

Kids



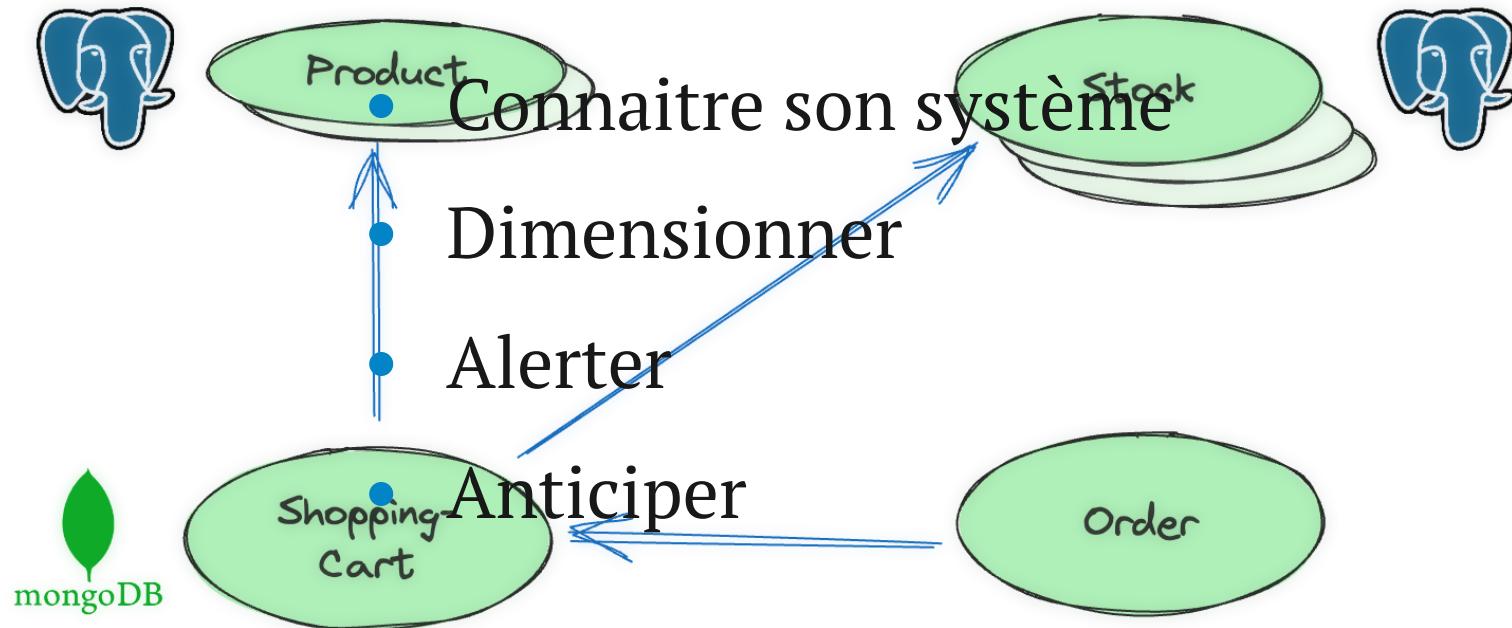
Vivien MALEZE

Technical Architect

- Background java ☕
- +10 ans d'xp
- +6 ans chez Ippon
- Bordeaux, France 🇫🇷
- Sujets du moments
 - Microservices ⚙️
 - DevOps 🔧
- 🐦 🐱 @vmaleze



Pourquoi moniturer ?



Logs

INFO -- [Mar 31 23:17:22] 127.0.0.1:58013 "GET /buddy_list.php HTTP/1.1" 200 189

INFO -- [Mar 31 23:17:22] 127.0.0.1:58013 "GET /presence/reconnect.php?__user=180001684819244&n=6&fb_dtsg=AQDJ95ij HTTP/1.1" 200 561

