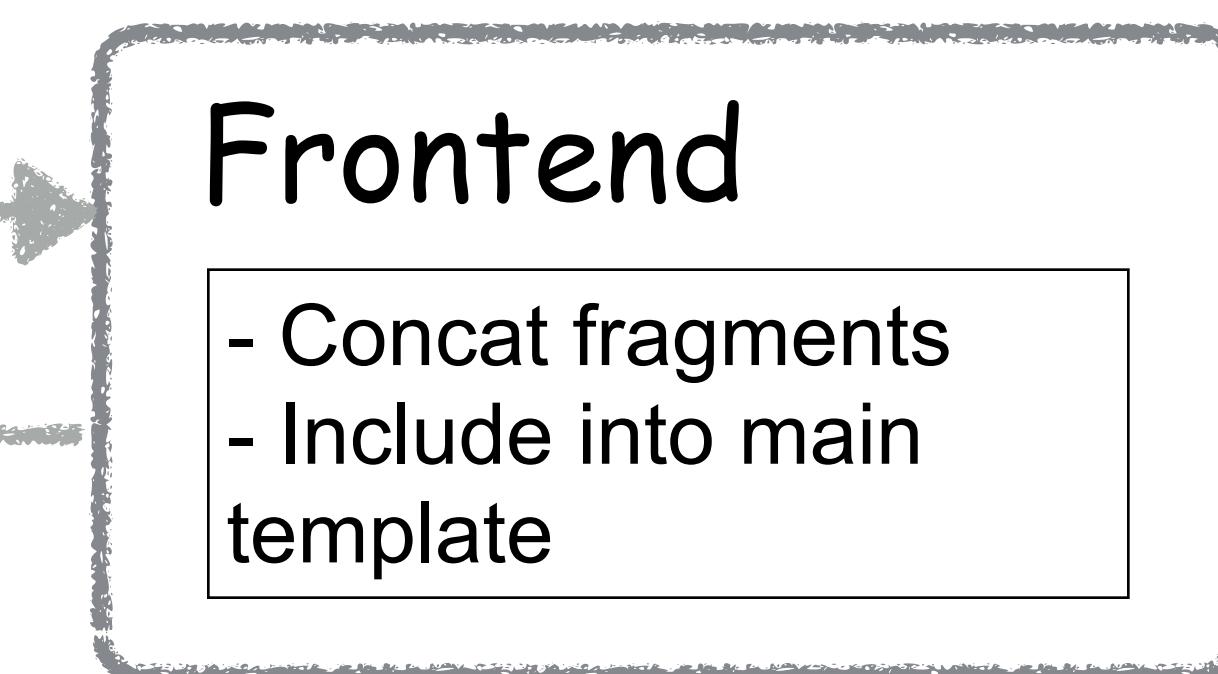
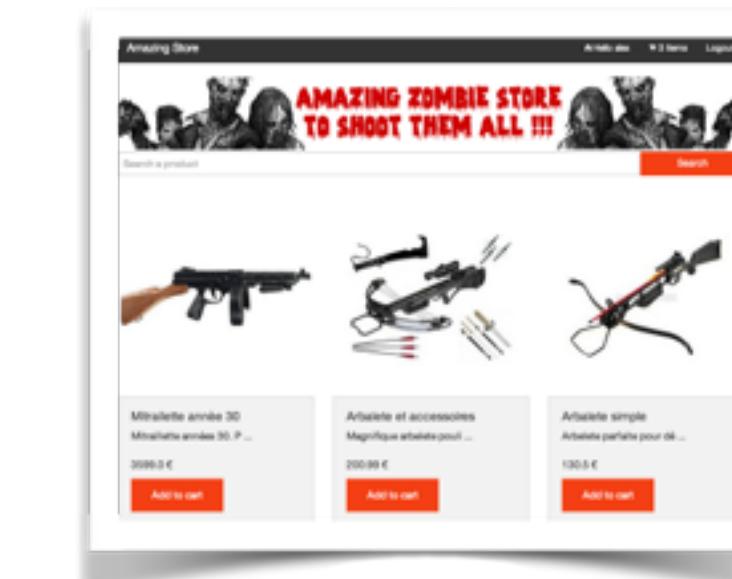
Données indexées pour la recherche

HTML pré-généré

Recherche



GET /products?q=zombie

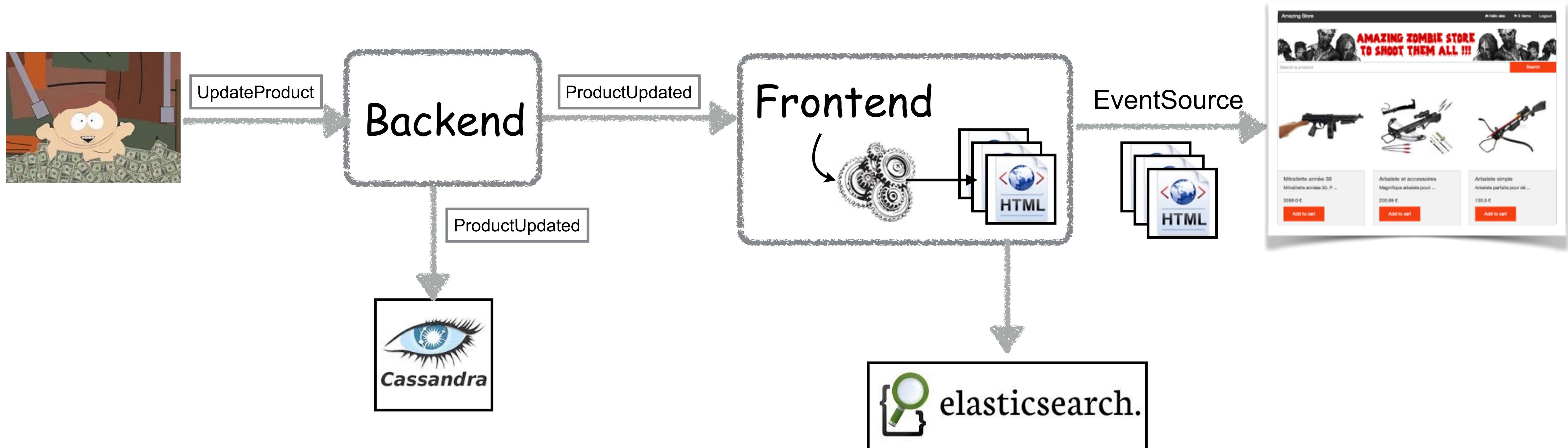


Search
zombie

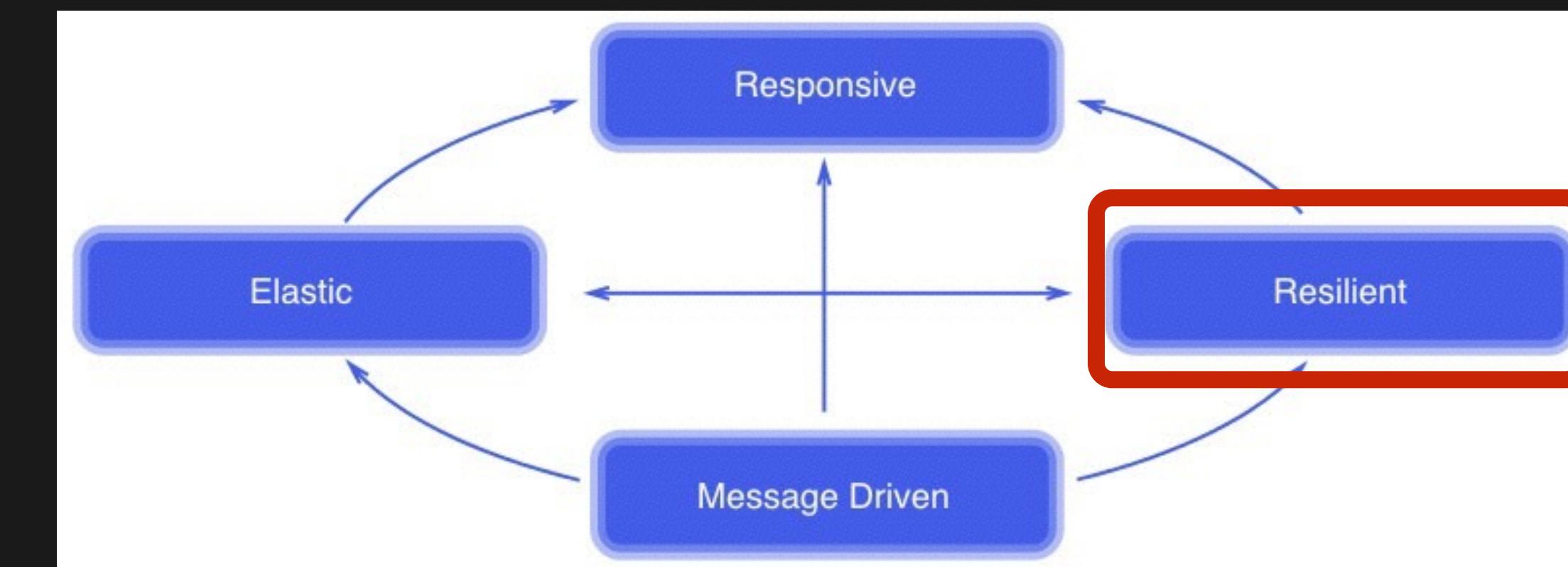
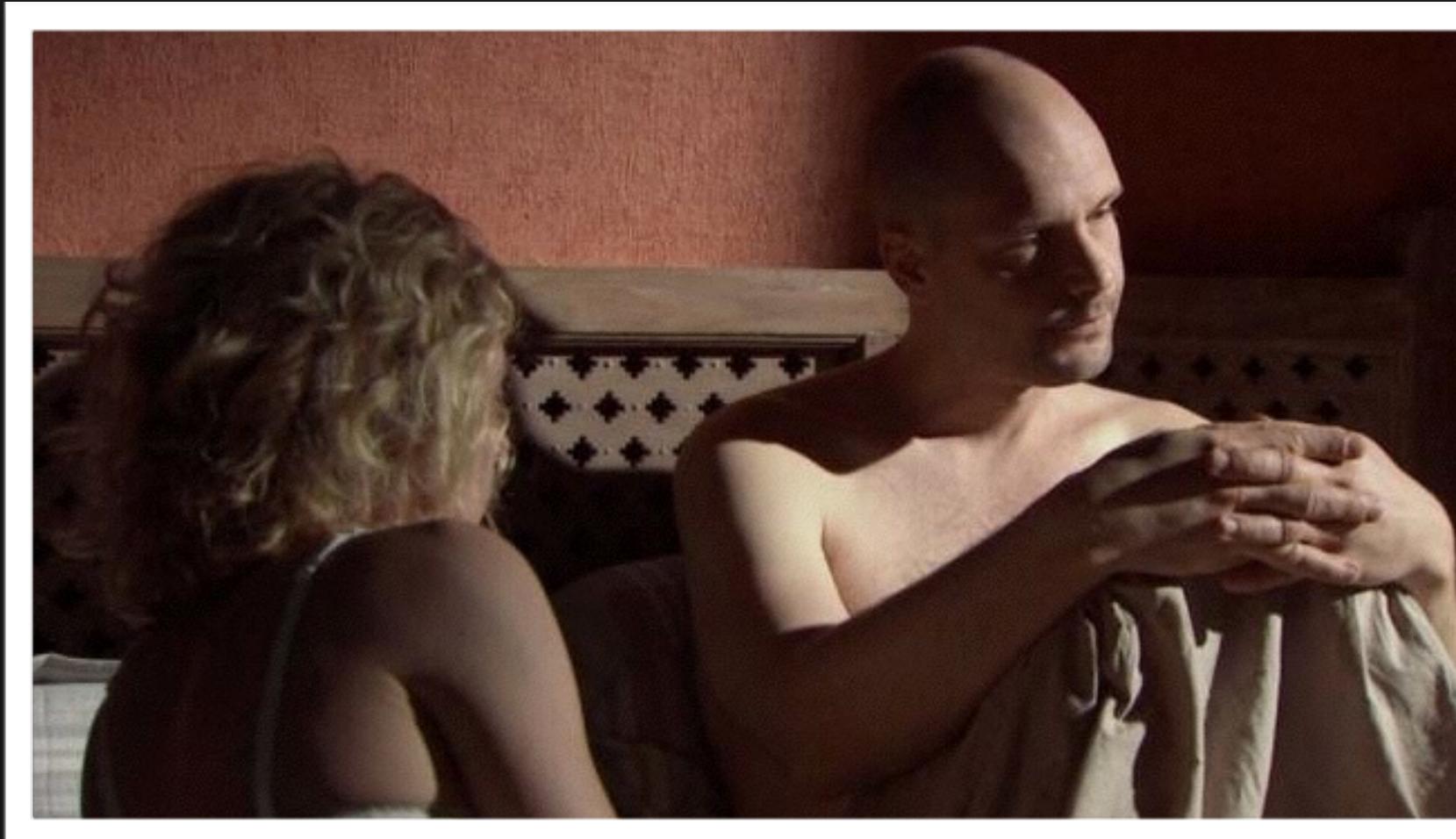


