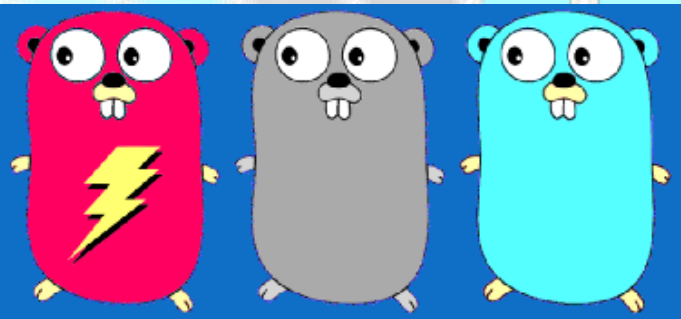
UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

université PARIS-SACLAY

Génie Logiciel pour le Calcul Scientifique



#4

13/12/2024

jean-michel.batto@cea.fr

cea

https://gogs.eldarsoft.com/M2_IHPS

Création des nœuds, connexion

INTERACTIF

- D'abord créer un swarm

```
docker swarm init
```

- Puis y attacher un réseau

```
docker network create --driver=overlay --attachable yml_mpinet
```

- Dans le répertoire du cours CM4 (ou CM3)

```
docker-compose up --scale mpihead=1 --scale mpinode=3 --scale grafana=1 -d
```

➔ utilise le fichier docker-compose.yml

➔ L'image docker n'est pas la même (jmbatto/**m2chps-mpi41-xmp**)

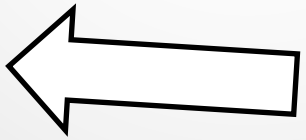
Se connecter à un shell docker (utiliser le bash) : (root car supervisor), ssh localhost:2022 → mpiuser, utiliser la clé ppk

Que contient le docker-compose ?

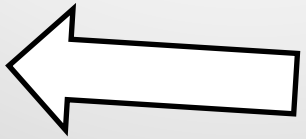
```
grafana:
    container_name: influxdb_local
    image: philhawthorne/docker-influxdb-grafana:latest
```

image disponible sur hub.docker.com

```
mpihead:
    image: jmbatto/m2chps-mpi41-xmp:latest
    shm_size: '512m'
        ports:
            - "2022:22"
```



on alloue 512mbyte pour l'espace IPC (communication inter processus, nécessaire pour MPI)

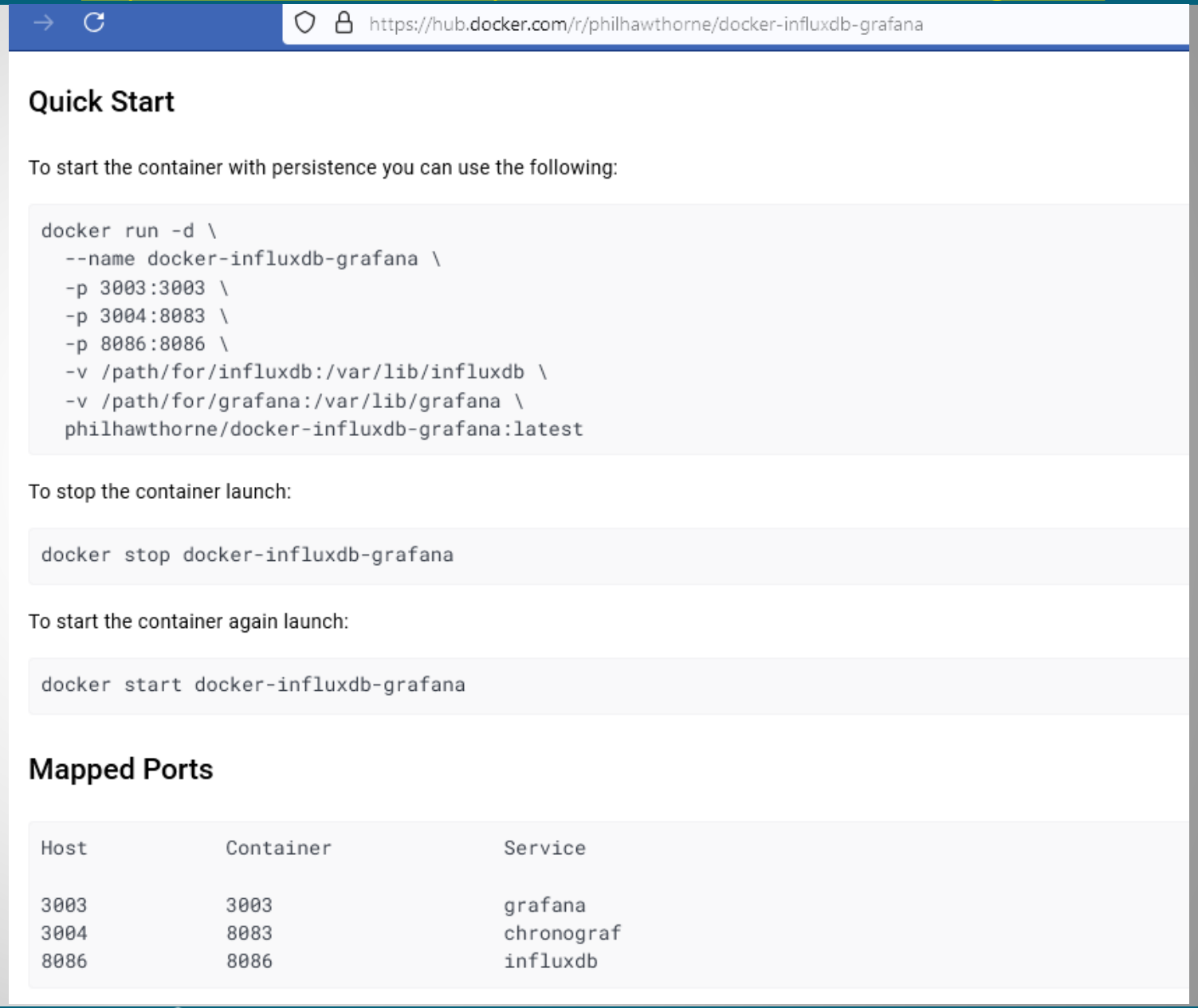


mapping du port SSH vers le port 2022

Que contient l'image ?