INFO -- [Mar 31 23:17:22] 127.0.0.1:58057 "GET /buddy_list.php HTTP/1.1" 200 189

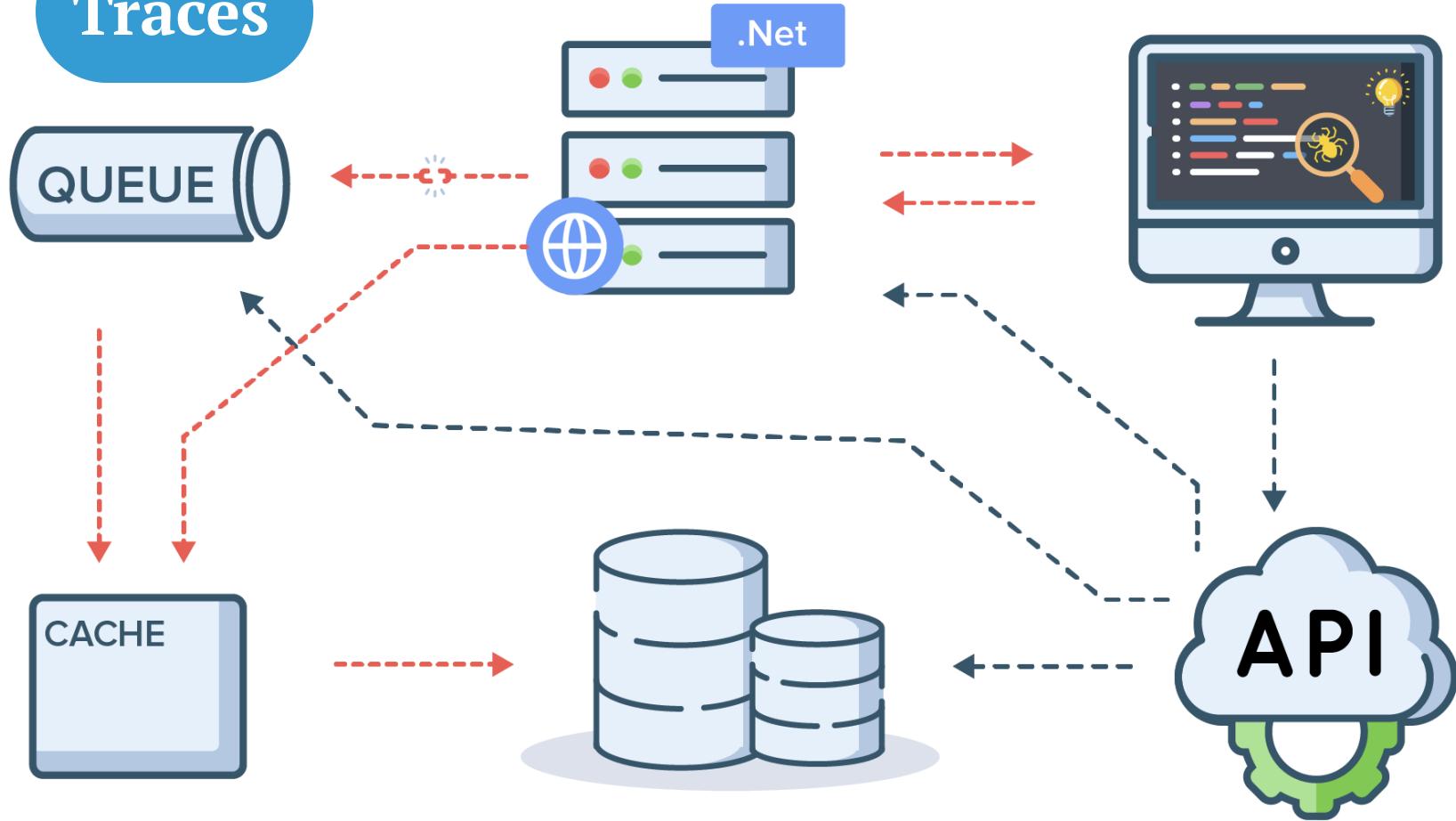
INFO -- [Mar 31 23:17:22] 127.0.0.1:58086 "GET /dPlayer.swf?v=1 HTTP/1.1" 200 21115

INFO -- [Mar 31 23:18:32] 127.0.0.1:58013 "GET /ping?partition=176&cb=12s6 HTTP/1.1" 200 127.0.0.1:58013 "HTTP/1.1" 200 98

Metrics



Traces



C'est quoi OpenTelemetry ?

C'est ✓

- Un projet open-source
- Un framework d'observabilité
- Un outil de collecte de données
- Un standard dans le monde du monitoring
- Une intégration dans la plupart des langages du marché

Ce n'est pas ✗

- Un outil tout en un comme datadog ou dynatrace
- Un backend de traitement de donnée

Pourquoi faire ?