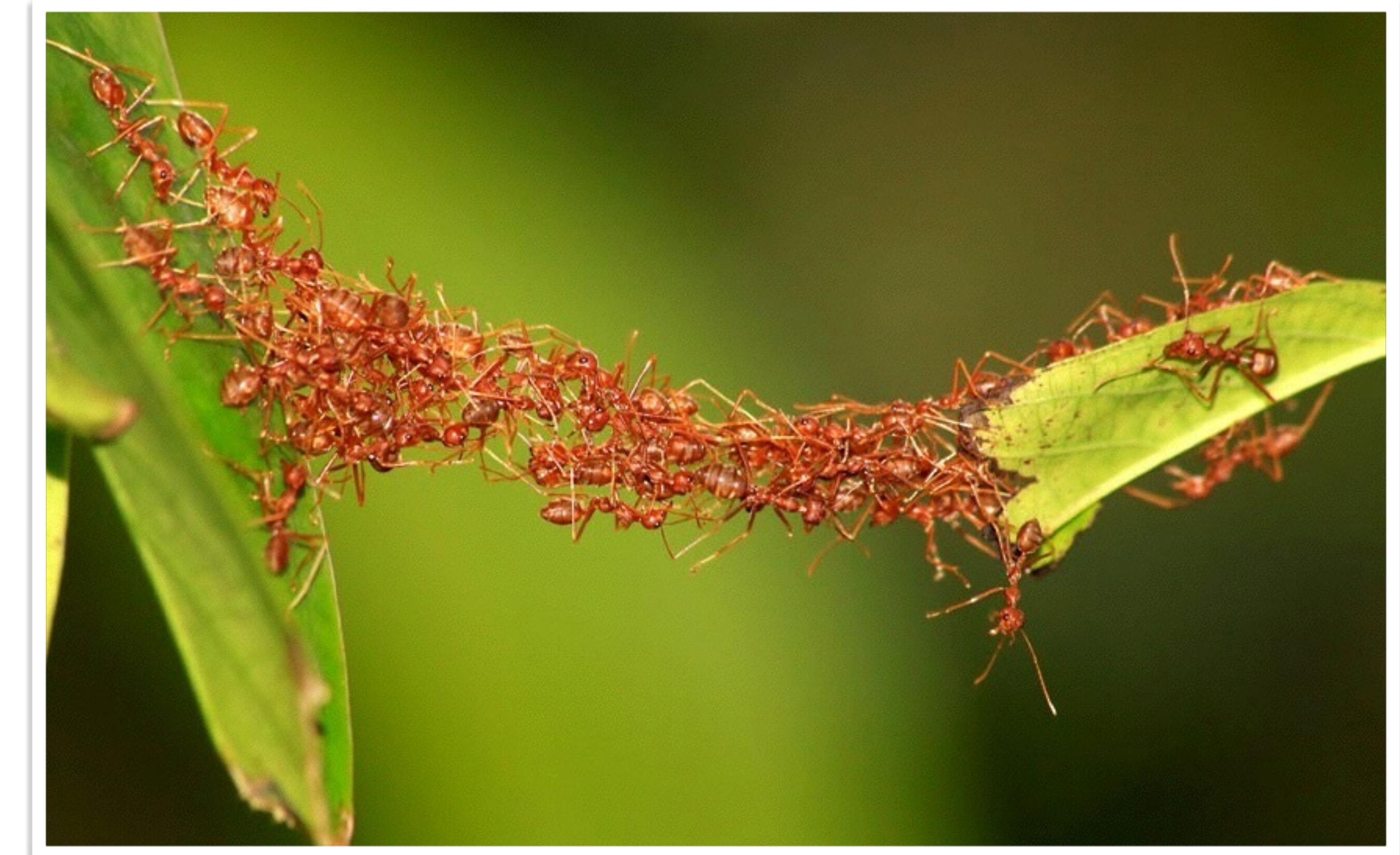
Créer / mettre à jour

DEVOXX France



Resilient



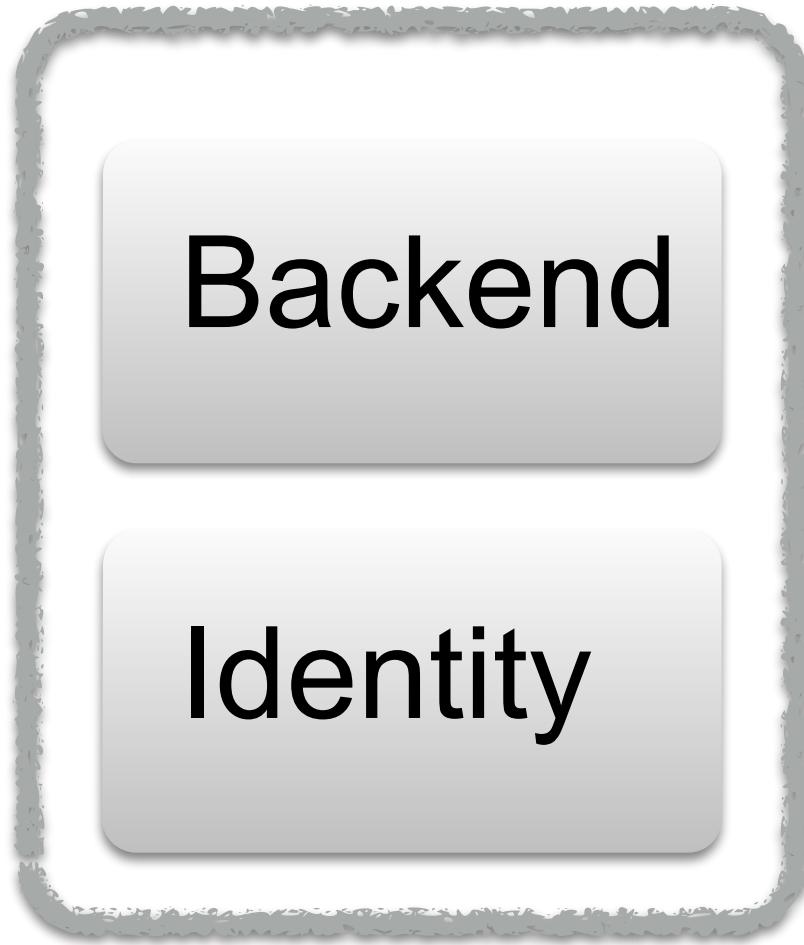
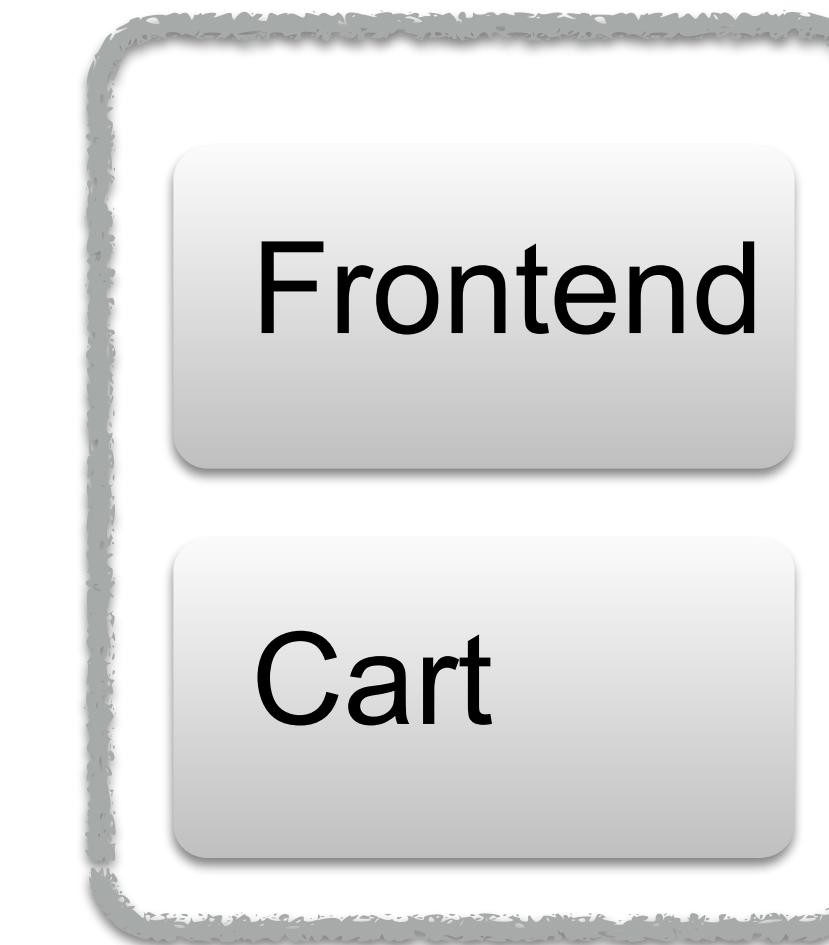
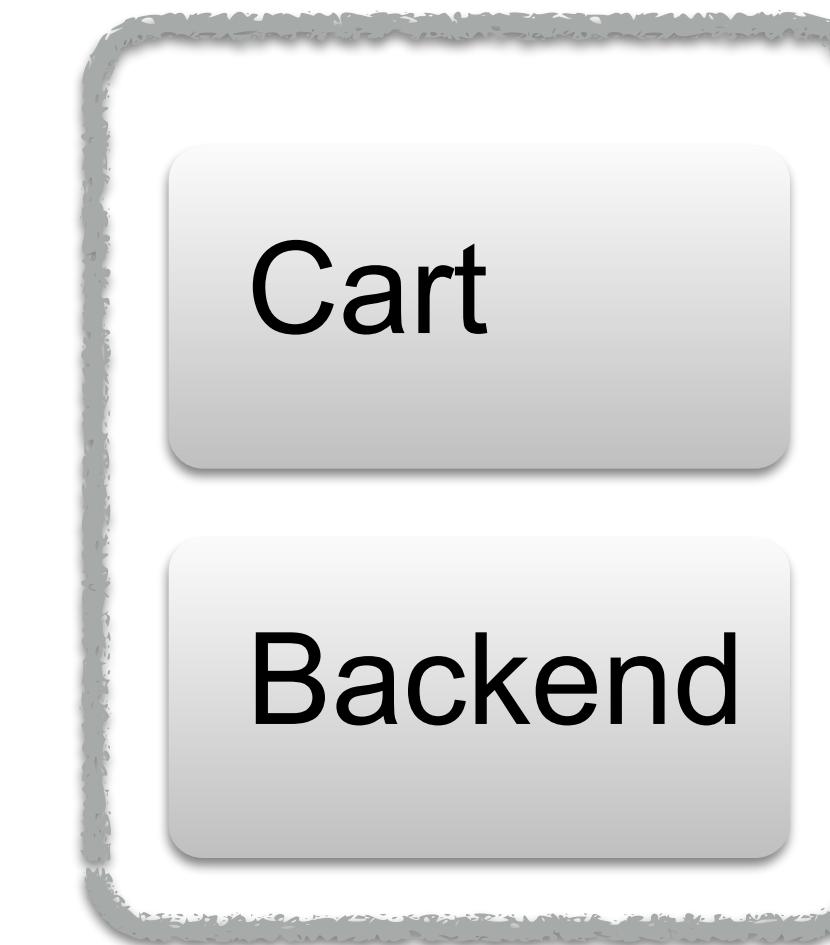
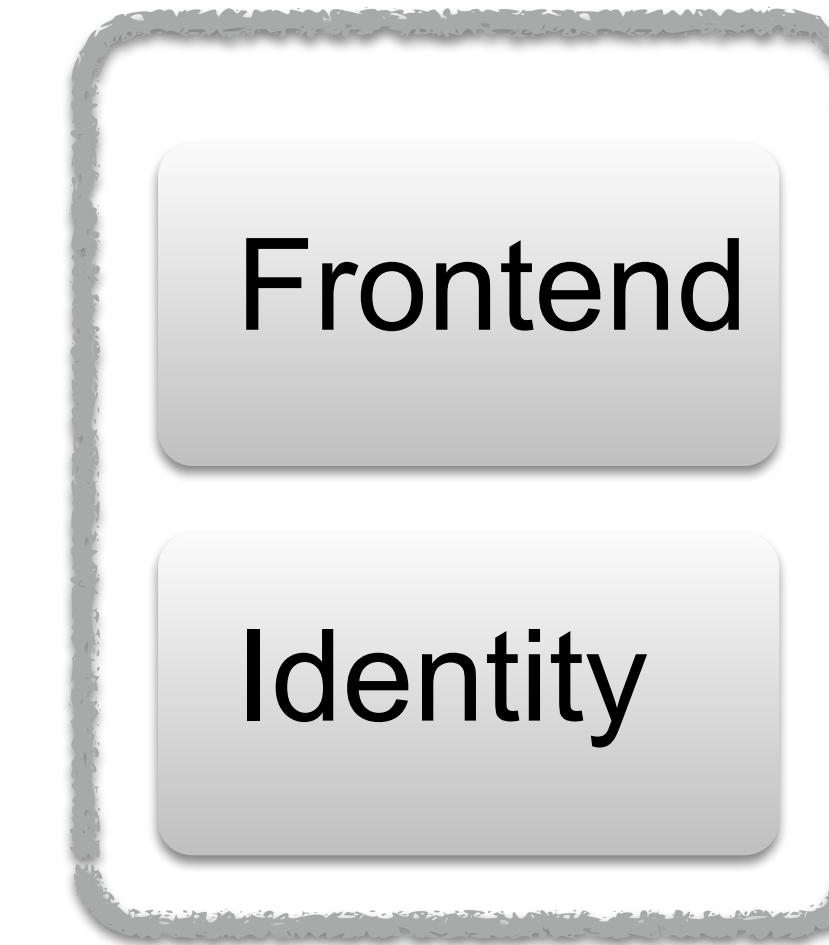