https://hub.docker.com/r/philhawthorne/docker-influxdb-grafana

- Doc depuis le site hub.docker.com
- Image qui contient
  - grafana (:3003)
  - chronograf (:3004)
  - Influxdb (:8086)



https://hub.docker.com/r/philhawthorne/docker-influxdb-grafana

Quick Start

To start the container with persistence you can use the following:

```
docker run -d \
  --name docker-influxdb-grafana \
  -p 3003:3003 \
  -p 3004:8083 \
  -p 8086:8086 \
  -v /path/for/influxdb:/var/lib/influxdb \
  -v /path/for/grafana:/var/lib/grafana \
  philhawthorne/docker-influxdb-grafana:latest
```

To stop the container launch:

```
docker stop docker-influxdb-grafana
```

To start the container again launch:

```
docker start docker-influxdb-grafana
```

Mapped Ports

Host	Container	Service
3003	3003	grafana
3004	8083	chronograf
8086	8086	influxdb

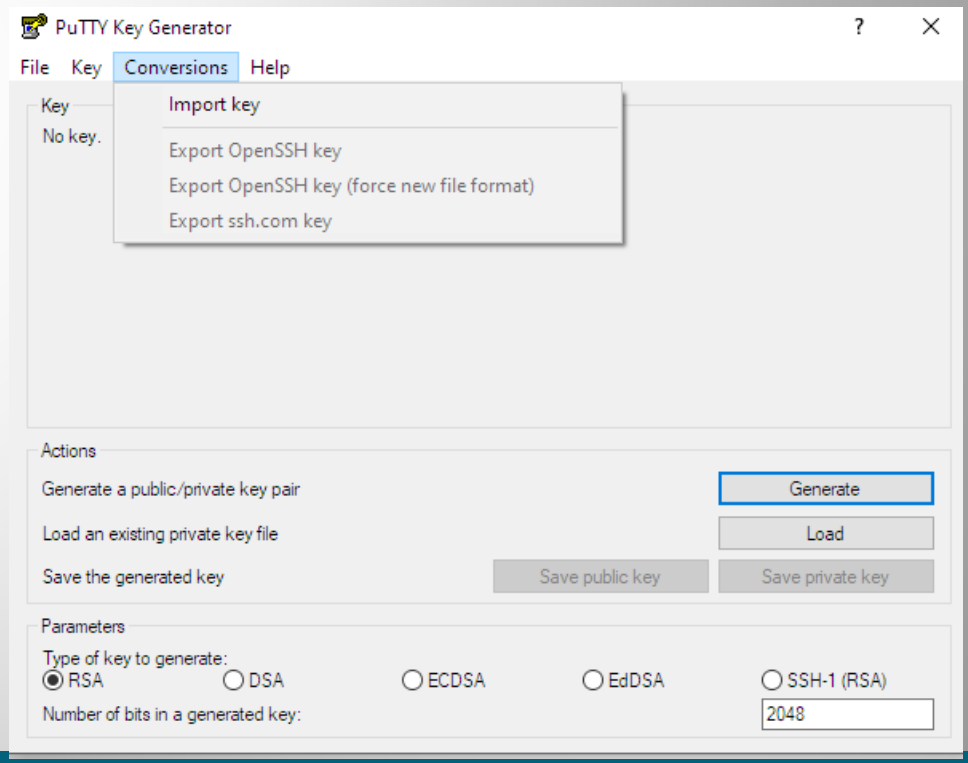
Création des nœuds, connexion

INTERACTIF

Se connecter à un shell docker (utiliser le bash) : (root car supervisor), ssh localhost:2022 → mpiuser, utiliser la clé ppk

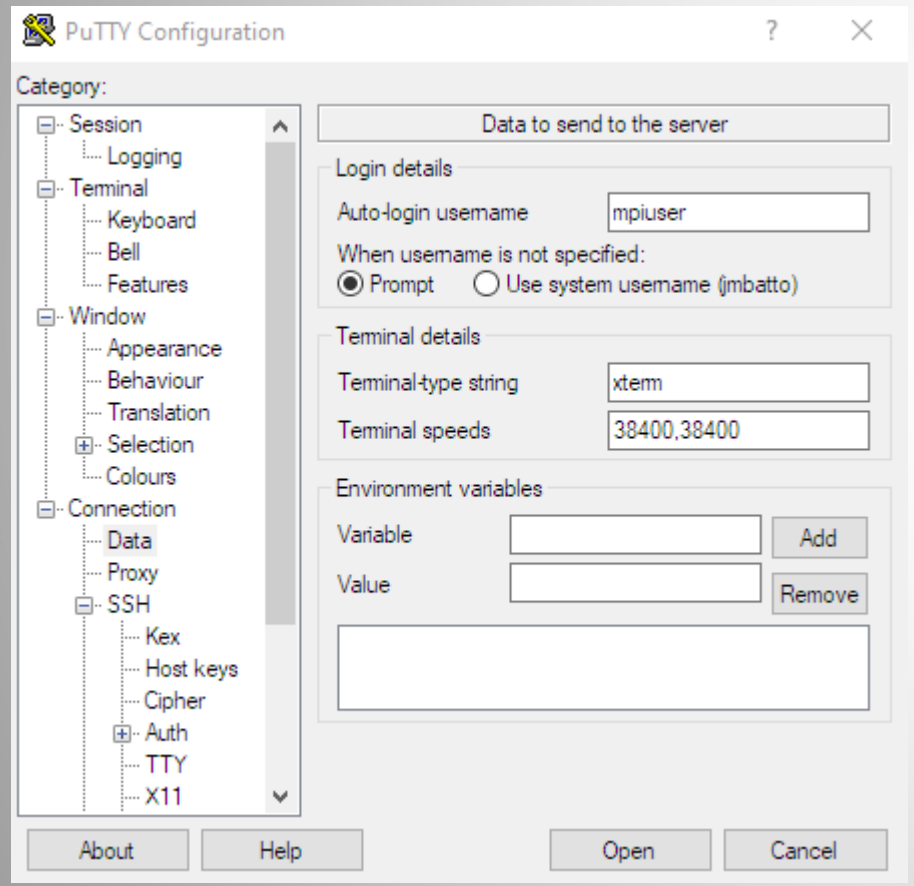
Si utilisation de Putty sous Windows

Pb : la clé fournie n'est pas au format ppk utilisé par Putty ➔ conversion de la clé au format PPK avec PuTTYgen



