SODIFRANCE/NETAPSYS VOUS SOUHAITE LA BIENVENUE

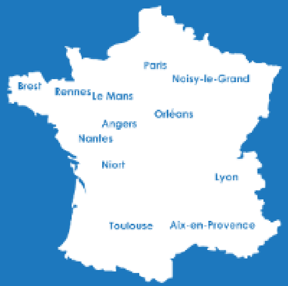
Spécialiste de la Transformation des SI

1300

COLLABORATEURS

16

IMPLANTATIONS



+150

Projets réalisés

Précurseur des architectures micros services

+100

M€ d'investissements
en R&D
depuis 20 ans

Plusieurs milliards de lignes de code traitées

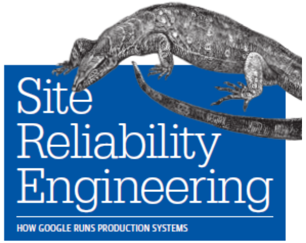
Des solutions présentes dans

14 pays

Sodifrance
IT transformation to digital

Le traitement des mesures des applications cloud-natives

MOST METRICS ARE BETTER THOUGHT OF AS *DISTRIBUTIONS* RATHER THAN AVERAGES


Site Reliability Engineering
HOW GOOGLE RUNS PRODUCTION SYSTEMS

“ Monitoring and alerting based only on the average latency would show no change in behavior ”

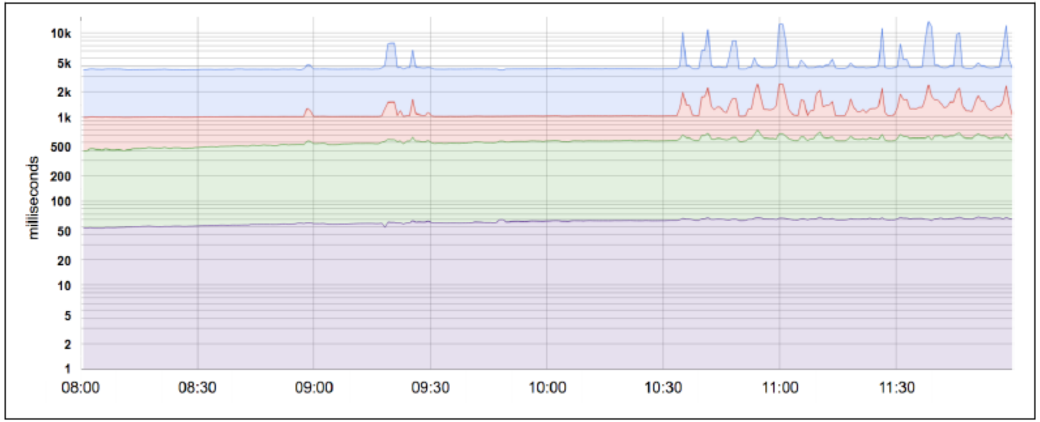


Figure 4-1. 50th, 85th, 95th, and 99th percentile latencies for a system. Note that the Y-axis has a logarithmic scale.

“ Using percentiles for indicators allows you to consider the shape of the distribution and its differing attributes: a high-order percentile, such as the ***99th or 99.9th, shows you a plausible worst-case value****, while using the 50th percentile (also known as the median) emphasizes the typical case****. The higher the variance in response times, the more the typical user experience is affected by long-tail behavior****, an effect exacerbated at high load by* ***queuing effects****. User studies have shown that people typically prefer a slightly slower system to one with high variance in response time, so some* ***SRE teams focus only on high percentile values, on the grounds that if the 99.9th percentile behavior is good, then the typical experience is certainly going to be.****”*