#RaspberryCluster #Zombies

@TrevorReznik @chanksleroux

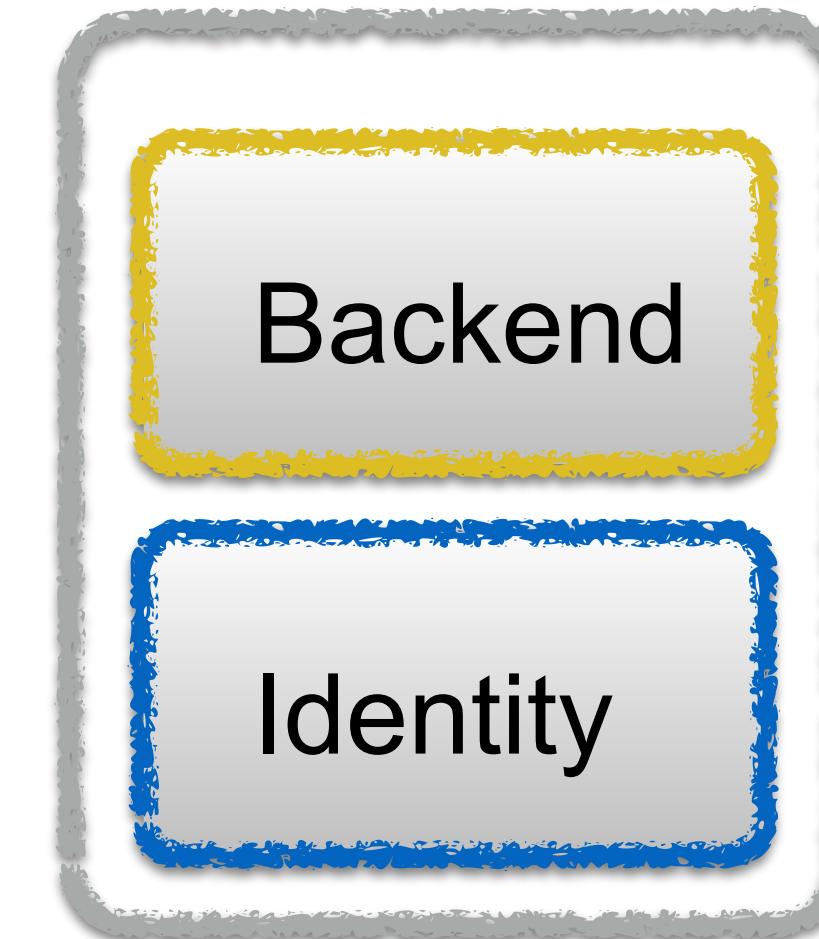
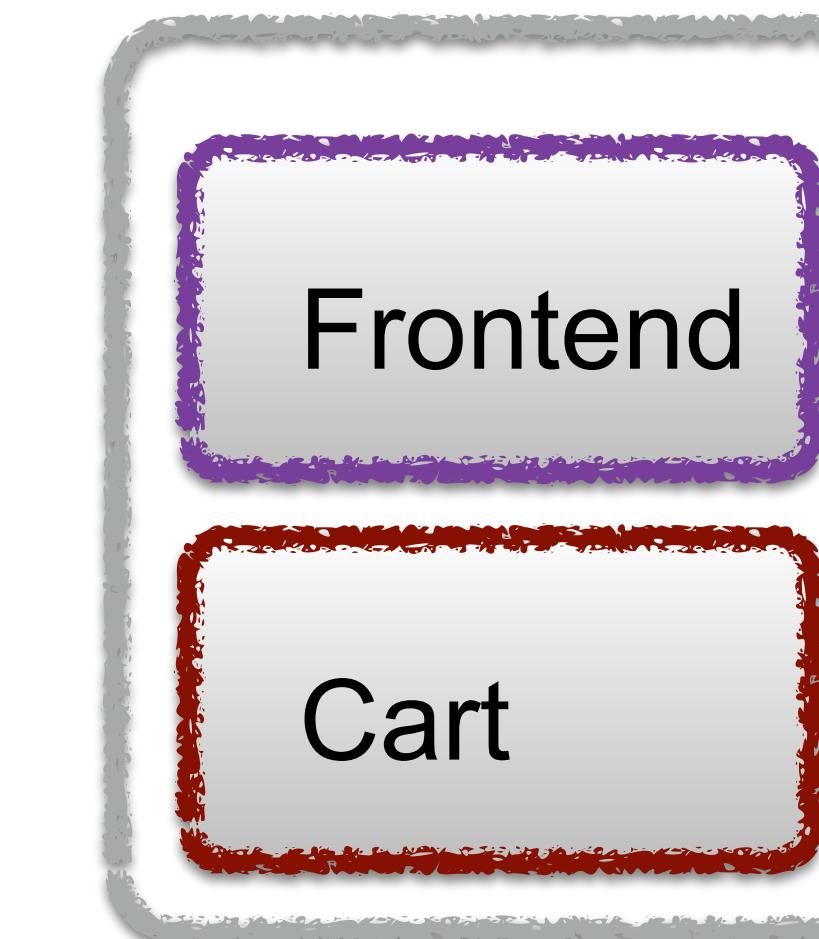
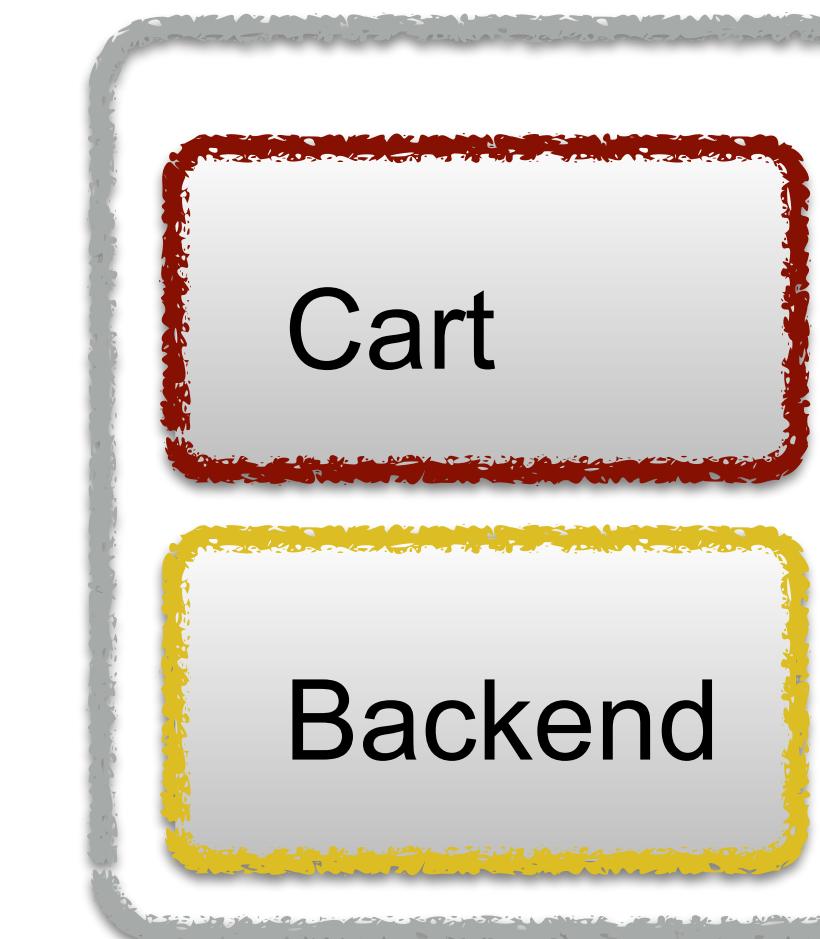
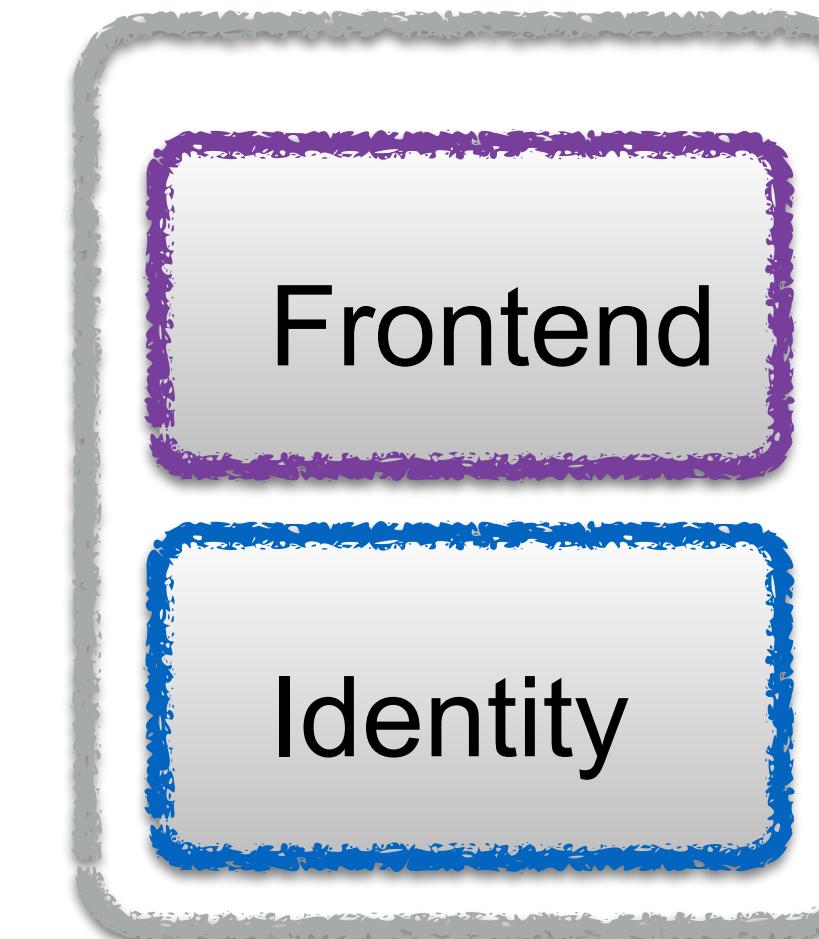
Infrastructure

DEVOXX France



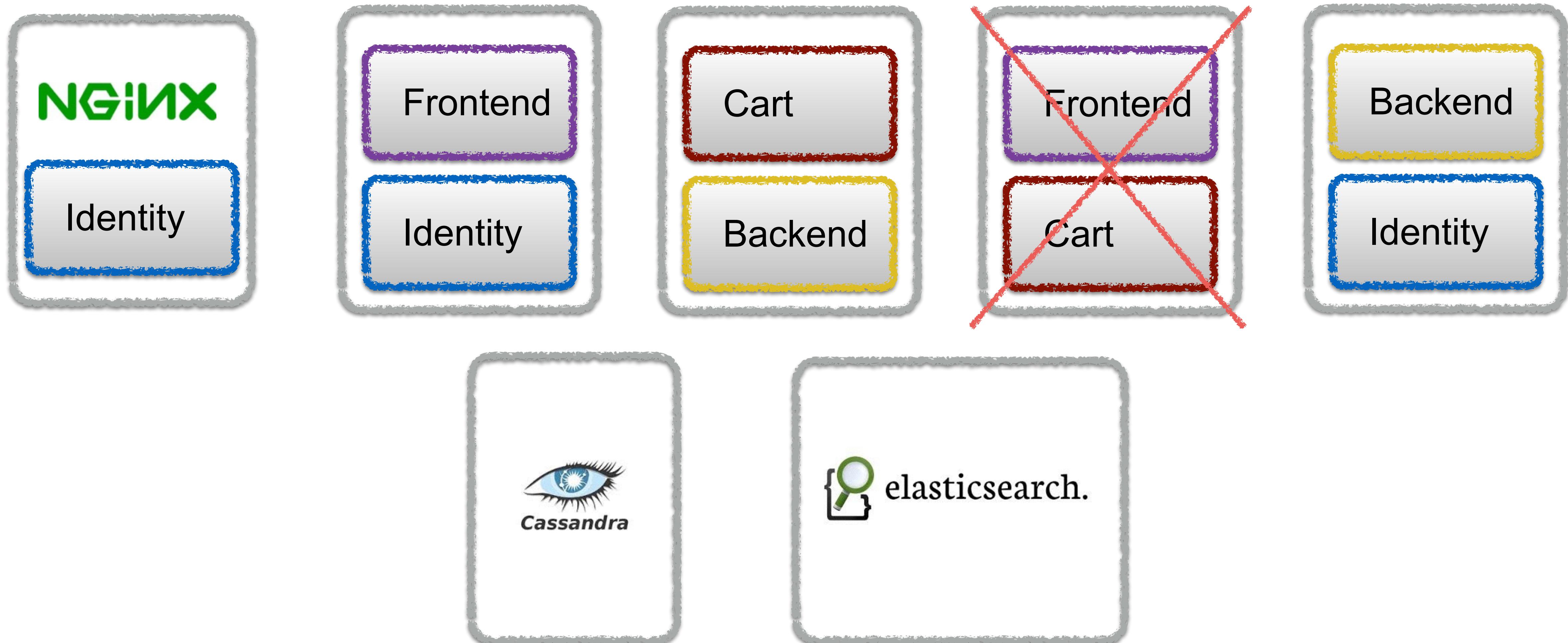
Infrastructure

DEVOXX France



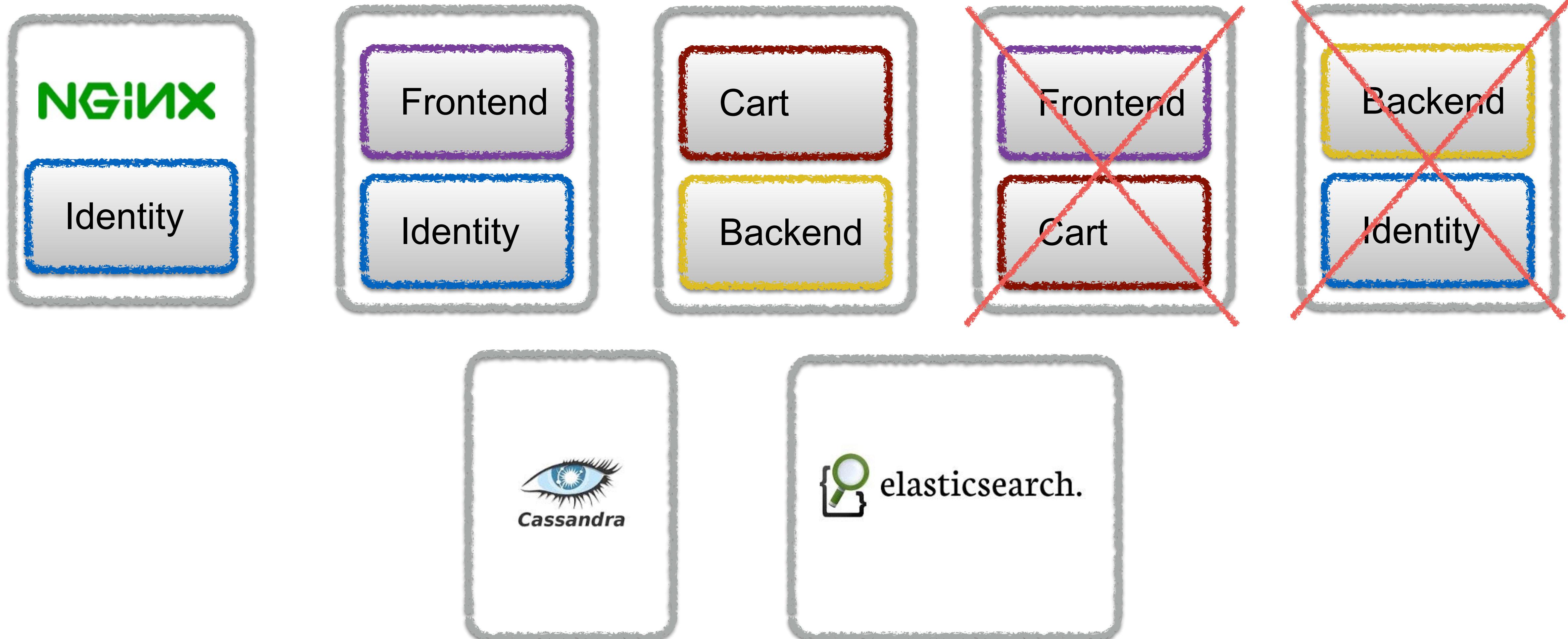
Infrastructure

DEVOXX France



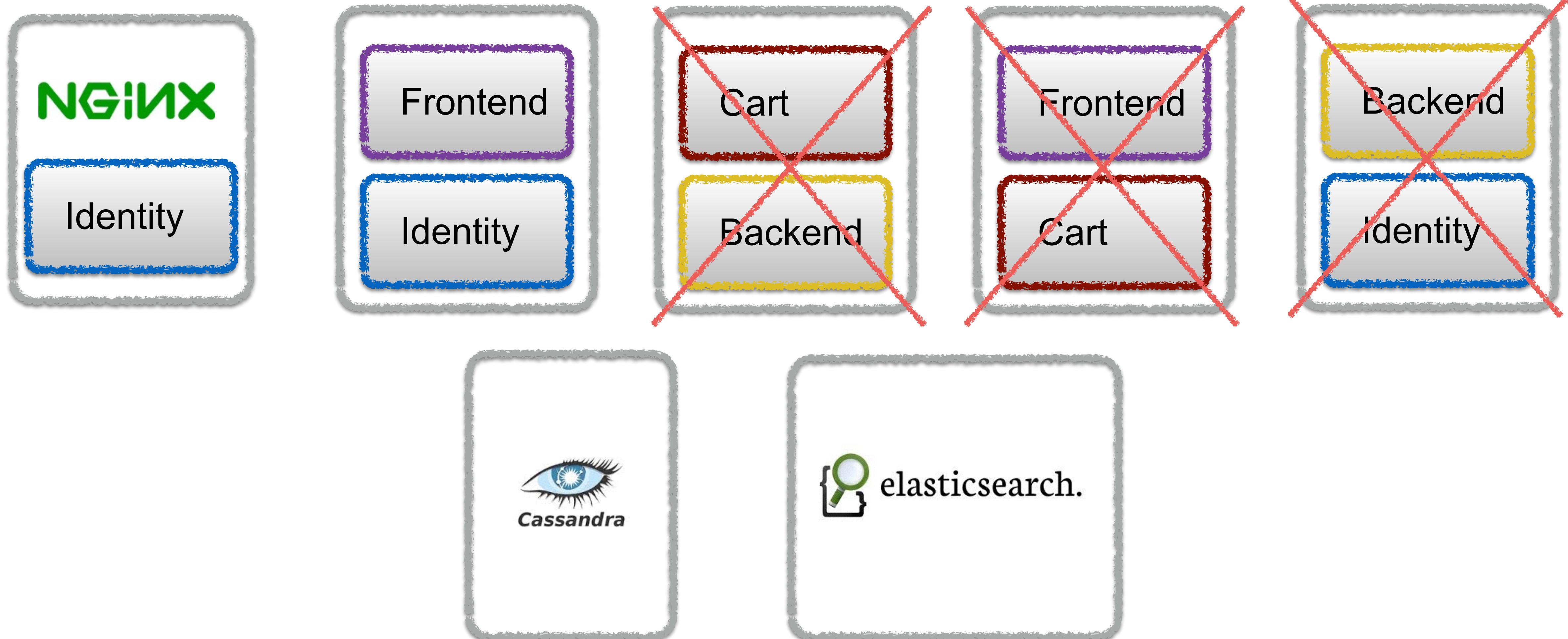
Infrastructure

DEVOXX France



Infrastructure

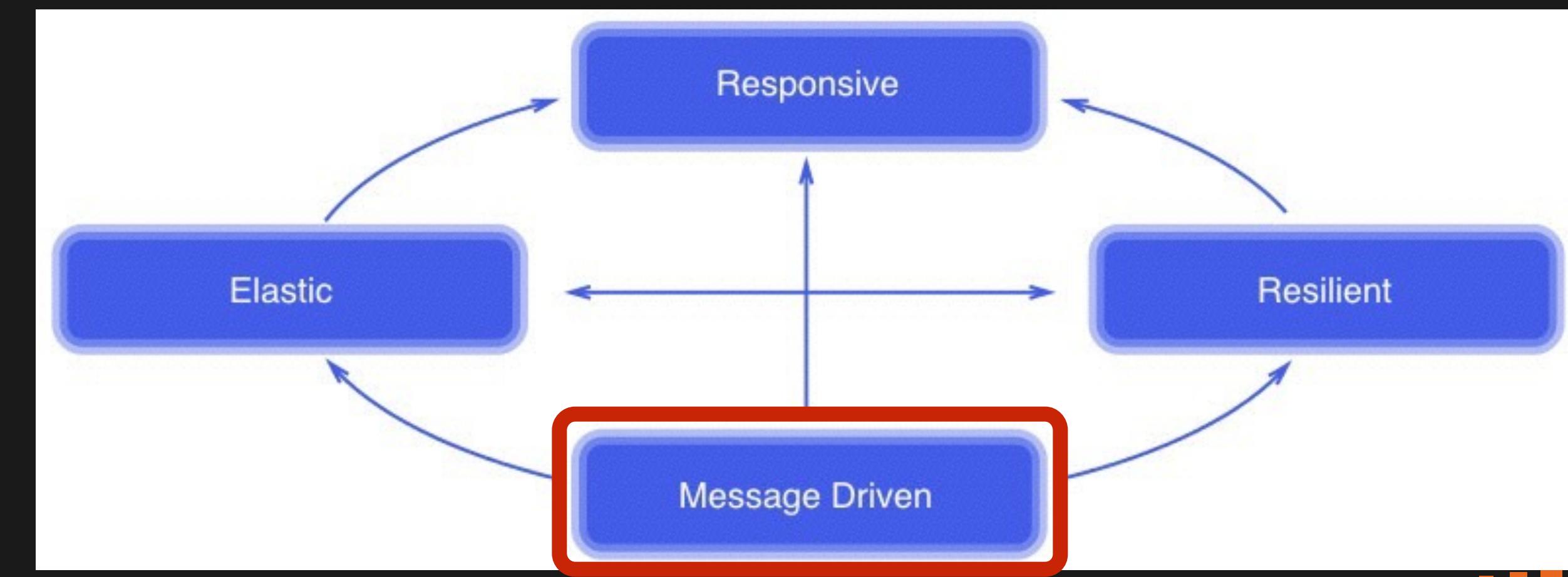
DEVOXX France



Démo : let it crash



Message driven



Akka ?