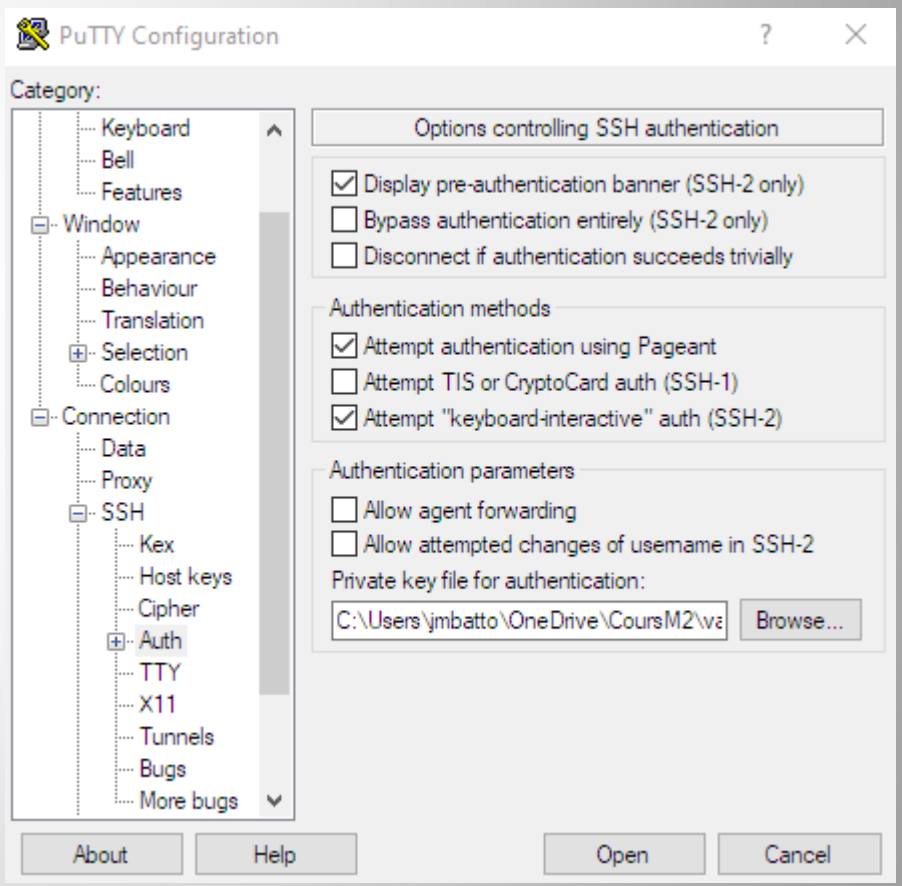
INTERACTIF

Connexion avec PuTTY, 2 paramétrages à effectuer



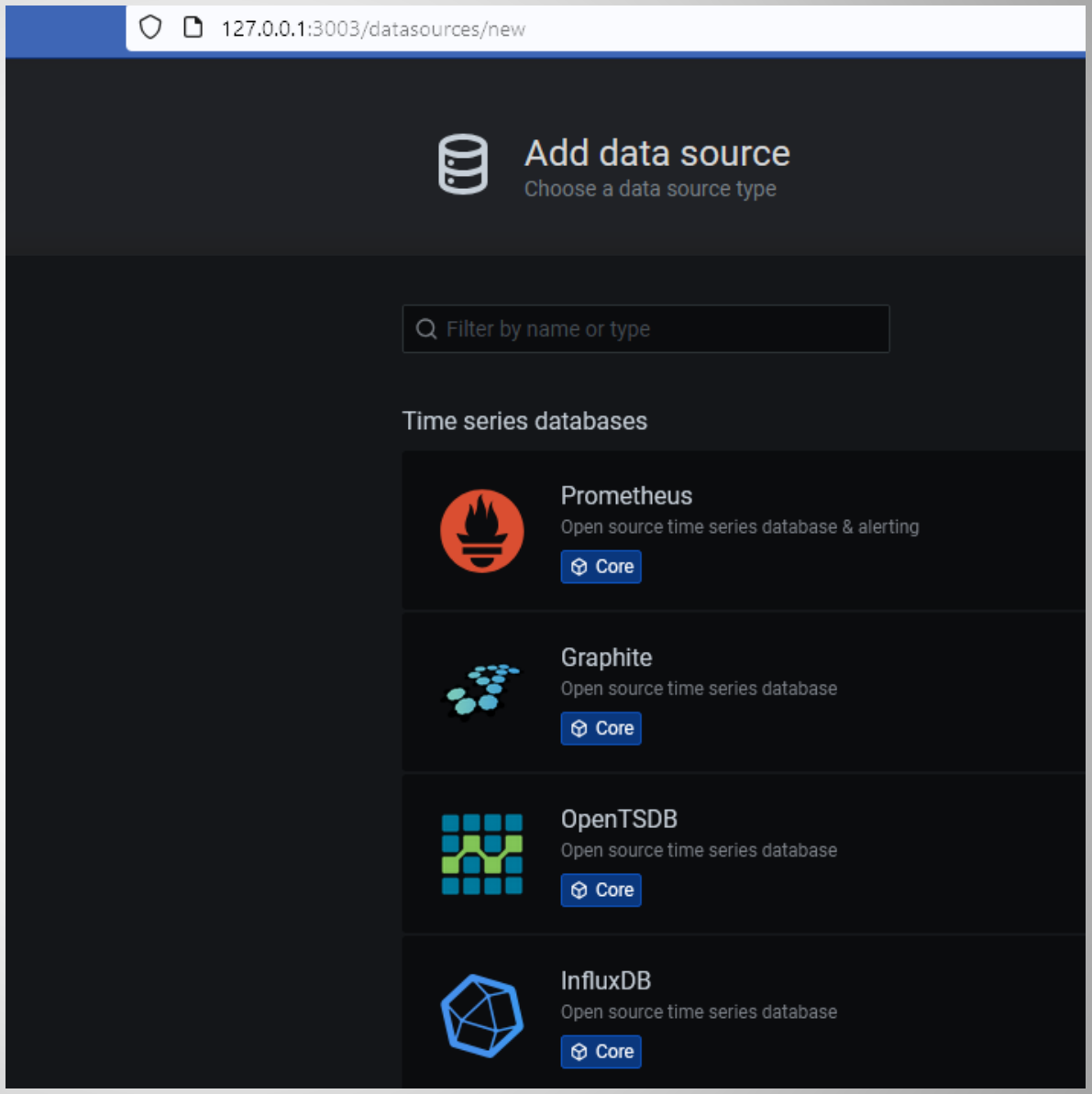
Le nom de login

La private key



- http://127.0.0.1:3003
- Login root/root
- DB : browser mode
- Target : telegraf