- Un outil unique
- Pas besoin de changer son code si on change d'outil
- Simplifier l'instrumentation



Et comment on instrumente ?



```
//opentelemetry spring auto-configuration
implementation("io.opentelemetry.instrumentation:opentelemetry-spring-boot:OTEL_
//opentelemetry
implementation("io.opentelemetry:opentelemetry-api:OTEL_VERSION")
//opentelemetry exporter
implementation("io.opentelemetry:opentelemetry-exporters-otlp:OTEL_VERSION")
```



```
FROM openjdk:17

RUN mkdir -p /app/bin

ADD https://github.com/open-telemetry/opentelemetry-java-instrumentation/
    releases/latest/download/opentelemetry-javaagent.jar
    /app/bin/opentelemetry-javaagent.jar

COPY target/*.jar /app/bin/app.jar

CMD java -javaagent:/app/bin/opentelemetry-javaagent.jar \
    -jar /app/bin/app.jar
```

Comment ça marche ?



Vers où exporte les données ?

Vers les backend directement

OTEL_SERVICE_NAME

Nom du service

OTEL_TRACES_EXPORTER / OTEL_EXPORTER_OTLP_TRACES_ENDPOINT

Export des traces distribuées

OTEL_METRICS_EXPORTER /
OTEL_EXPORTER_OTLP_METRICS_ENDPOINT

Export des metrics

OTEL_LOGS_EXPORTER / OTEL_EXPORTER_OTLP_LOGS_ENDPOINT

Export des logs

Vers où exporte-t'on les données ?