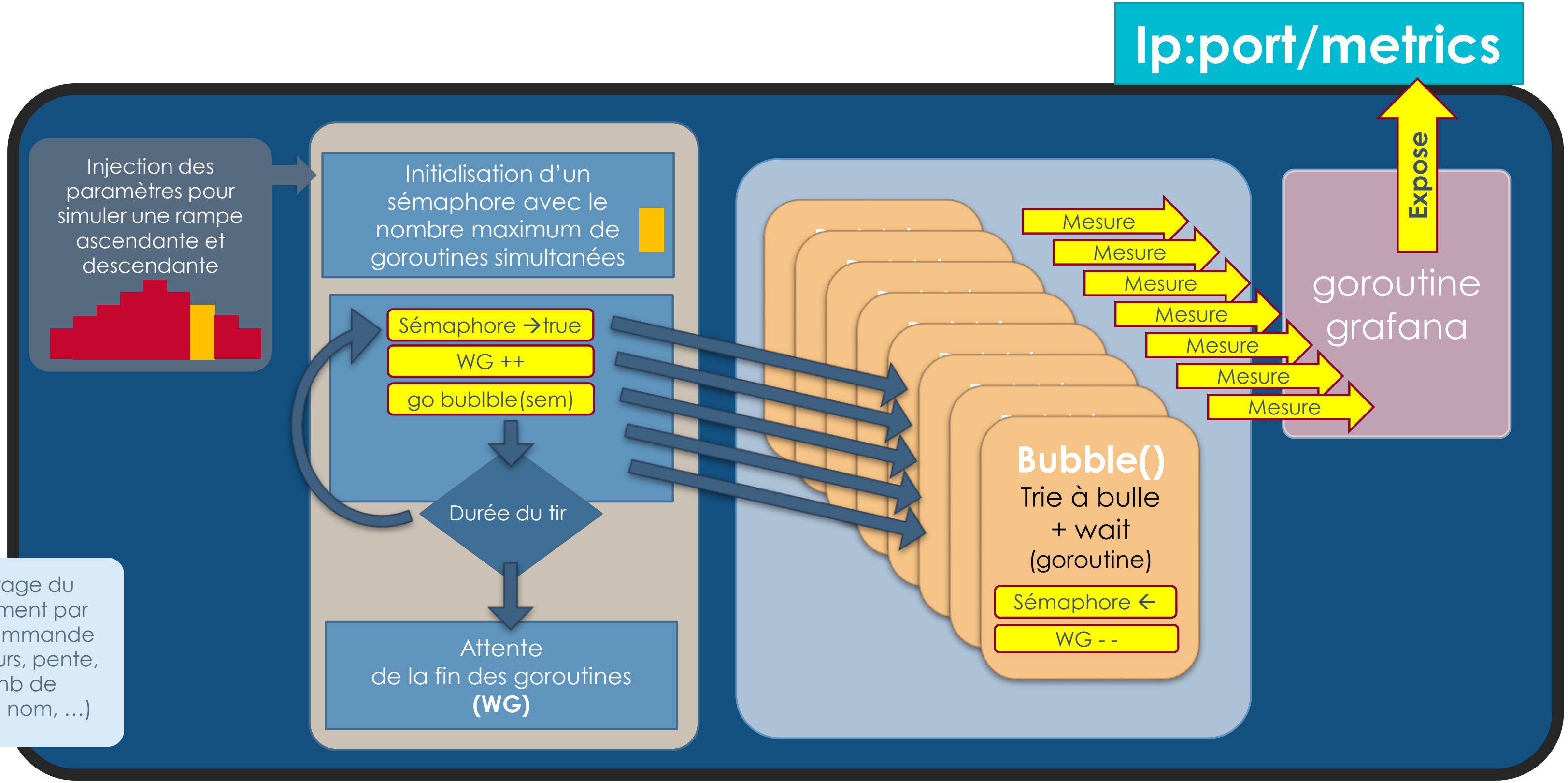
ROB PIKE'S 5 RULES OF PROGRAMMING

- **Rule 1.** You can't tell where a program is going to spend its time. Bottlenecks occur in surprising places, so don't try to second guess and put in a speed hack until you've proven that's where the bottleneck is.
- **Rule 2.** Measure. Don't tune for speed until you've measured, and even then don't unless one part of the code overwhelms the rest.
- **Rule 3.** Fancy algorithms are slow when n is small, and n is usually small. Fancy algorithms have big constants. Until you know that n is frequently going to be big, don't get fancy. (Even if n does get big, use Rule 2 first.)
- **Rule 4.** Fancy algorithms are buggier than simple ones, and they're much harder to implement. Use simple algorithms as well as simple data structures.
- **Rule 5.** Data dominates. If you've chosen the right data structures and organized things well, the algorithms will almost always be self-evident. Data structures, not algorithms, are central to programming.