- ➔on ne va pas utiliser grafana



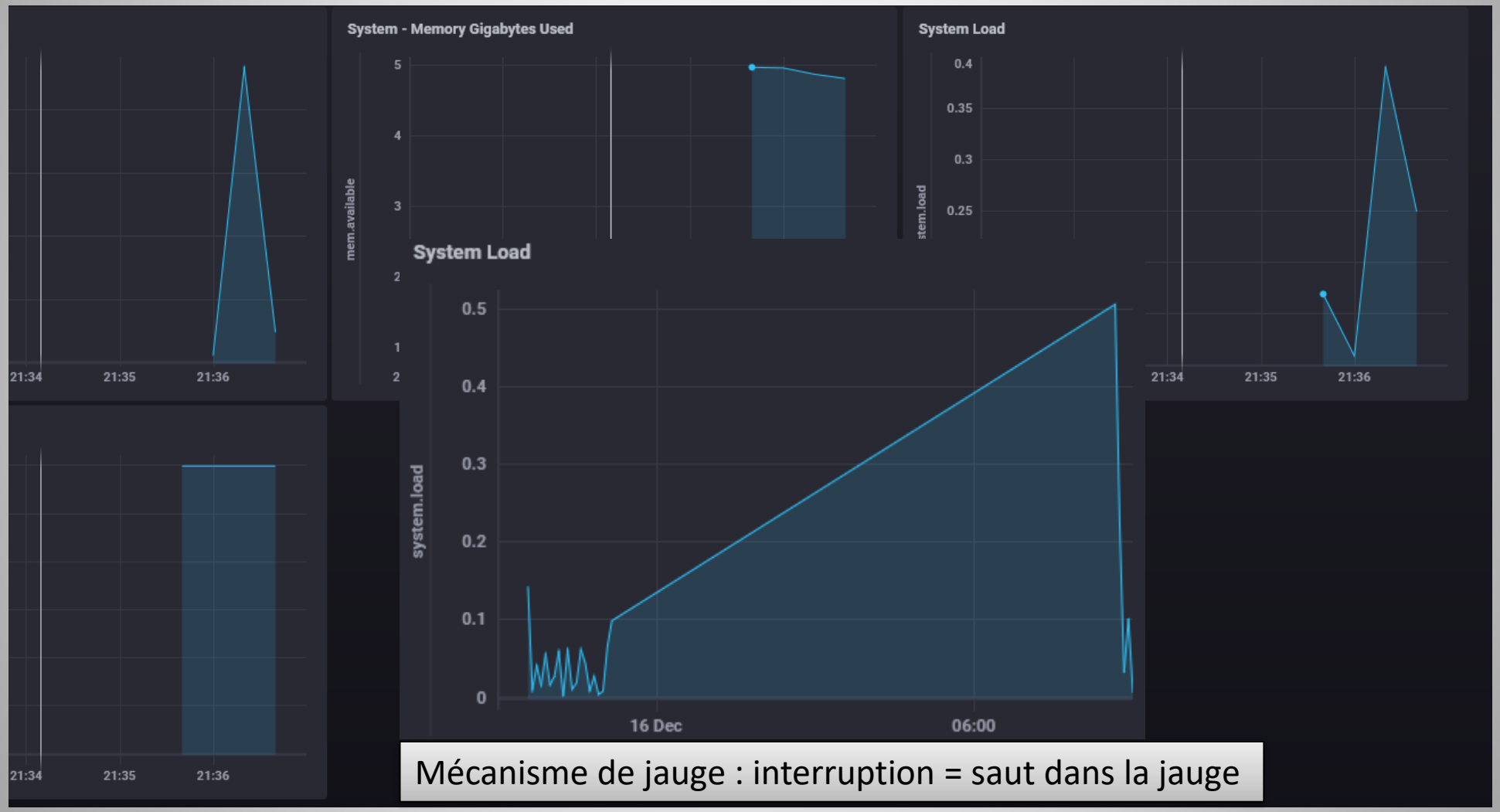
Métrologie chronograf

- http://127.0.0.1:3004
- On va travailler avec chronograf

Vision de la télémétrie technique

INTERACTIF

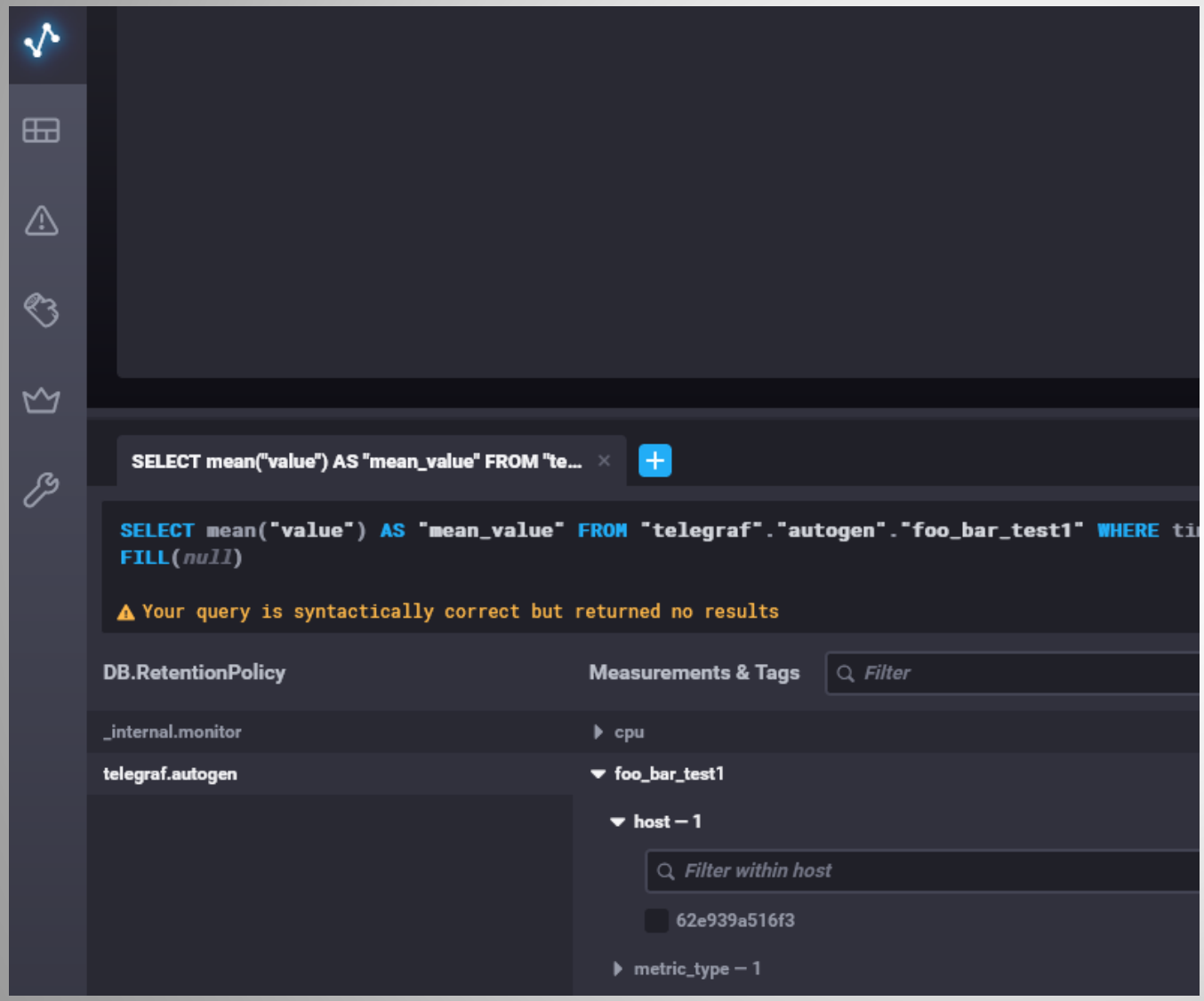
Chaque nœud a son service telegraf – qui peut suivre des applications et des services