GETTY IMAGES



MAKE GIFS AT GFSOUP.COM

Akka ??



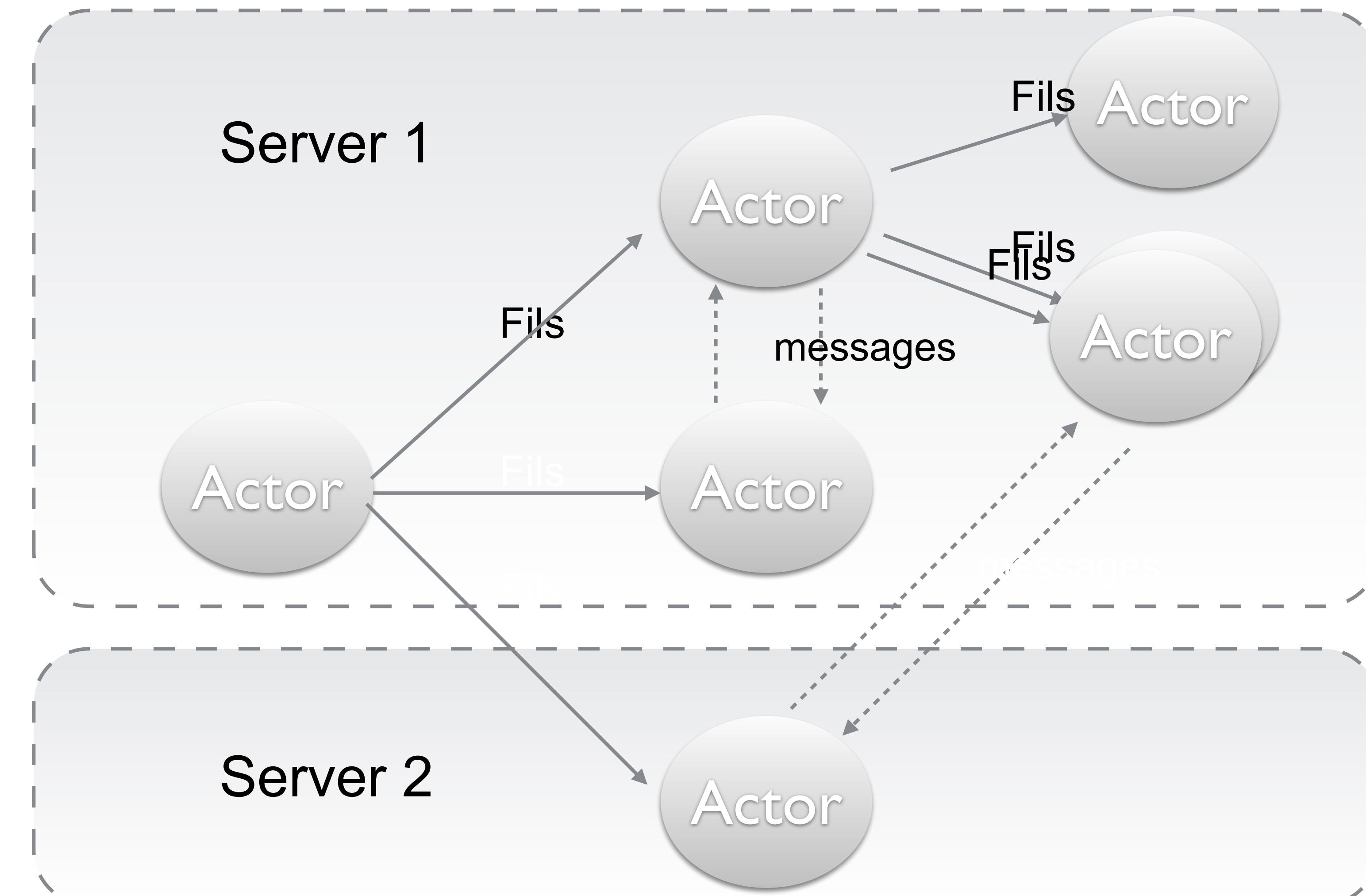
Akka

- Akka
- Modèle acteur
- Un acteur = Une entité statefull
- Communication entre acteurs par messages (même à distance)
- Un acteur peut créer/détruire des enfants
- Un acteur peut surveiller d'autres acteurs
- Plus de problèmes de concurrence, asynchrone par nature
- Résistant aux pannes



Akka

DEVOXX France



Akka

```
import akka.actor.{Props, ActorSystem, ActorLogging, Actor}

case class Greeting(who: String)

class GreetingActor extends Actor with ActorLogging {
  def receive = {
    case Greeting(who) => log.info("Hello " + who)
  }
}

val system = ActorSystem("MySystem")
val greeter = system.actorOf(Props[GreetingActor], name = "greeter")
greeter ! Greeting("Charlie Parker")
```


Struts²

Play 2

- Framework web
- java or scala
- Support pour les websocket et le server sent event
- Asynchrone
- pré-requis pour une application orientée événements



Play async

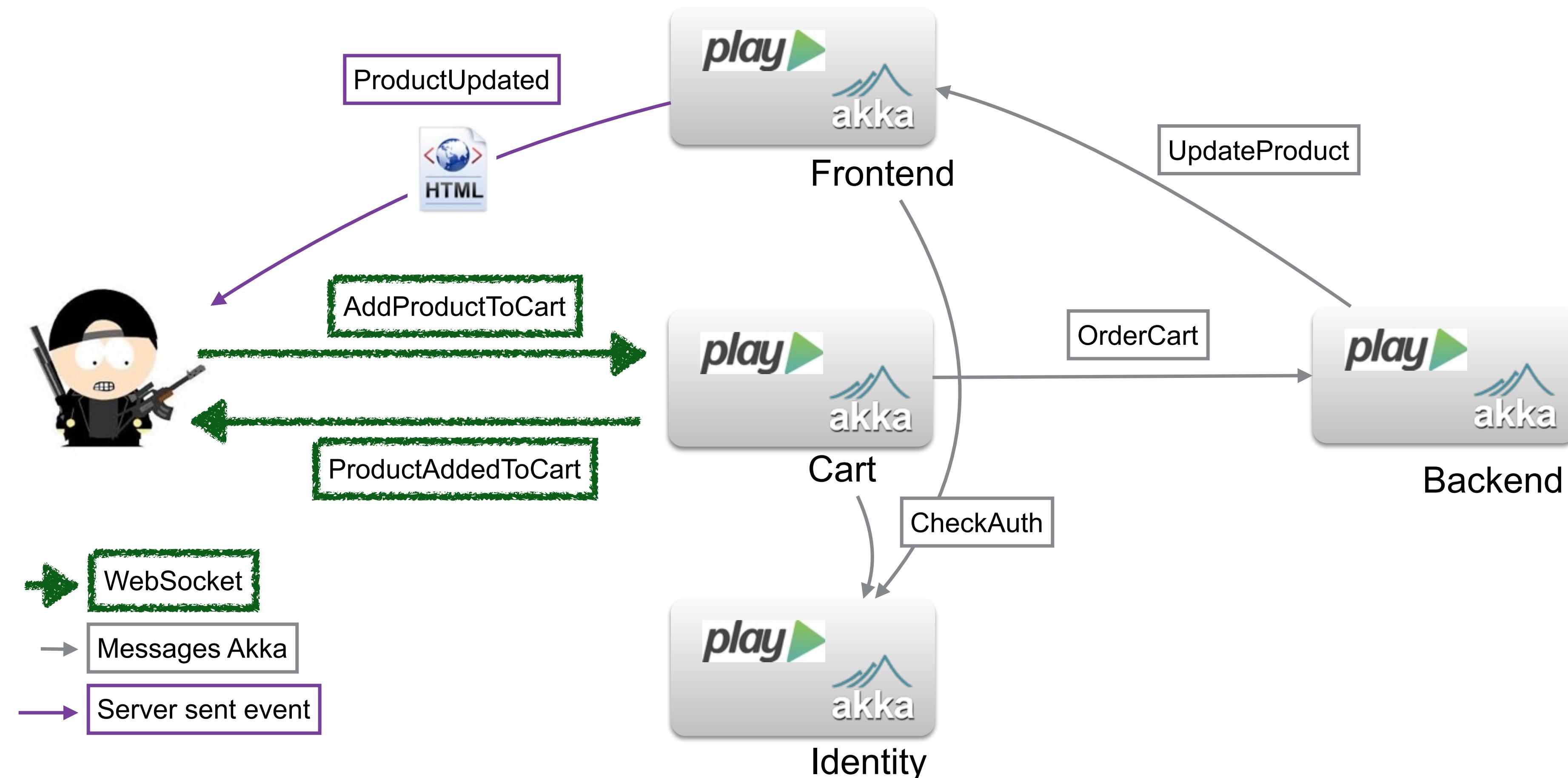
```
case class ListProductsQuery()

class ProductView extends Actor {
  override def receive: Receive = {
    case ListProductsQuery() => models.Product.list() pipeTo sender()
  }
}

class ProductsController extends Controller {
  private def listProducts(): Future[List[models.Product]] = {
    (Actors.productView() ? ListProductsQuery()).mapTo[List[models.Product]]
  }
  def index() = Action.async {
    listProducts().map(products => Ok(views.html.index(products)))
  }
}
```

Messages

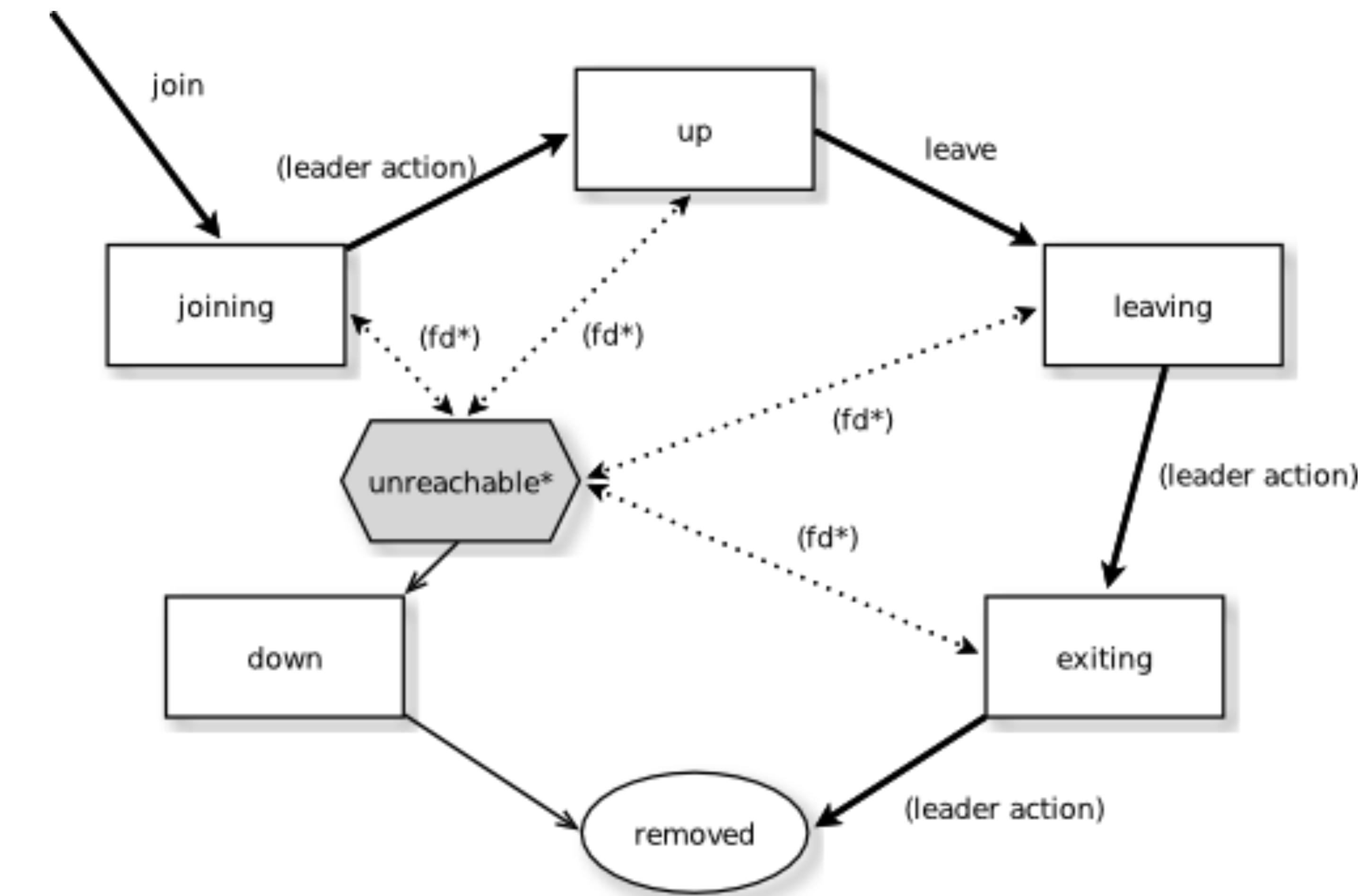
TM
DEVOXX France



Cluster

- Utilisation de Akka-cluster
- Librairie permettant de former un cluster de systèmes d'acteurs
 - simple service d'adhésion
 - tolérant aux pannes
 - décentralisé (P2P, gossip)
 - pas de SPOF
 - pas de SPOB
 - détection des pannes