Pike's rules 1 and 2 restate Tony Hoare's famous maxim "Premature optimization is the root of all evil." Ken Thompson rephrased Pike's rules 3 and 4 as "When in doubt, use brute force.". Rules 3 and 4 are instances of the design philosophy KISS. Rule 5 was previously stated by Fred Brooks in The Mythical Man-Month. Rule 5 is often shortened to "write stupid code that uses smart objects".

http://users.ece.utexas.edu/~adnan/pike.html

SYNOPTIQUE DU SIMULATEUR

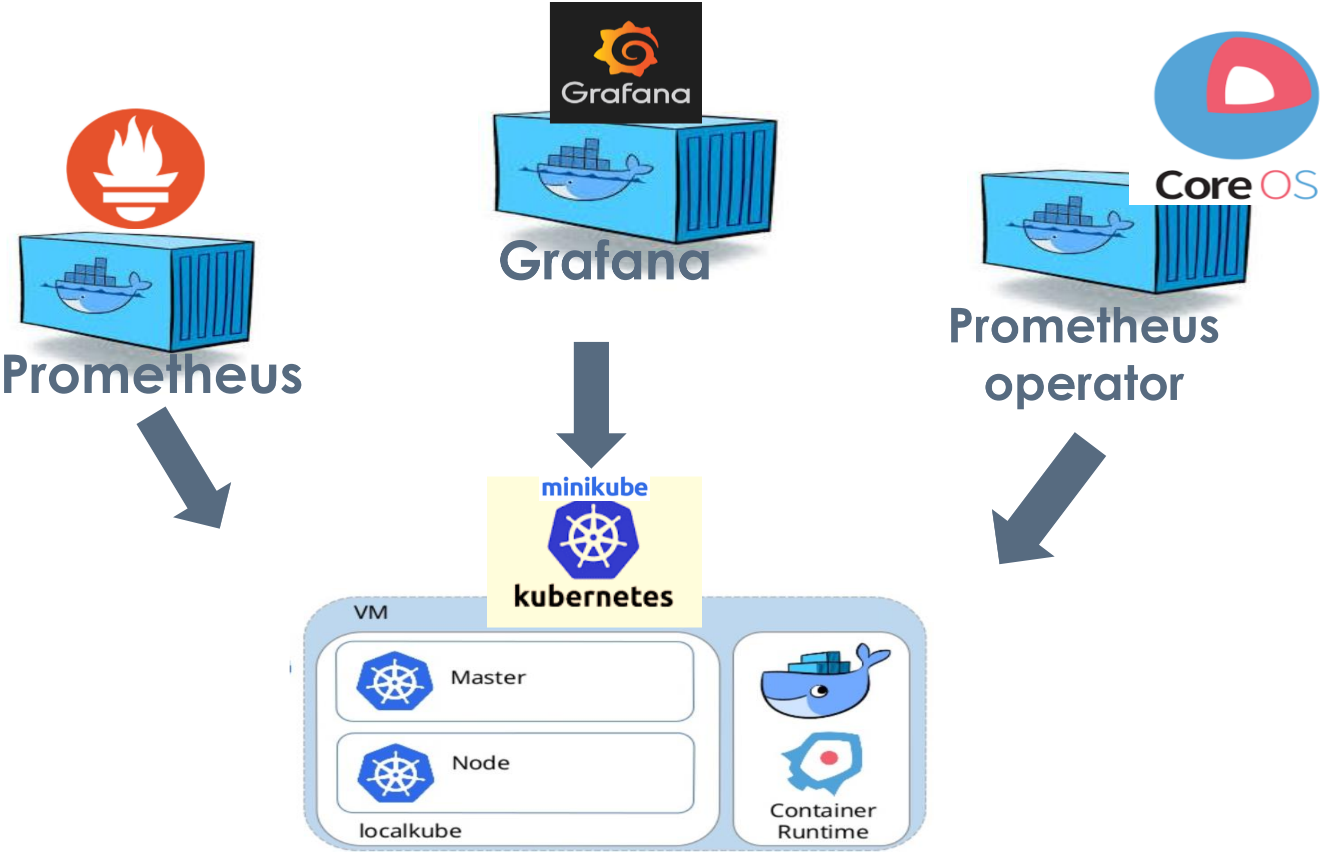


Ip:port/metrics
Injection des paramètres pour simuler une rampe ascendante et descendante
Initialisation d'un sémaphore avec le nombre maximum de goroutines simultanées
Sémaphore →true
WG ++
go bublble(sem)
Durée du tir
Attente de la fin des goroutines **(WG)**
Bubble()
Trie à bulle + wait (goroutine)
Sémaphore ←
WG - -
Mesure
Mesure
Mesure
Mesure
Mesure
Mesure
Mesure
goroutine grafana
Expose