Mécanisme de jauge : interruption = saut dans la jauge

Depuis mpihead-1

- Dans un shell faire la cmd `echo "foo.bar.test1:+1|g" | nc -u localhost 8125`



apparait foo_bar_test1 dans chronograf

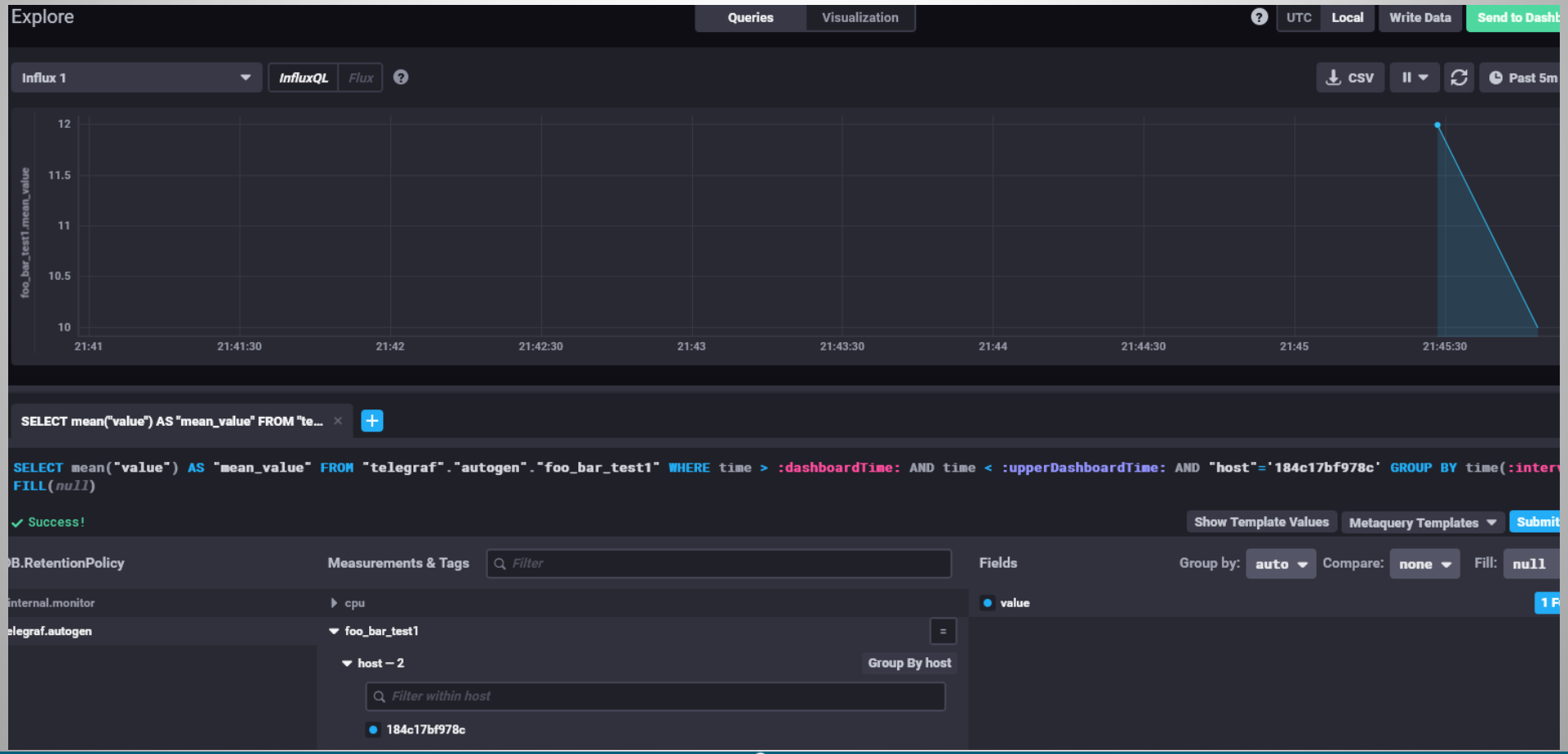
Télémétrie applicative

INTERACTIF

- statsd (2011) est un service d'audit répandu qui écoute sur un port (socket)
- Écriture avec netcat sur le port 8125 → redirection vers influxdb

```
echo "foo.bar.test1:+1|g" | nc -u localhost 8125
```

Telegraf ajoute une encapsulation https et redirige localhost vers un collecteur



Paramétrage telegraf

INTERACTIF

- ➔fichier telegraf.conf
- Le paramétrage est « anonyme » - on ne précise pas le nom du nœud
- Le service telegraf peut encrypter la communication (confidentialité)

Question : comment faire un netcat applicatif?

INTERACTIF

- 2 pistes (git clone) :
- https://github.com/guzlewski/netcat.git

- https://github.com/romanbsd/statsd-c-client.git
- (attention à LD_LIBRARY_PATH)

TD individuel pour le 22 décembre 2024

INTERACTIF

- EB : faire la télémétrie applicative de 2 nœuds MPI (osu_bibw.c)
- SFD : voici un exemple de fonction en langage Go, réalisé une fonction en C et utilisez là pour mesurer la durée de chaque benchmark.

```
//transaction : le nom de la portion de code instrumentée
//since une durée
func Chrono(since time.Duration, transaction string) {
        var client *statsd.Client
        client = statsd.NewClient("localhost:8125", statsd.MaxPacketSize(1400),
   statsd.MetricPrefix(ChronoGeneralTag+"."))← le point est supprimé
        client.PrecisionTiming("requestTime",
                since,
                statsd.StringTag("transaction", transaction))
                        client.Close()
   }
}
```

TD individuel pour le 22 décembre 2024 minuit

- A me rendre :
- Mettre dans la Forge Gogs un rapport du TD2 (**1 page** PDF – présente un benchmark dans un tableau avec une introduction qui donne le contexte, avec le nom de l'étudiant et la date)
- Avec la/les captures « chronograf » commentées
- +code source commenté (dedans : date, nom étudiant)

Idée de la séquence

- 1/ tester en mode multi-nœud MPI un code XMP
  - https://omni-compiler.org/