Akka cluster

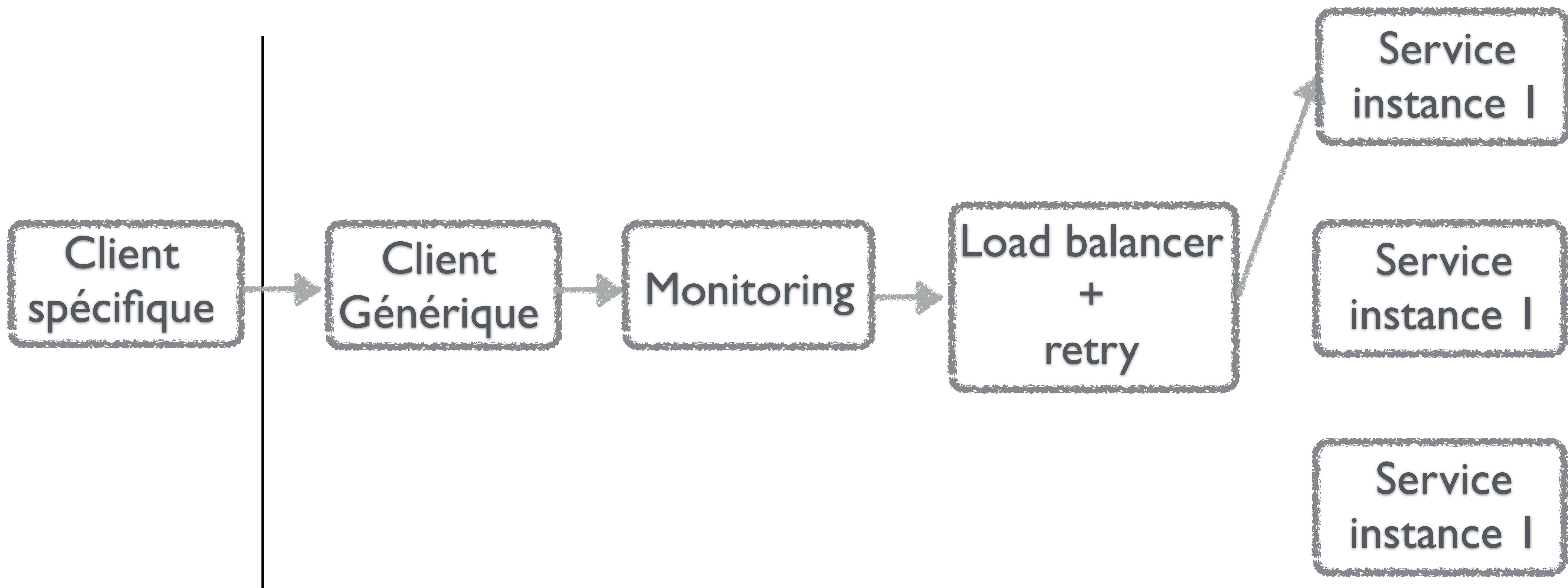


Cluster services

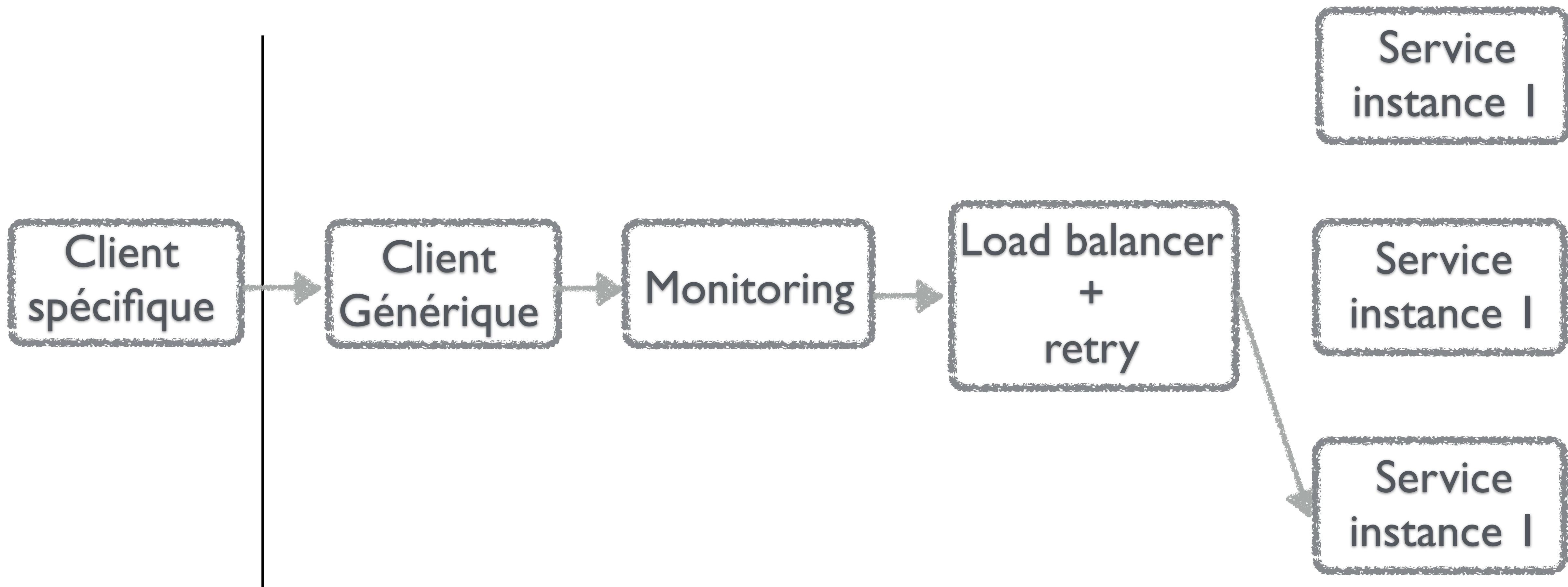
- Librairie de découverte de services distribués
- Exposition descripteurs de services (URL, protocole, version, nom)
- Repose sur les memberships du cluster Akka
- Clients de services
 - HTTP, Akka, Thrift, Memcached ...
- Petits plus
 - Circuit breaker
 - Monitoring
 - Load balancing (pas très intelligent)
 - Retry with exponential back off



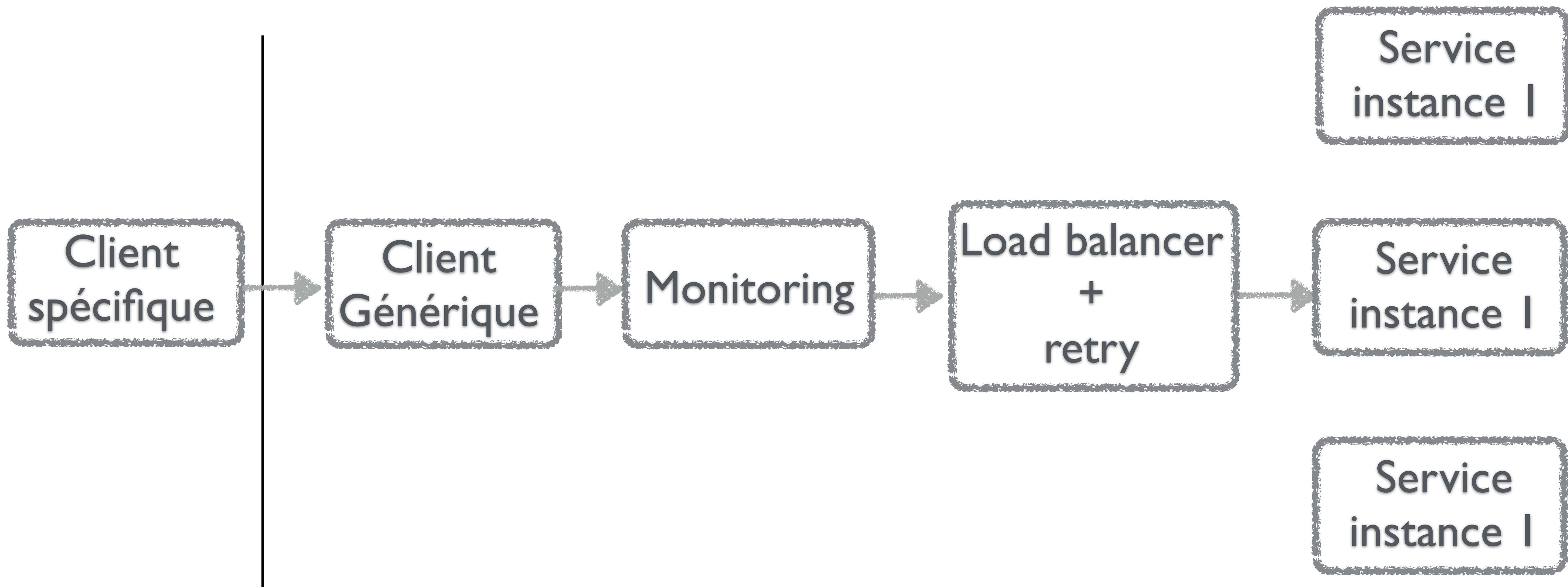
Cluster services



Cluster services

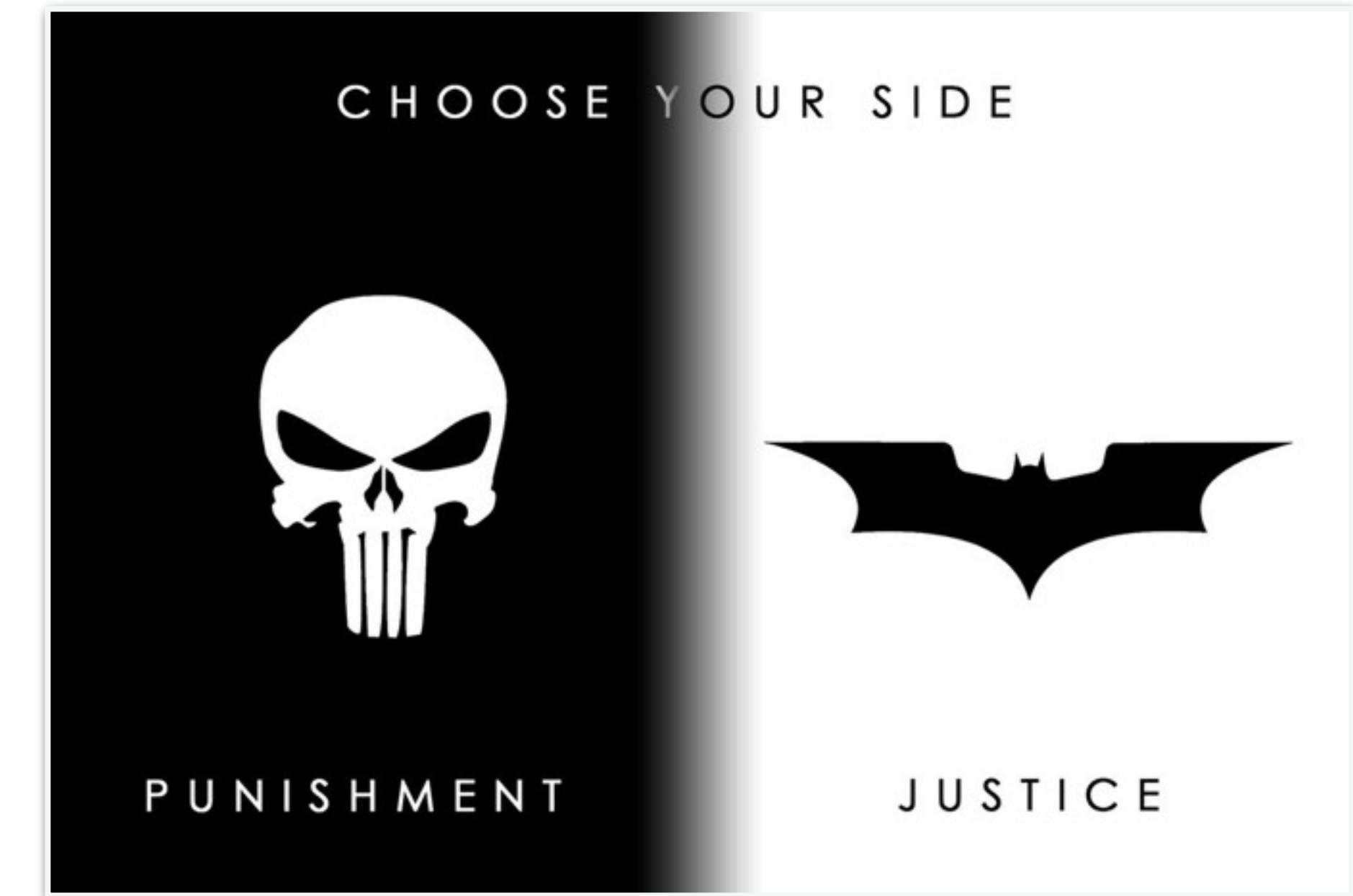


Cluster services

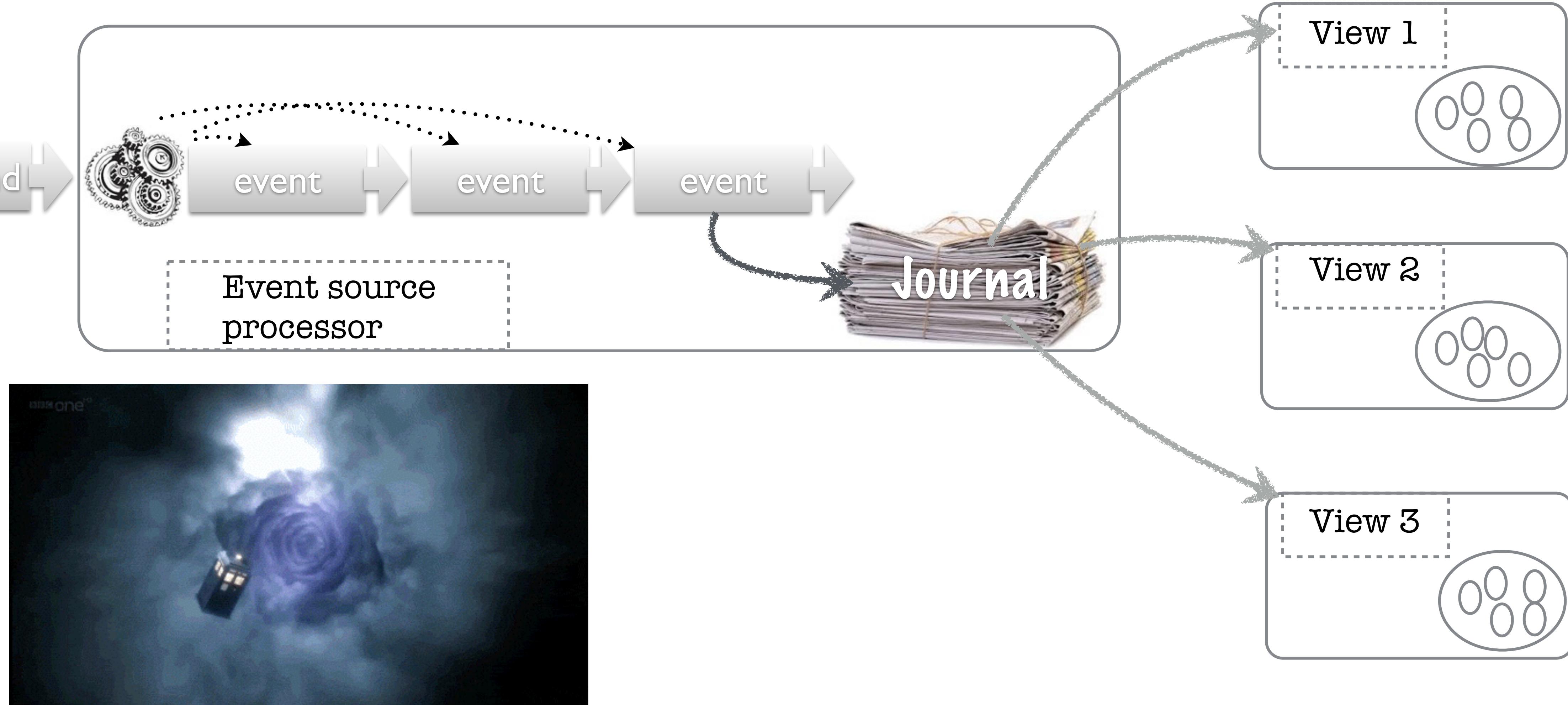


CQRS & EventSourcing

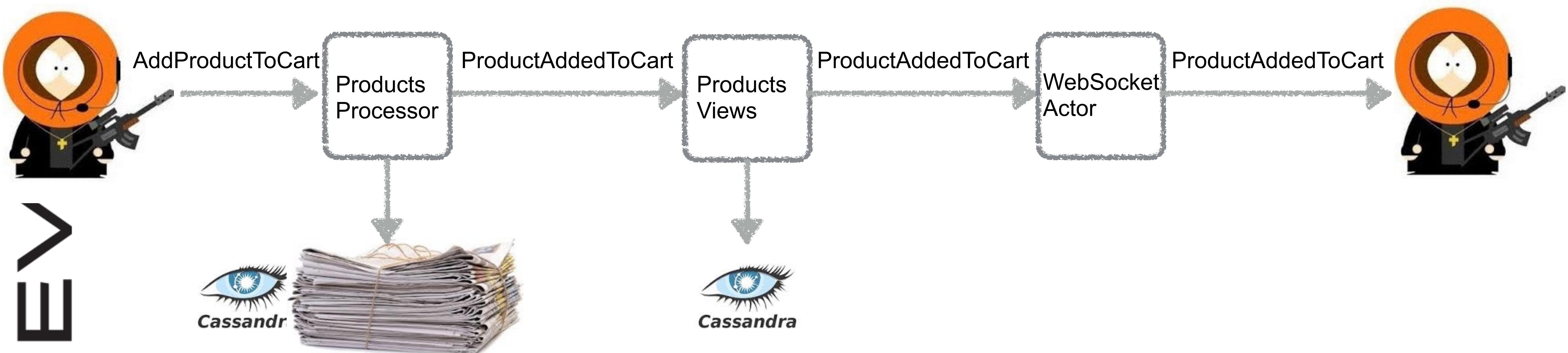
- Command Query Responsibility Segregation
 - Command : Enregistrement
 - Query : Lecture
- 2 modèles distincts
- Séparation des services
- Event sourcing
- Stockage des événements