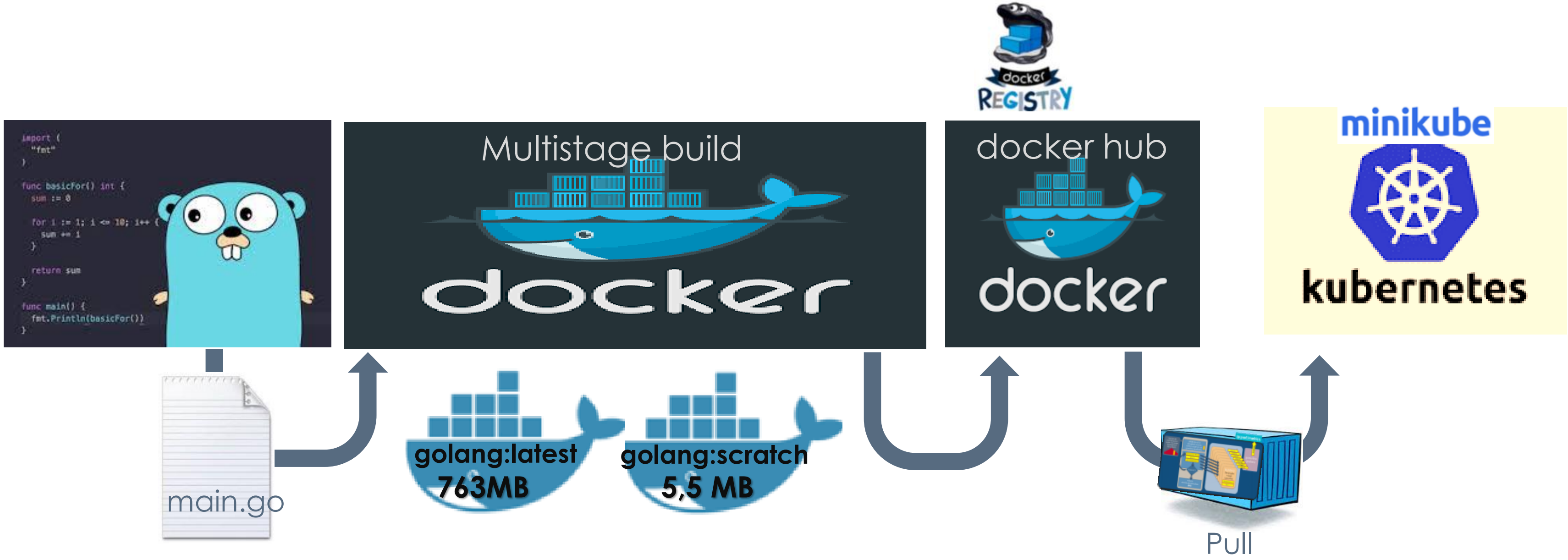
Paramétrage du comportement par ligne de commande (nb de cœurs, pente, durée, nb de goroutines, nom, …)

DEMO

BOOTSTRAP DU CLUSTER

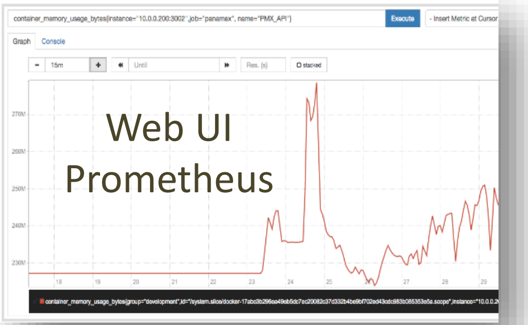


DU CODE AU DÉPLOIEMENT DANS LE CLUSTER



```
import (
  "fmt"
)

func basicFor() int {
  sum := 0

  for i := 1; i <= 10; i++ {
    sum += i
  }

  return sum
}

func main() {
  fmt.Println(basicFor())
}
```