XcalableMP, XMP for short, is a directive-based language extension which allows users to develop parallel programs for distributed memory systems easily and to tune the performance by having minimal and simple notations.

Support typical parallelization under "global-view model"

XMP enables parallelizing the original sequential code using minimal modification with simple directives.

Support coarray to use one-sided communication easily under "local-view model"

Programmer can use coarray syntax in both XMP/Fortran and XMP/C. In particular, XMP/Fortran is designed to be compatible with Coarray Fortran.

Combination of MPI and OpenMP

In order to call an MPI program from an XMP program, XMP provides the MPI programming interface. Moreover, OpenMP directives can be combined into XMP as a hybrid programming.

Idée de la séquence

- 2/test d'un code XMP, lancement du code
- 3/adaptation du code pour afficher le nom du noeud

```
mpiuser@62e939a516f3:~/YMLEnvironment/test-spawn-xmp$ cd /usr/local/var/mpishare/
mpiuser@62e939a516f3:/usr/local/var/mpishare$ mpirun --mca orte_base_help_aggregate 0 --mca btl_tcp_if_inclu
de 10.0.1.0/24 -n 4 -host 10.0.1.7,10.0.1.4,10.0.1.5,10.0.1.6 worker_program
rank = 0 ; Processeur 0 - Nom : 62e939a516f3 (Longueur du nom : 12)

(0, 0, 0) Processeur 2 - Nom : 21cdccdc88b1 (Longueur du nom : 12)
rank = 2 ;
(2, 2, 0)
(2, 2, 1)
(2, 3, 0)
(2, 3, 1) rank = 3 ; Processeur 1 - Nom : f387910ddb9b (Longueur du nom : 12)
Processeur 3 - Nom : 3fca871ee6fd (Longueur du nom : 12)

(3, 2, 2)
(3, 2, 3)
(3, 3, 2)
(3, 3, 3)
(0, 0, 1)
(0, 1, 0)
(0, 1, 1) rank = 1 ;
(1, 0, 2)
(1, 0, 3)
(1, 1, 2)
(1, 1, 3) mpiuser@62e939a516f3:/usr/local/var/mpishare$
```

Test du code XMP

INTERACTIF

- Dans /YMLEnvironment/test-spawn-xmp → faire un make
- sudo curl --unix-socket /var/run/docker.sock http://localhost/containers/json | jq -r 'map(.NetworkSettings[]."yml_mpinet"."IPAMConfig"."IPv4Address") []`
- →pb dans la liste obtenue, il y a l'ip de l'image influxdb → faire un ifconfig –a sur l'image pour récuper l'ip

En rouge ce qui doit être adapté

```
mpirun --mca orte_base_help_aggregate 0 --mca btl_tcp_if_include
  10.0.1.0/24 -n 4 -host 10.0.1.7,10.0.1.4,10.0.1.5,10.0.1.6
  worker_program
```