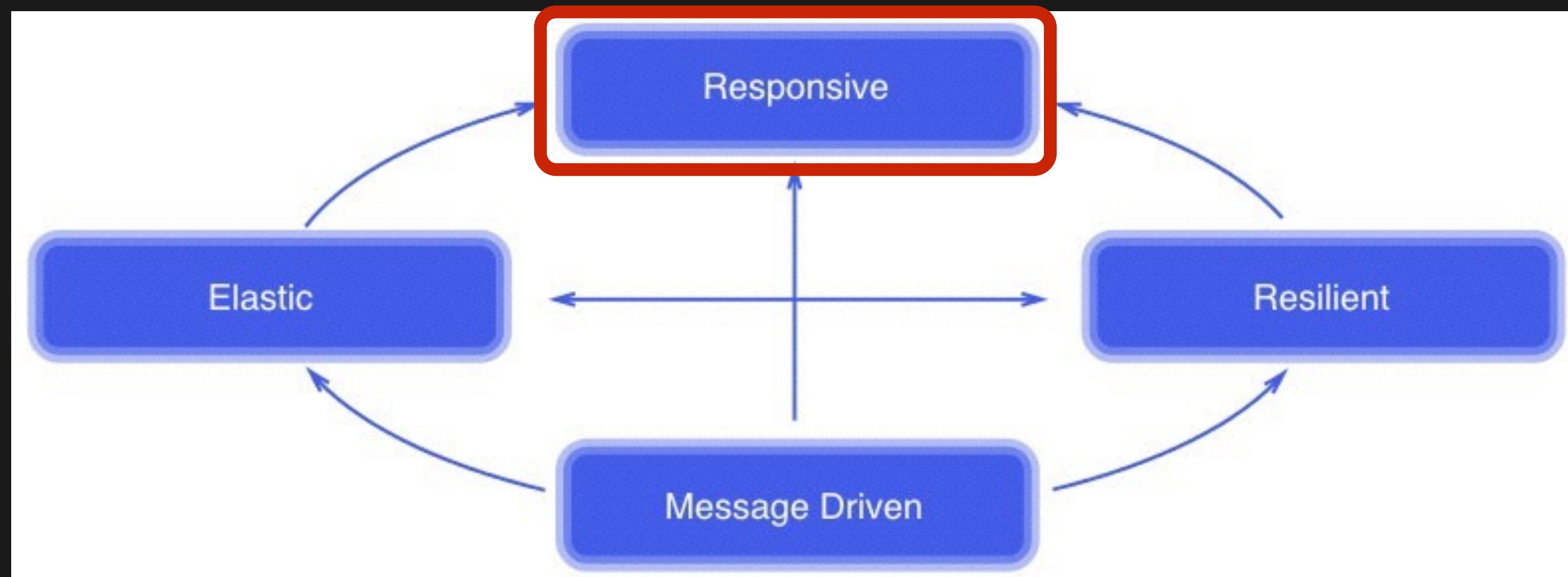
Persistence



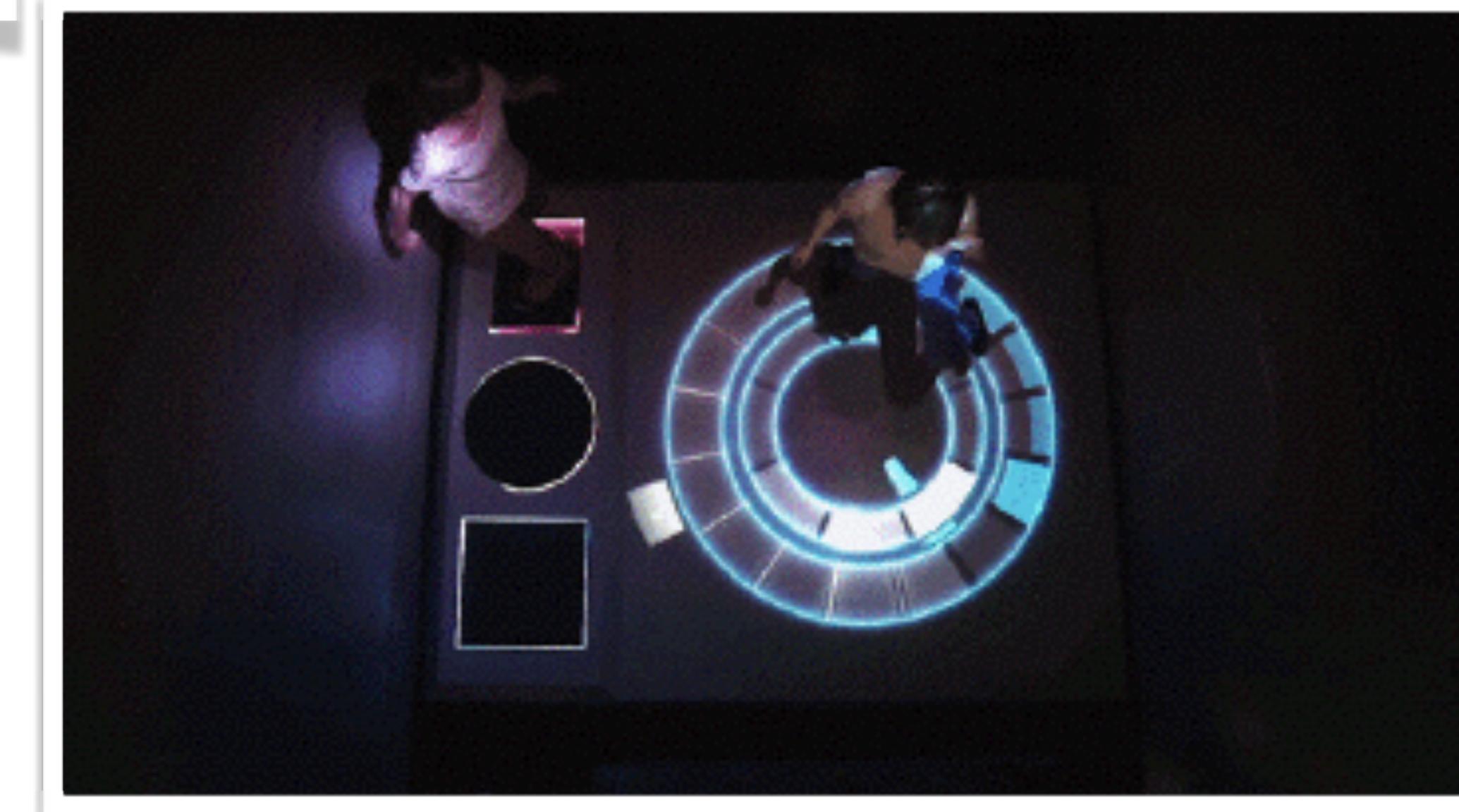
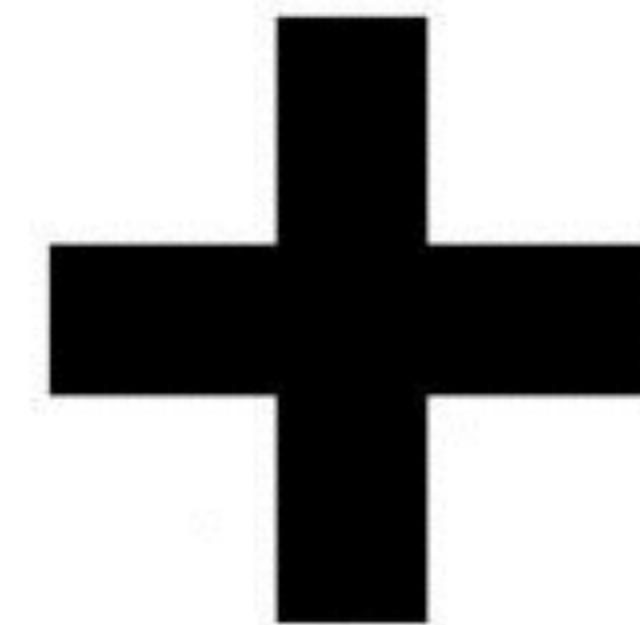
Exemple



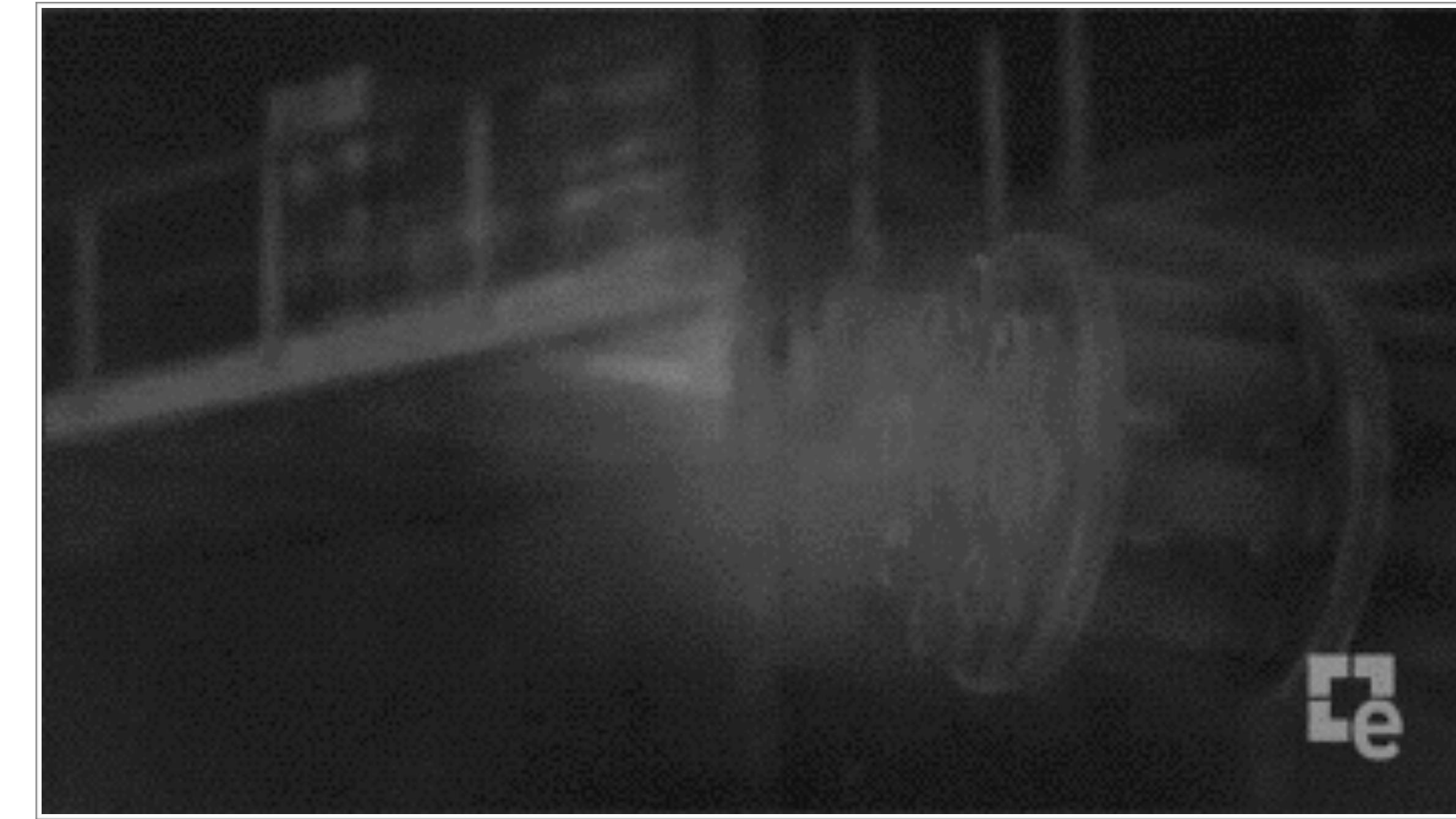
Responsive



Responsive



Et les perfs dans tout ca ?