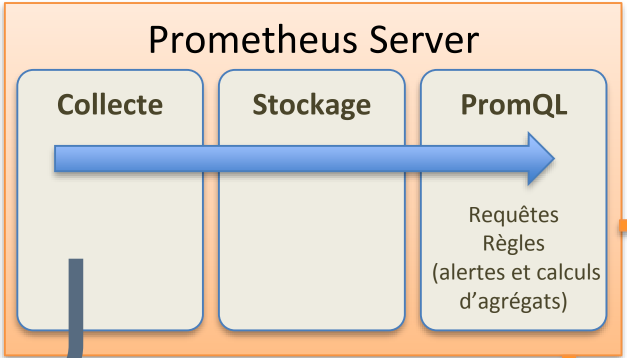
DU CLUSTER AU TABLEAU DE BORD



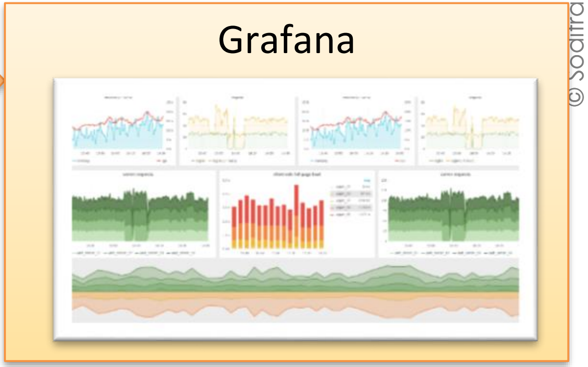
```
container_memory_usage_bytes{instance="10.0.0.200:3002",job="panamax", name="PMX_API"}
```

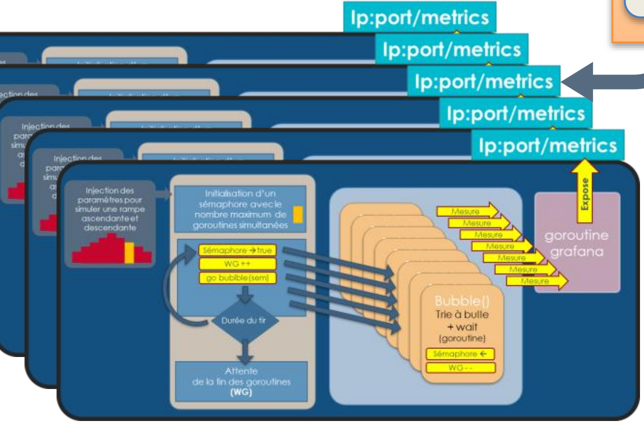
container_memory_usage_bytes{group="development",id="/system.slice/docker-17abc3b296ea49eb5dc7ec20082c37d332b4be9bf702ed43cdc683b086353e5a.scope",instance="10.0.0.2

Web UI Prometheus



Requêtes
Règles
(alertes et calculs d'agrégats)