Init.c ➔ explication

```
#include <stdio.h>
#include "Matrix.xmptype.h"
#pragma xmp nodes p(2,2)
#pragma xmp template t(0:3,0:3)
#pragma xmp distribute t(block,block) onto p
XMP_Matrix A[4][4];
#pragma xmp align A[i][j] with t(j,i)
XMP_Matrix B[4][4];
#pragma xmp align B[i][j] with t(j,i)
XMP_Matrix C[4][4];
#pragma xmp align C[i][j] with t(j,i)
int main(int argc, char ** argv){
  int rank;
  MPI_Barrier(MPI_COMM_WORLD);
  MPI_Comm_rank(MPI_COMM_WORLD,&rank);
  int i,j,n;
  n=4;
  fprintf(stderr,"rank = %d ; ",rank);

  #pragma xmp loop (i,j) on t(j,i)
  for (i=0;i<n;i++){
    for(j=0;j<n;j++){
      fprintf(stderr,"\n(%d, %d, %d) ",rank,i,j);
      C[i][j] = 0;
      A[i][j] = 1;
      B[i][j] = i*n+j+1;
    }
  }
  MPI_Barrier(MPI_COMM_WORLD);
}
```

Modification du code init.c

INTERACTIF

```
// Allouer de la mémoire (dans cet exemple, pour une chaîne de caractères)
    char* processor_name = malloc(256 * sizeof(char)); // Alloue de l'espace
pour le nom du processeur
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Afficher le résultat
    printf("Processeur %d - Nom : %s (Longueur du nom : %d)\n", rank,
processor_name, name_len);

    // Libérer la mémoire allouée
    free(processor_name);
```

Le code modifié

INTERACTIF

mpiuser@62e939a516f3: ~/YMLEnvironment/test-spawn-xmp

```
#pragma xmp nodes p(2,2)
#pragma xmp template t(0:3,0:3)
#pragma xmp distribute t(block,block) onto p

XMP_Matrix A[4][4];
#pragma xmp align A[i][j] with t(j,i)

XMP_Matrix B[4][4];
#pragma xmp align B[i][j] with t(j,i)

XMP_Matrix C[4][4];
#pragma xmp align C[i][j] with t(j,i)

int main(int argc, char ** argv){
  int rank;
  MPI_Barrier(MPI_COMM_WORLD);
  MPI_Comm_rank(MPI_COMM_WORLD,&rank);
  int i,j,n;
  n=4;
  fprintf(stderr,"rank = %d ; ",rank);
// Allouer de la mémoire (dans cet exemple, pour une chaîne de caractères)
    char* processor_name = malloc(256 * sizeof(char)); // Alloue de l'espace pour le nom du processeur
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);

    // Afficher le résultat
    printf("Processeur %d - Nom : %s (Longueur du nom : %d)\n", rank, processor_name, name_len);

    // Libérer la mémoire allouée
    free(processor_name);
  #pragma xmp loop (i,j) on t(j,i)
  for (i=0;i<n;i++){
    for(j=0;j<n;j++){
      fprintf(stderr,"\n(%d, %d, %d) ",rank,i,j);
      C[i][j] = 0;
      A[i][j] = 1;
      B[i][j] = i*n+j+1;
    }
  }
  MPI_Barrier(MPI_COMM_WORLD);
}


mpiuser@62e939a516f3:~/YMLEnvironment/test-spawn-xmp$
```

Test du code XMP

INTERACTIF

- Dans /YMLEnvironment/test-spawn-xmp → faire un make
- sudo curl --unix-socket /var/run/docker.sock http://localhost/containers/json | jq -r 'map(.NetworkSettings[]."yml_mpinet"."IPAMConfig"."IPv4Address") []`
- →pb dans la liste obtenue, il y a l'ip de l'image influxdb → faire un ifconfig -a sur l'image pour récuper l'ip

En rouge ce qui doit être adapté

```
mpirun --mca orte_base_help_aggregate 0 --mca btl_tcp_if_include
  10.0.1.0/24 -n 4 -host 10.0.1.7,10.0.1.4,10.0.1.5,10.0.1.6
  worker_program
```

Les patrons / patterns

Les 23 patterns décrits :

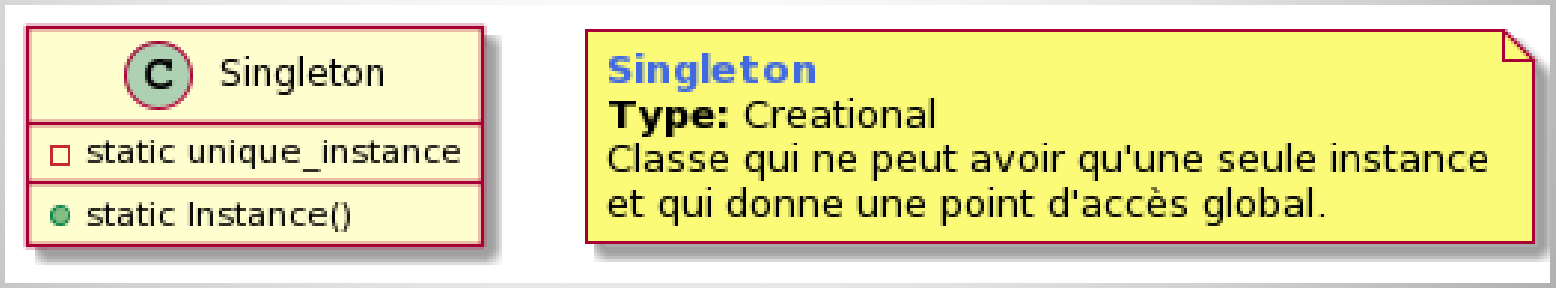
	Creational	Structural	Behavioral
Class	Factory Method	Adapter (class)	Interpreter
			Template Method
Object	Abstract Factory	Adapter (object)	13 Chain of Responsibility
	14 Builder	9 Bridge	Command
	Prototype	Composite	Iterator
	8 Singleton	10 Decorator	Mediator
		11 Facade	12 Memento
		Flyweight	Observer
		Proxy	State
			Strategy
			15 Visitor