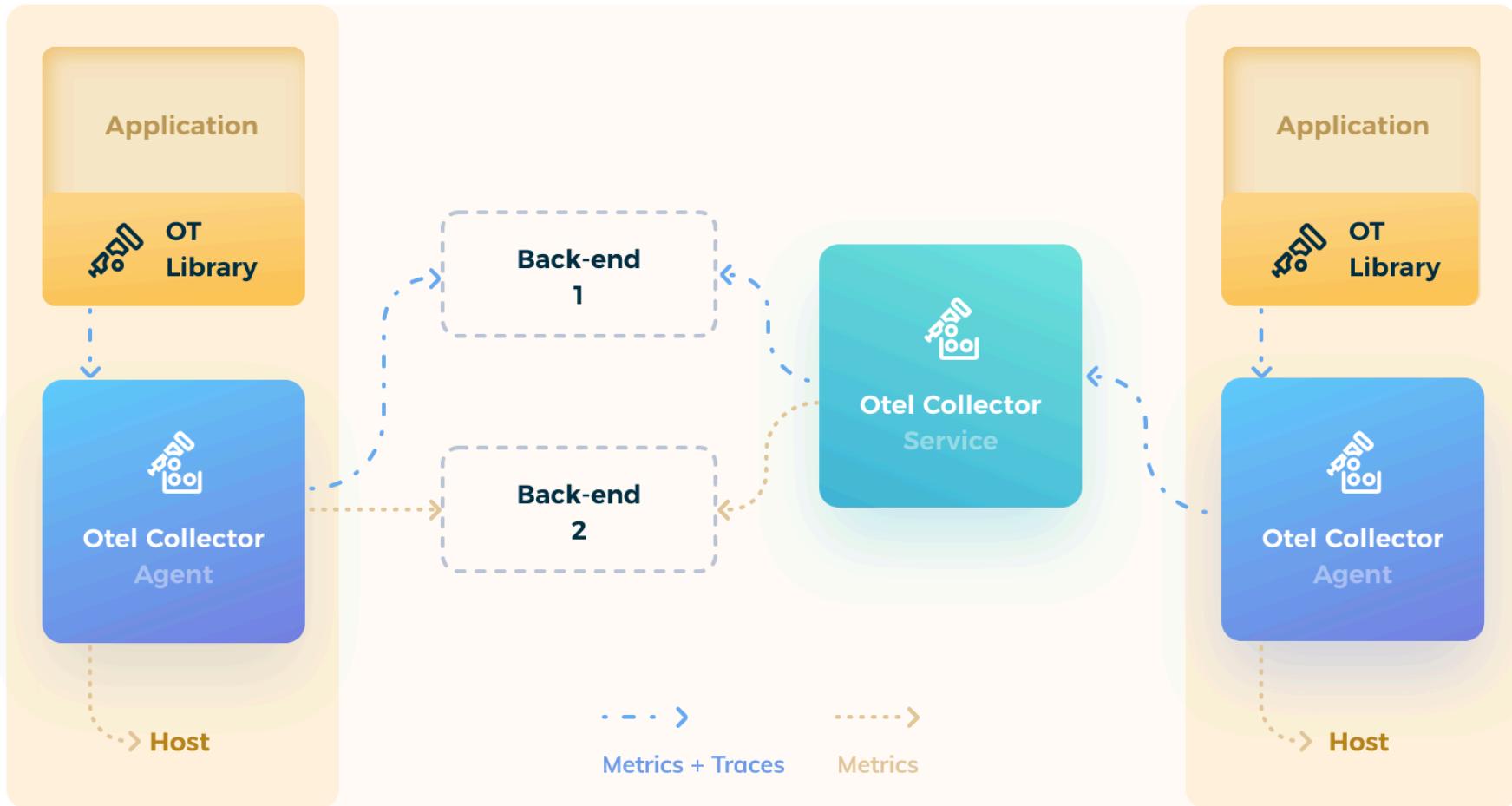
Vers un collecteur opentelemetry

OTEL_SERVICE_NAME

Nom du service

OTEL_EXPORTER_OTLP_ENDPOINT

Collecteur opentelemetry



Au travail



On ajoute quoi au cluster ?

Operator

Manage les resources OpenTelemetry via les CRDs k8s

Collector

Reçoit et exporte les données

Auto Instrumentation

Injecte l'agent java directement dans le conteneur docker

Le collecteur

Déploiement et configuration

- Mode de déploiement
 - Sidecar
 - Daemonset
- Collecte des données
 - Push / Pull
- Export des données
 - Liste des différents backend
 - Exposition de données
- Config des différentes services

```
1 apiVersion: opentelemetry.io/v1alpha1
2 kind: OpenTelemetryCollector
3 metadata:
4   name: otel-sidecar
5 spec:
6   mode: sidecar
7   config: |
8
9     receivers:
10       otlp:
11         protocols:
12           grpc:
13           http:
14     processors:
15       batch:
```

Injection de l'agent

Auto Instrumentation

```
apiVersion: opentelemetry.io/v1alpha1
kind: Instrumentation
metadata:
  name: otel-auto-instrumentation
spec:
  exporter:
    endpoint: http://0.0.0.0:4317
```

On applique ça sur le déploiement

- Ajout du sidecar
- Injection de l'agent java

```
1 template:  
2   metadata:  
3     annotations:  
4       sidecar.opentelemetry.io/inject: "true"  
5       instrumentation.opentelemetry.io/inject-java: "true"
```

A stylized cartoon character with spiky black hair and a wide, toothy grin. He is wearing an orange long-sleeved shirt with white horizontal stripes across the chest. His arms are raised in excitement.

Et c'est parti !

HELLO IT
DID YOU TRY TURNING
IT OFF AND ON AGAIN?

Y'a quoi derrière tout ça ?



Pour les traces

Tempo

- Outil de gestion des traces distribuées
- Basé sur le standard opentelemetry
- Compatible avec les formats open source
 - Zipkin
- Intégration poussée avec grafana

Grafana Tempo

Pour les logs

Loki

- Outil d'aggregation de logs
- Parse les entrées pour en extraire les informations importantes
 - Dates
 - Erreurs
 - Traces
- Basé sur un collecteur de logs tels que
 - FluentBit/FluentD
 - FileLogReceiver



Pour les metrics

Prometheus



- Système de monitoring et d'alerting basé sur des metrics
- *Scrape* les données plutôt que de les recevoir
- Gère les données grâce à un format basé sur les time-series

Prometheus

Le scaling automagique

Horizontal Pod Autoscaler

- Nombre de replicas
- Sur quoi on se base ?
 - Metrics de reference
 - Valeur moyenne
- Comportement custom



```
1 apiVersion: autoscaling/v2
2 kind: HorizontalPodAutoscaler
3 ...
4   minReplicas: 1
5   maxReplicas: 10
6   metrics:
7     - type: Pods
8       pods:
9         metric:
10           name: http_server_requests_per_seconds
11           target:
12             type: AverageValue
13             averageValue: "8"
14   behavior:
15     scaleDown:
16       stabilizationWindowSeconds: 60
```




www.ippon.fr

contact@ippon.fr — +33 1 46 12 48 48 — @ipponTech