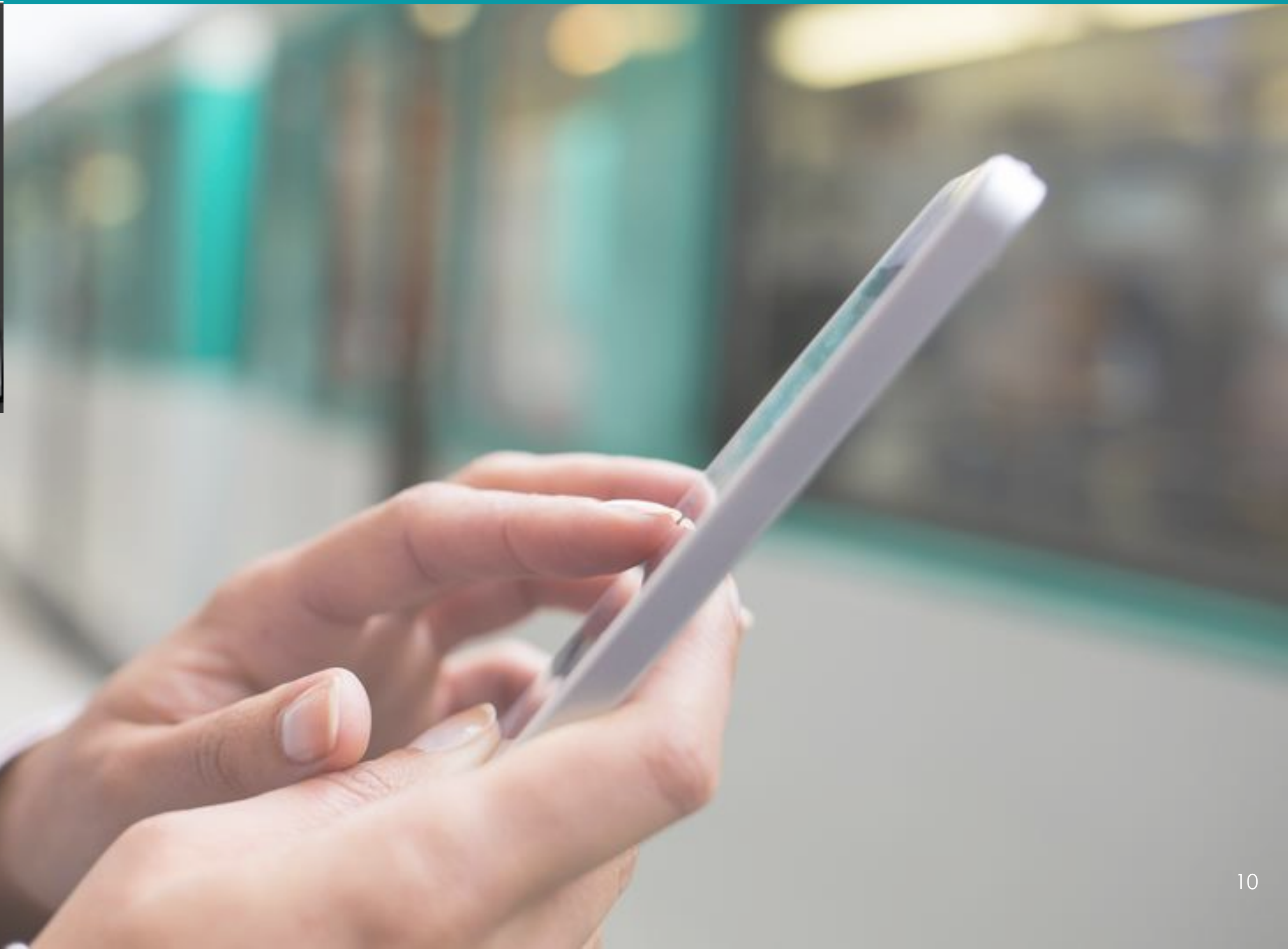


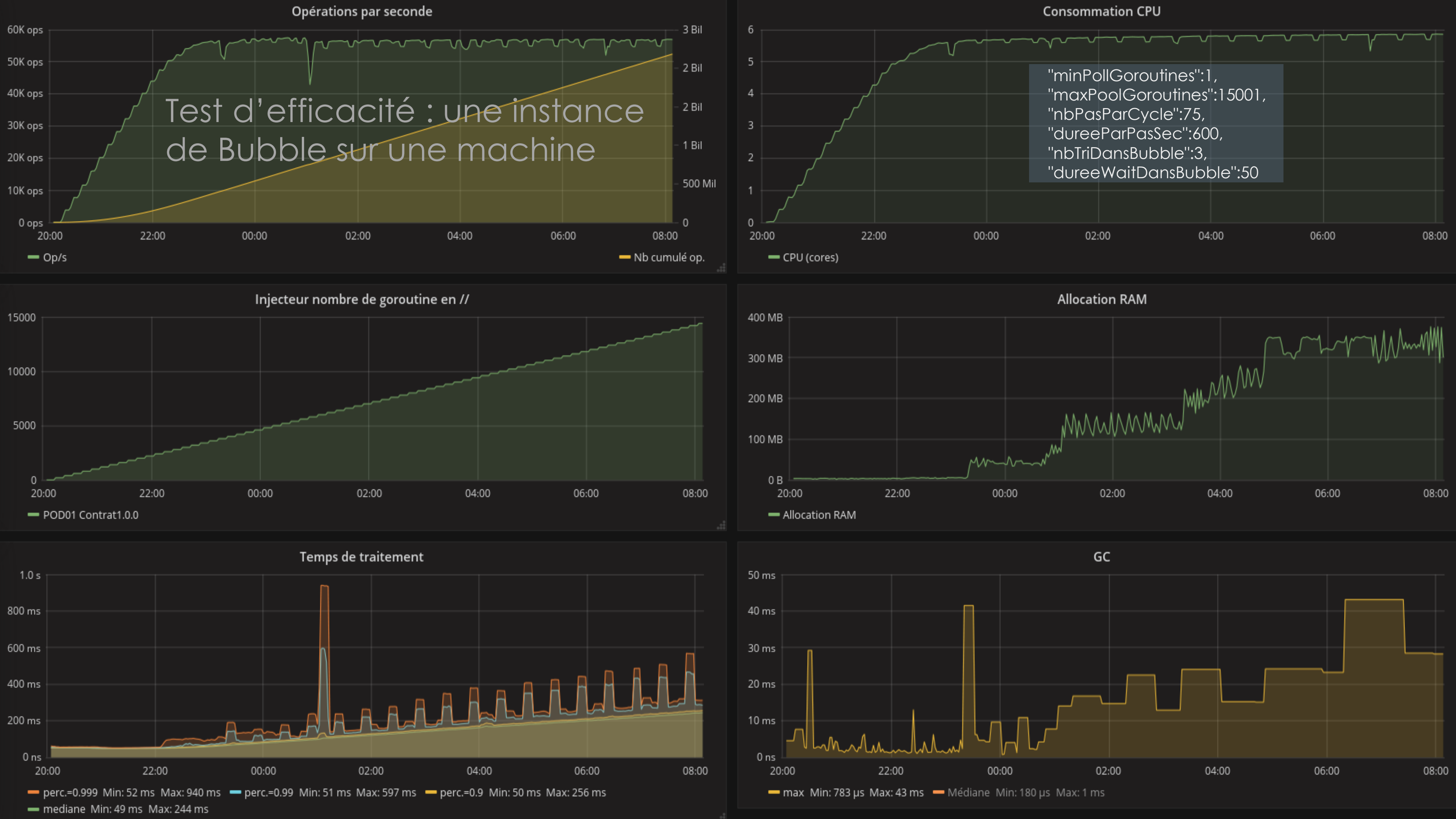


Ip:port/metrics
Ip:port/metrics
Ip:port/metrics
Ip:port/metrics
Ip:port/metrics
Expose
Injection des paramètres pour simuler une rampe ascendante et descendante
Initialisation d'un sémaphore avec le nombre maximum de goroutines simultanées
Sémaphore → true
WG ++
go bublble(sem)
Durée du tir
Attente de la fin des goroutines (WG)
Bubble()
Trie à bulle + wait (goroutine)
Sémaphore ←
WG - -
Mesure
Mesure
Mesure
Mesure
Mesure
Mesure
Mesure
goroutine grafana

La question de la découvrabilité est centrale !!!

LIVE
DEMO





Test d’efficacité : une instance de Bubble sur une machine

```
"minPollGoroutines":1,
"maxPoolGoroutines":15001,
"nbPasParCycle":75,
"dureeParPasSec":600,
"nbTriDansBubble":3,
"dureeWaitDansBubble":50
```

CONCLUSION

- Le traitement des mesures est une chose …
- Nous n’avons travaillé que sur la détectabilité des problèmes
- Maintenant, il faut les analyser pour les résoudre
- C’est le job des traces
  - Comme pour les mesures, elles doivent être adaptées aux applications cloud native
  - https://www.cncf.io/projects/ CLOUD NATIVE COMPUTING FOUNDATION

OpenTracing