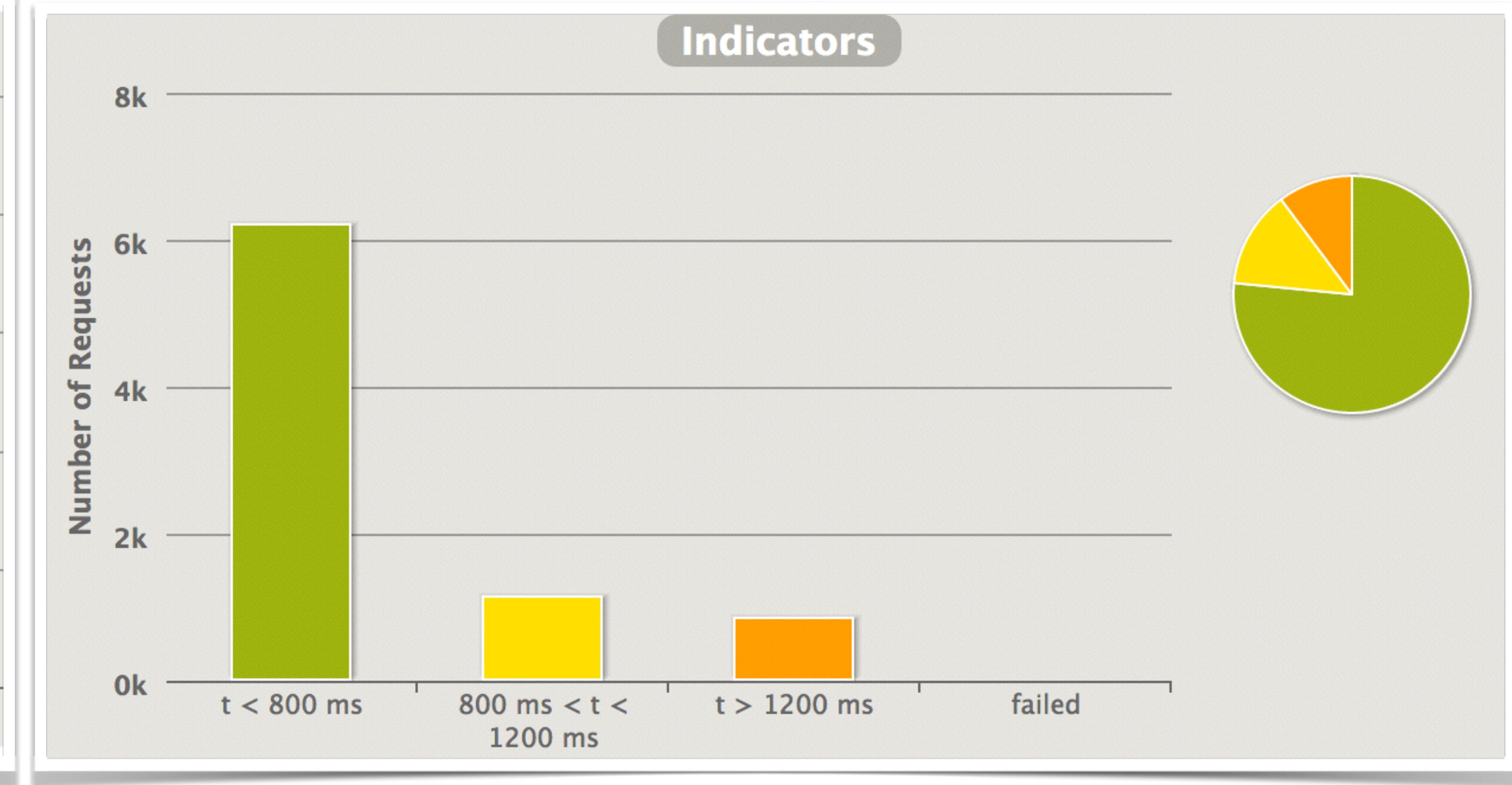
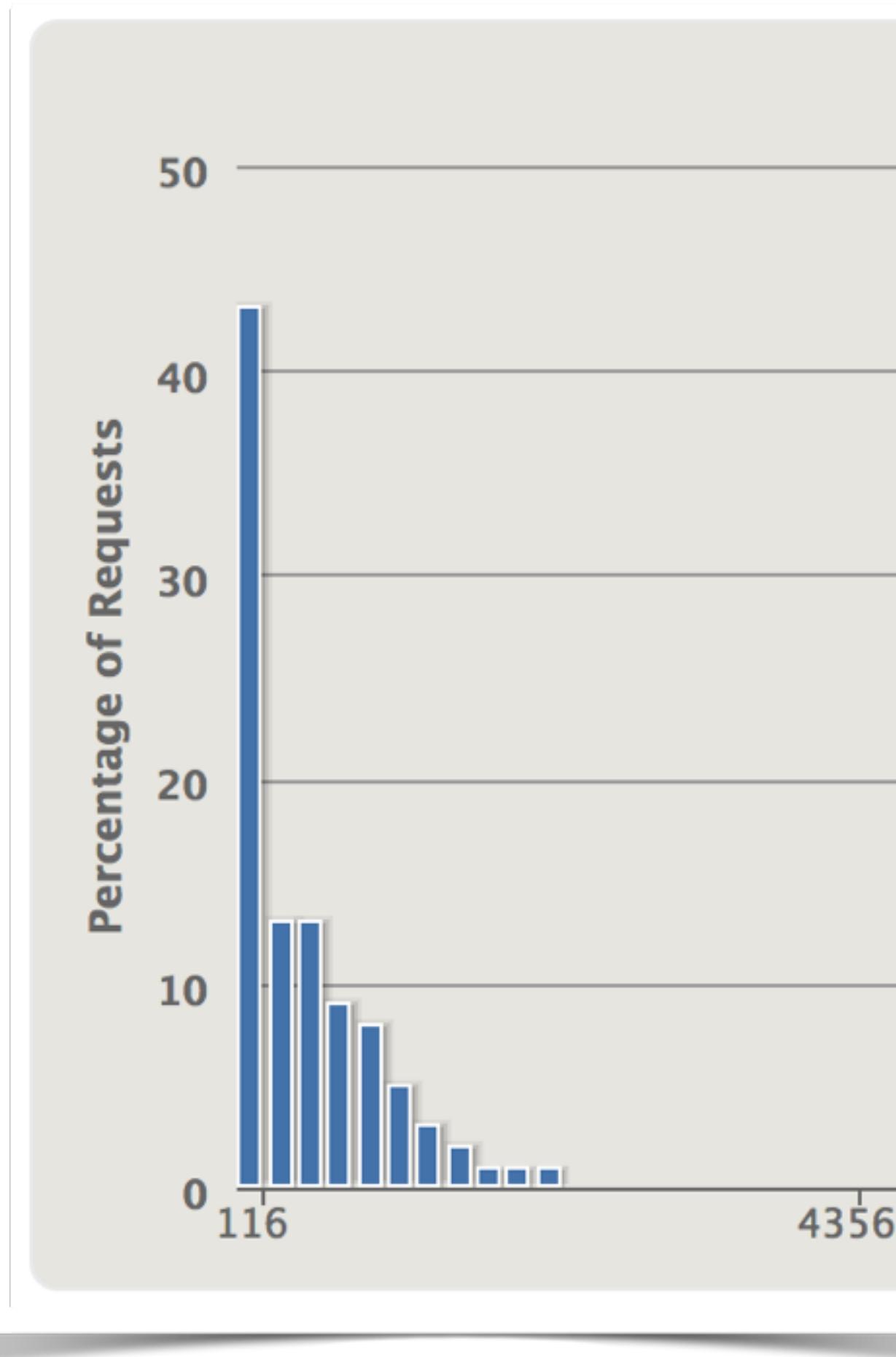
Les chiffres

367 998 456 756

Les chiffres

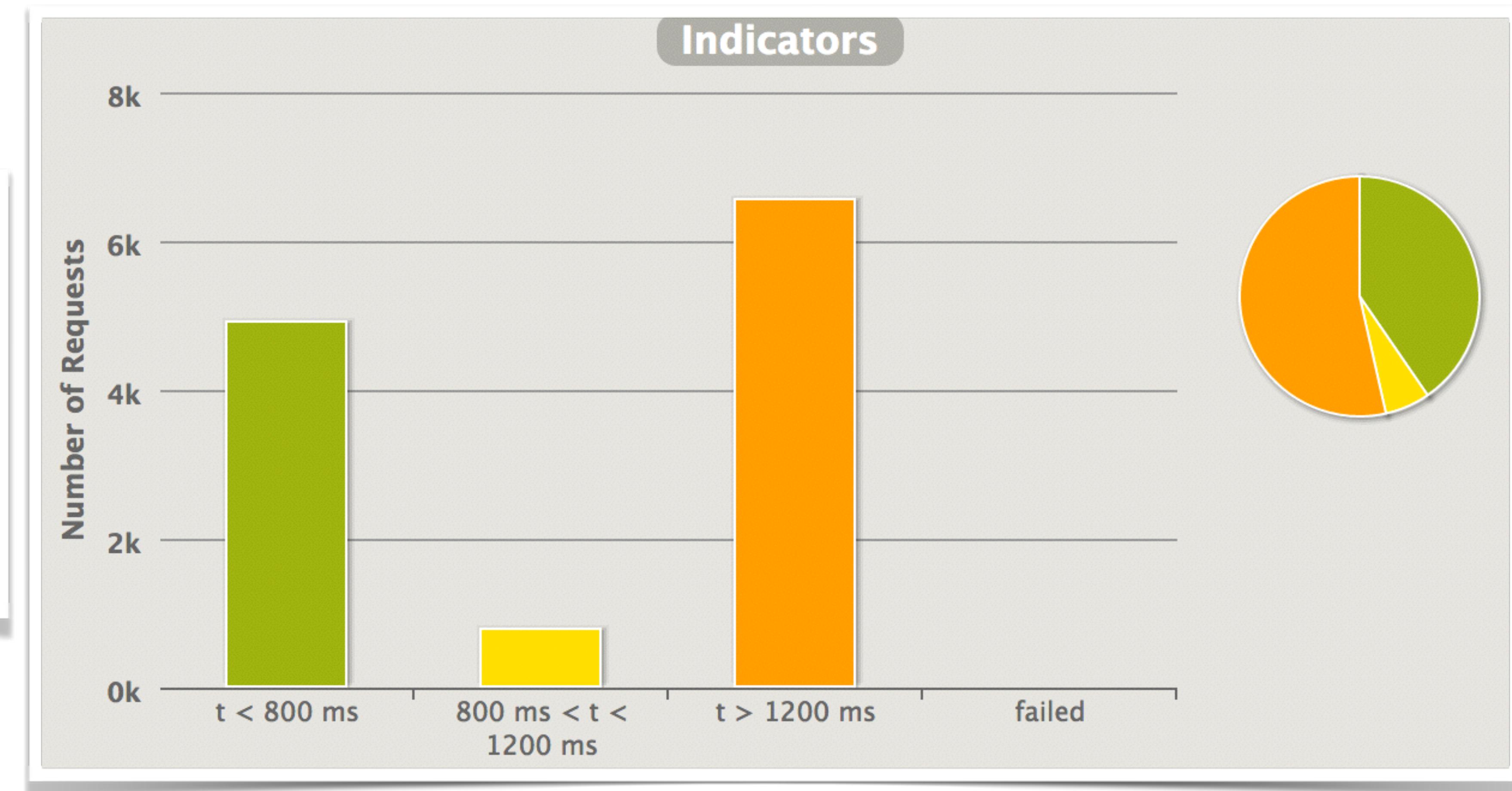
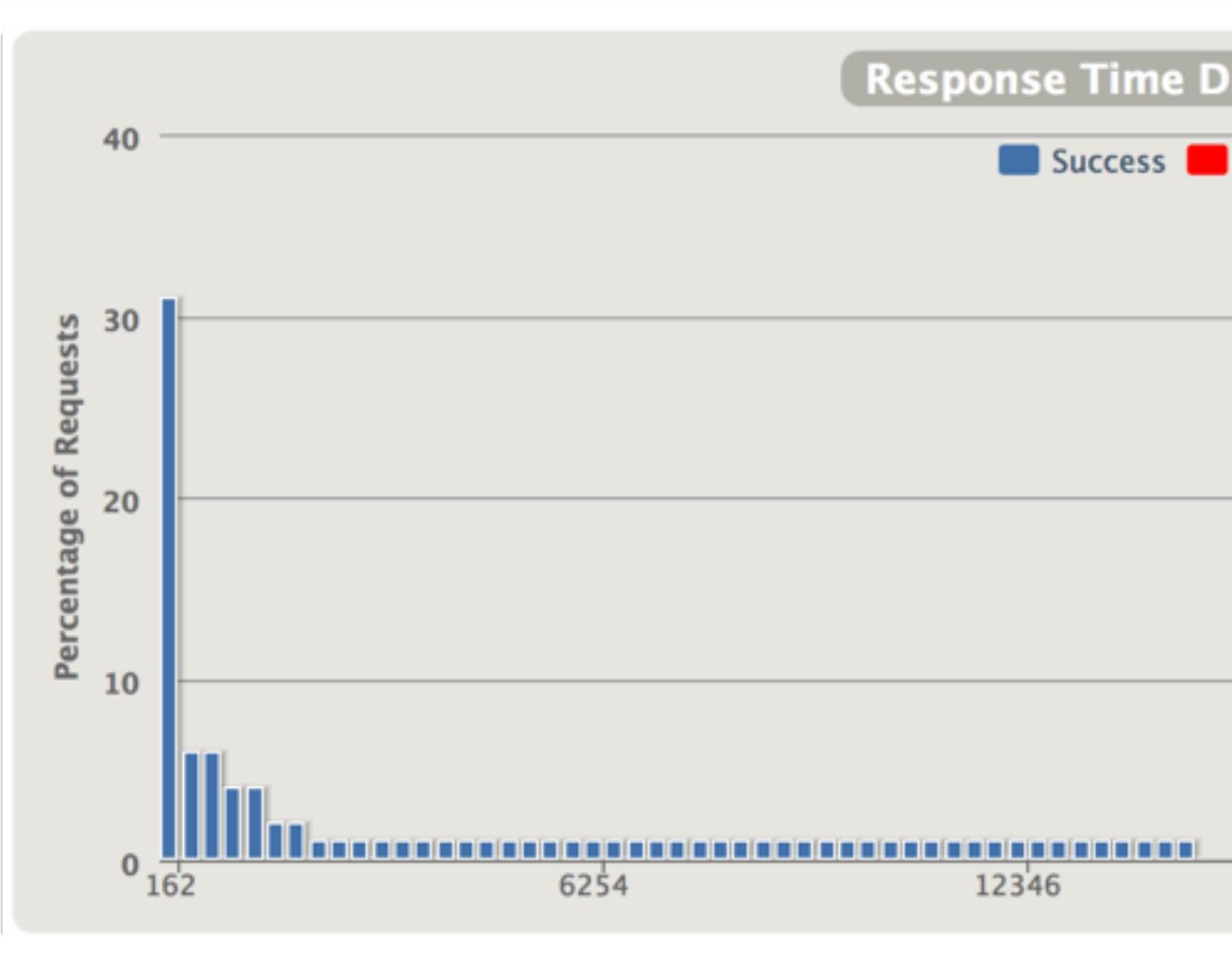
367 998 456 756 €

200 clients



400 clients

DEVONIX™
Software Development



Fr
an
ce

X
D
E
V
O
P

E
S
Q

Metrics



Cassandra

Messages akka

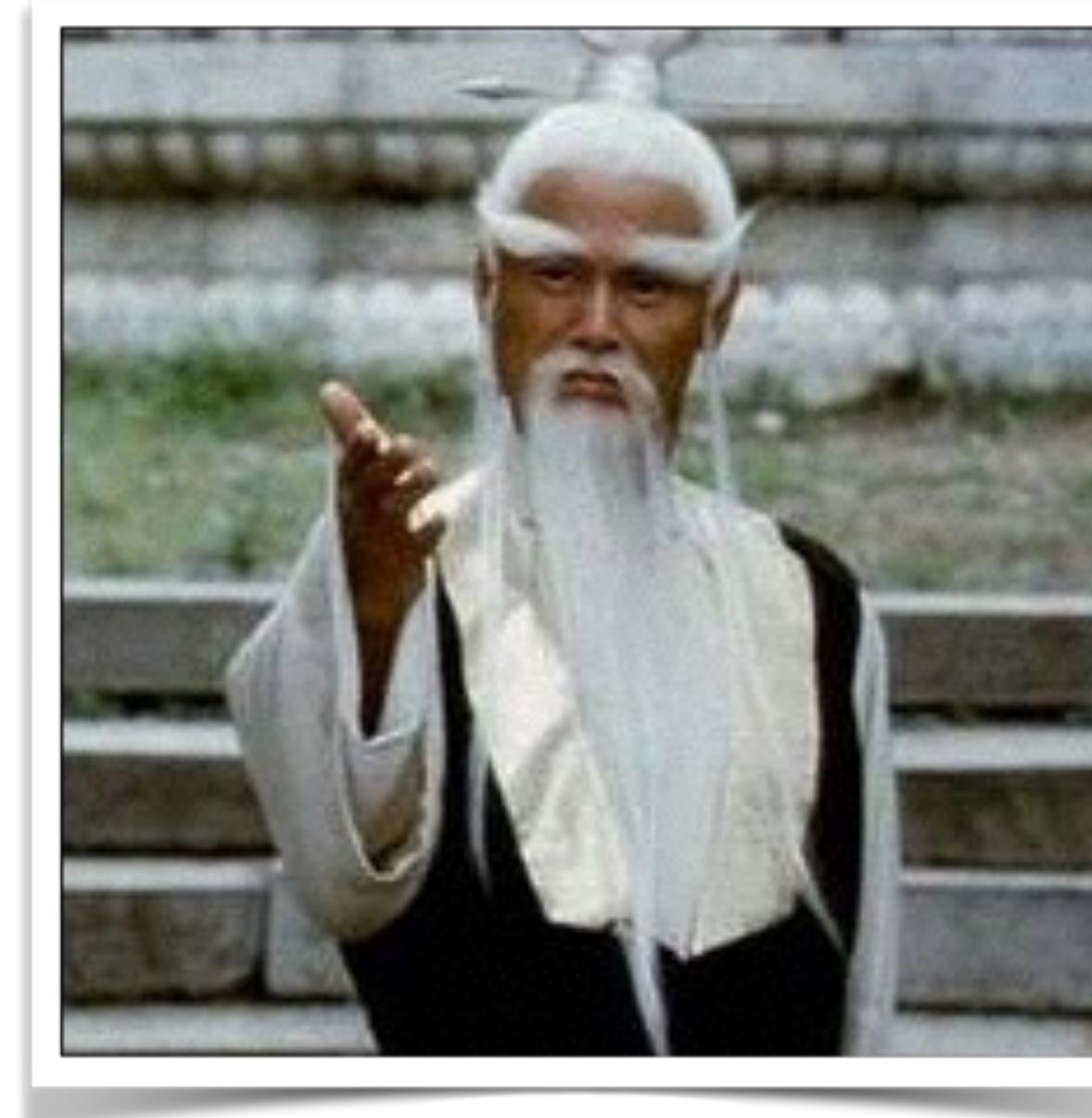
Journal

Elasticsearch

Les problèmes rencontrés



Nginx / mongo ...



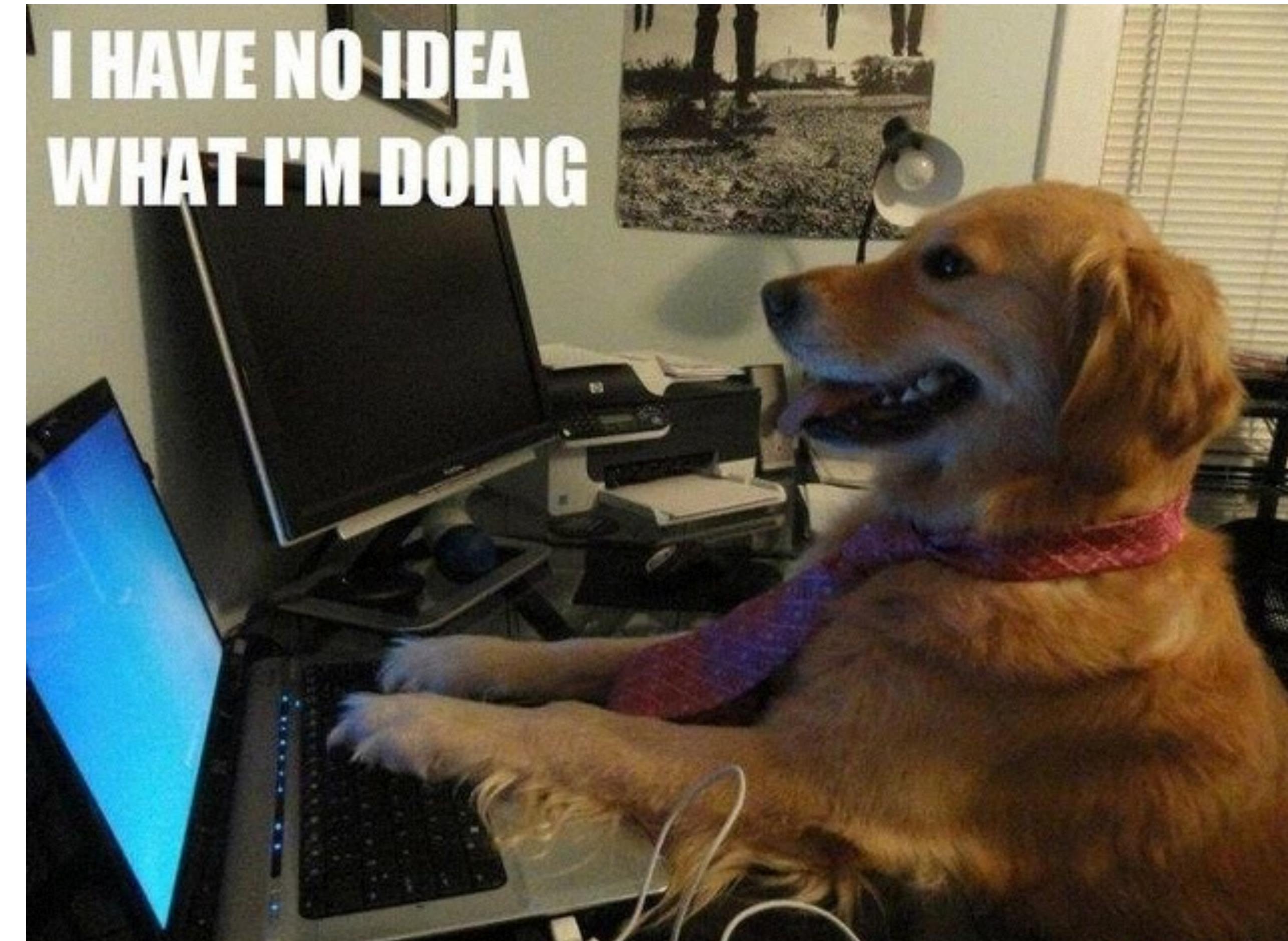
Cassandra



Bench web socket



Configuration akka-cluster



En java

- RX java : helper pour l'async, implémentation des reactive-streams
- Ratpack : framework http async pour java 8 ou groovy, implémentation des reactive-streams
- reactor : lib de messages sur la ivm, implémentation des reactive-streams
- vert.x : plateforme polyglote à la node js
- play et akka, implémentation des reactive-streams

Avec servlet et java 8

DEVVOXX France

```
public class AsyncServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        AsyncContext asyncContext = req.startAsync();
        //Traitement async
        CompletableFuture<String> futureString = CompletableFuture.supplyAsync(() -> "Reponse");
        futureString
            .thenAcceptAsync((string) -> {
                try {
                    asyncContext.getResponse().getWriter().print(string);
                } catch (IOException e) {
                    //Gérer l'erreur
                }
            })
            //Terminer :
            .whenCompleteAsync((aVoid, throwable) -> {
                asyncContext.complete();
            });
    }
}
```

Avec spring mvc et rx java

```
@SpringBootApplication
public class Application {

    @RequestMapping("asyncGet")
    public DeferredResult<String> asyncGet(){
        DeferredResult<String> deferredResult = new DeferredResult<>();
        Observable
            .from(CompletableFuture.supplyAsync(() -> "Reponse"))
            .map(String::toUpperCase)
            .subscribe(
                deferredResult:: setResult,
                deferredResult:: setErrorResult,
                () -> {}
            );
        return deferredResult;
    }
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

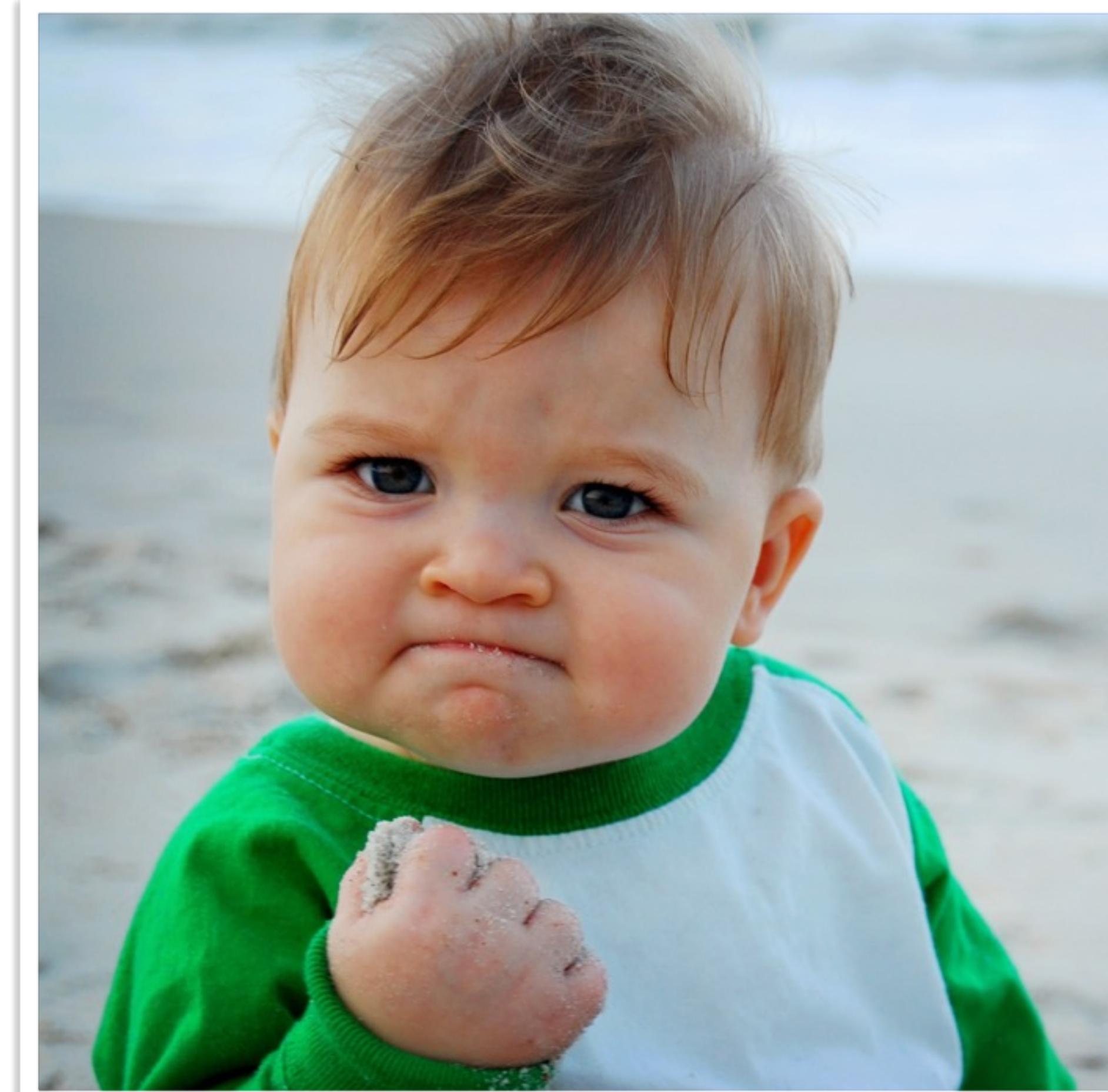
Liens

- Le projet
 - <https://github.com/mathieuancelin/reactive-raspberry-pi-cluster>
- Le modèle réactif
 - <http://www.reactivemanifesto.org/>
 - <http://reactive-streams.org>
 - <https://www.playframework.com>
 - <http://akka.io/>

Liens

- Nosql
 - <https://www.elastic.co/>
 - <http://cassandra.apache.org/>
- Systèmes distribués
 - <http://the-paper-trail.org/blog/distributed-systems-theory-for-the-distributed-systems-engineer>
- Exemple app reactives
 - <http://fr.lichess.org/>
 - <http://fr.lichess.org/monitor>

Conclusion



DEVOXX France

scala

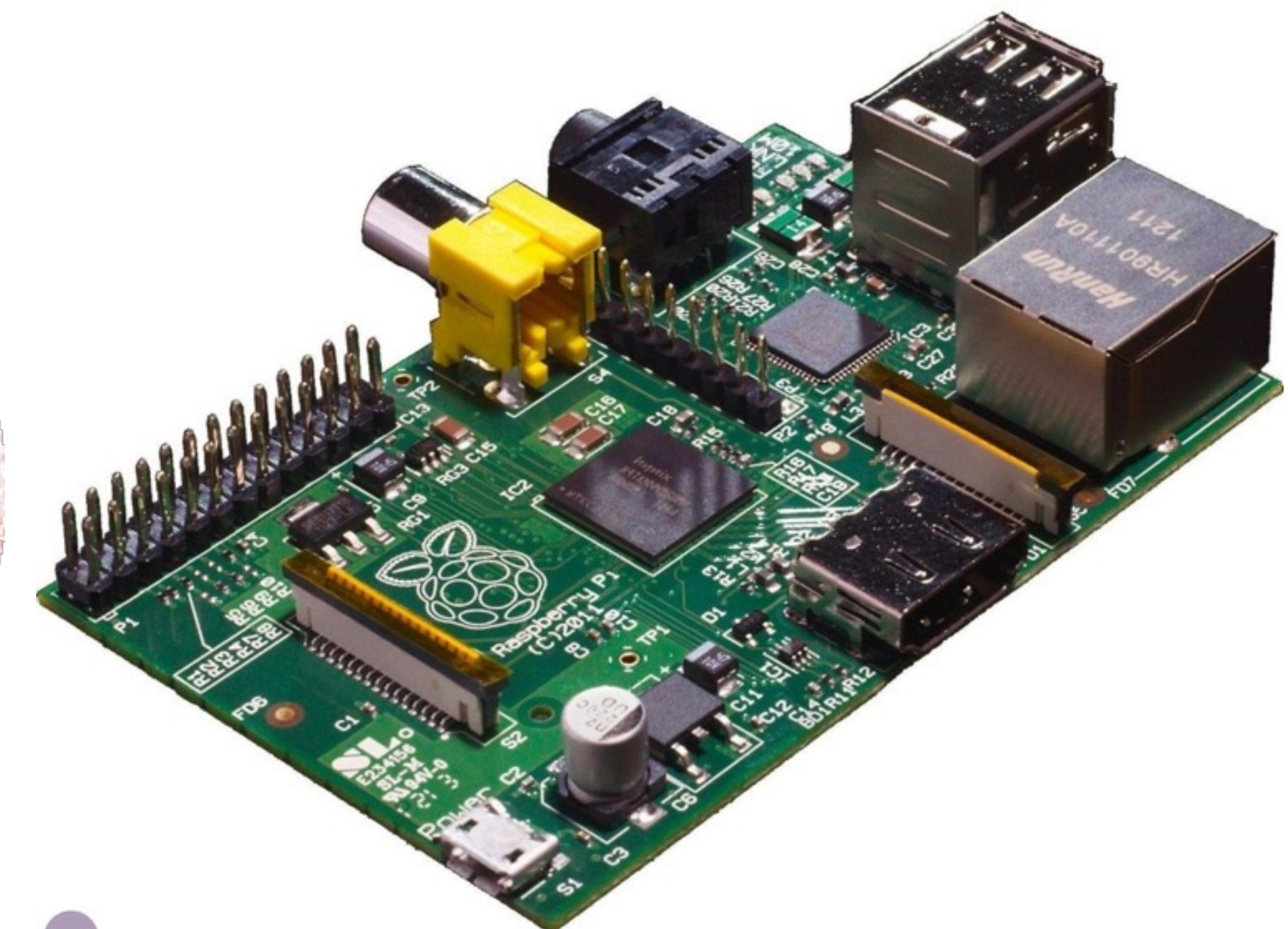


Scalable

play



net



#RaspberryCluster #Zombies

@TrevorReznik @chanksleroux

This is the end



#RaspberryCluster #Zombies

@TrevorReznik @chanksleroux