Distributed Tracing API

Jaeger
Distributed Tracing

distributed transaction monitoring

performance and latency optimization

root cause analysis

service dependency analysis

distributed context propagation

https://github.com/jaegertracing/jaeger/tree/master/examples/hotrod

QUELQUES LIENS …

- https://github.com/marcdivet/Bubble
- https://prometheus.io/docs/introduction/overview/
- https://grafana.com/
- https://coreos.com/blog/the-prometheus-operator.html
  - https://coreos.com/operators/prometheus/docs/latest/user-guides/getting-started.html
  - https://coreos.com/operators/prometheus/docs/latest/
- https://github.com/prometheus/node_exporter
- https://github.com/kubernetes/kube-state-metrics/
- https://kubernetes.io/docs/getting-started-guides/minikube/
- https://docs.docker.com/engine/installation/
- http://blog.cloud66.com/how-to-create-the-smallest-possible-docker-image-for-your-golang-application/
  - https://www.habitus.io

C’EST FINI … MAINTENANT :

CONSEIL TECHNOLOGIQUE

TRANSFORMATION DIGITALE

DATA MANAGEMENT

MODERNISATION DES SI

SERVICE POUR LES APPLICATIONS

SERVICE D’INFRASTRUCTURES

Sodifrance

NETAPSYS

IT transformation to digital