Typologie	Nom du Design Pattern	Ce qui doit être ajuster	verbe	composition sous jacente	mélange de classes ?
Creational	Abstract Factory	famille d'objets dépendants		structure	non
	Builder	Comment créer un objet composite dont la structure du composite est indépendante		liste	non
	Factory Method	Sous classe d'un objet qui est instanciée sans connaitre la classe ancêtre (connaissance retardée)			filtrage vtab
	Prototype	Classe d'objet qui est instanciée grâce à un constructeur de copie	copie=verbe	liste	non
	Singleton	La seule instance d'une classe	copie=o=verbe		non
Structural	Adapter	accède à un objet en modifiant l'interface			non
	Bridge	Fait l'implémentation d'un objet par découplage	découplage		non
	Composite	structure et composition d'un objet vue de manière uniforme		arbre	non
	Decorator	Responsabilité d'un objet sans héritage - ajout dynamique			filtrage vtab
	Facade	Exposer une interface à un sous-système			filtrage vtab
	Flyweight	cout de stockage d'un objet, partage de l'état	état=verbe	liste	non
	Proxy	Comment un objet est accédé, son emplacement (mémoire, disque) - effet miroir		queue	non
Behavioral	Chain of Responsibility	Un objet qui peut répondre à une demande avec découpplage	découplage	queue	filtrage vtab
	Command	quand et comment une commande peut être faite - la commande devient un objet		structure	non
	Interpreter	grammaire et interprétation d'un objet			non
	Iterator	se déplacer dans une structure d'objet sans en connaitre le détail		liste	filtrage vtab
	Mediator	Comment et avec quels objets sont décrites les interactions			non
	Memento	Quelles sont les informations privées qui sont stockée à part et quand?	état=verbe	état (queue=infinie)	non
	Observer	l'effectif des objets observés et quand s'effectue la mise à jour		queue	non
	State	les états d'un objet sont des variables, avec un handler()	état=verbe	structure	filtrage vtab
	Strategy	un algorithme	extension=verbe	structure	filtrage vtab
	Template Method	les étapes/squelette d'un algorithme		structure	non
	Visitor	les opérations élémentaires sont appliquées à un objet sans modifier sa classe		liste	non

Creational / Singleton

- Classe qui ne donne qu'une seule instance
- https://godbolt.org/z/5916Mc6Ez

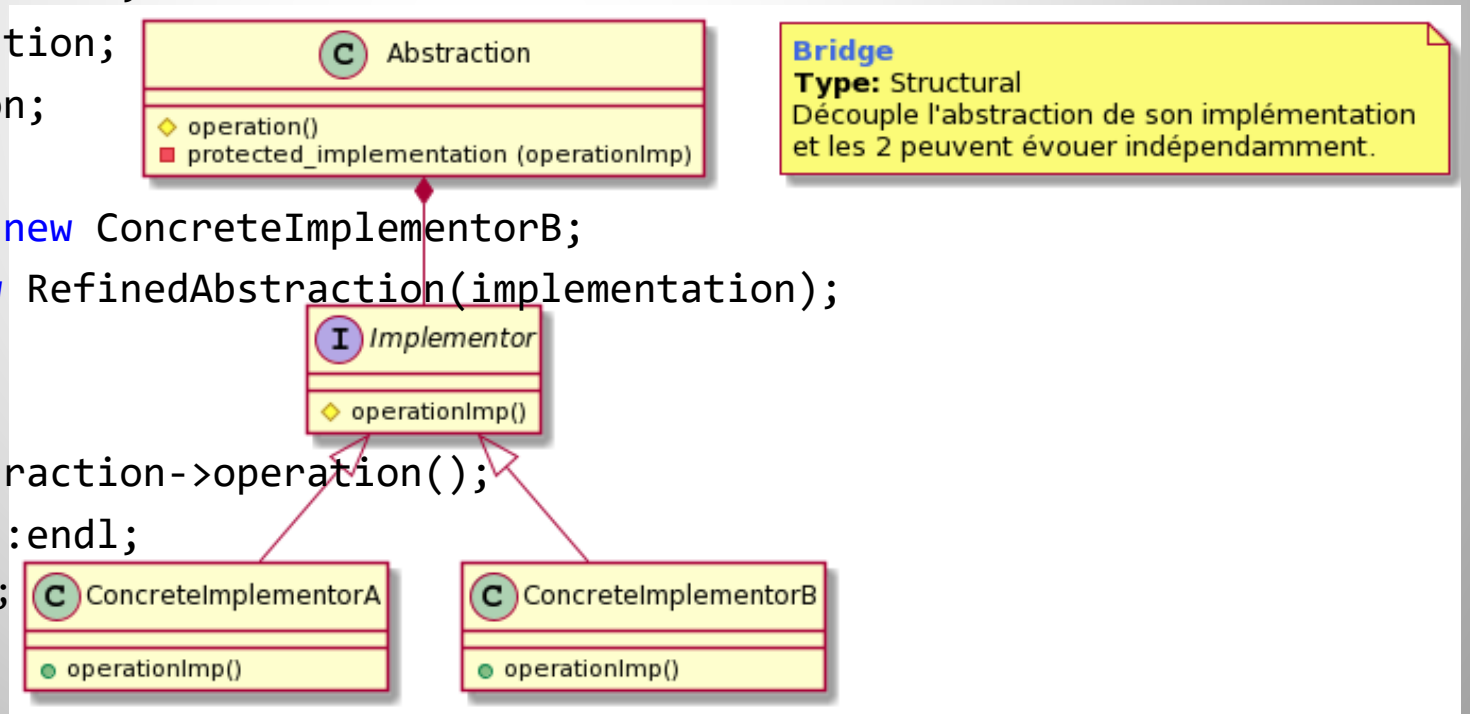
```
int main(int argc, char* argv[]) {
  Singleton *singleton = Singleton::Instance();
  singleton->checkSingleton();
  //we create a new singleton2 but...
  Singleton *singleton2 = Singleton::Instance();
  singleton2->checkSingleton();
}
```



Structural / Bridge

- Découpler le contrat et l'implémentation
- https://godbolt.org/z/EfafGzMWG

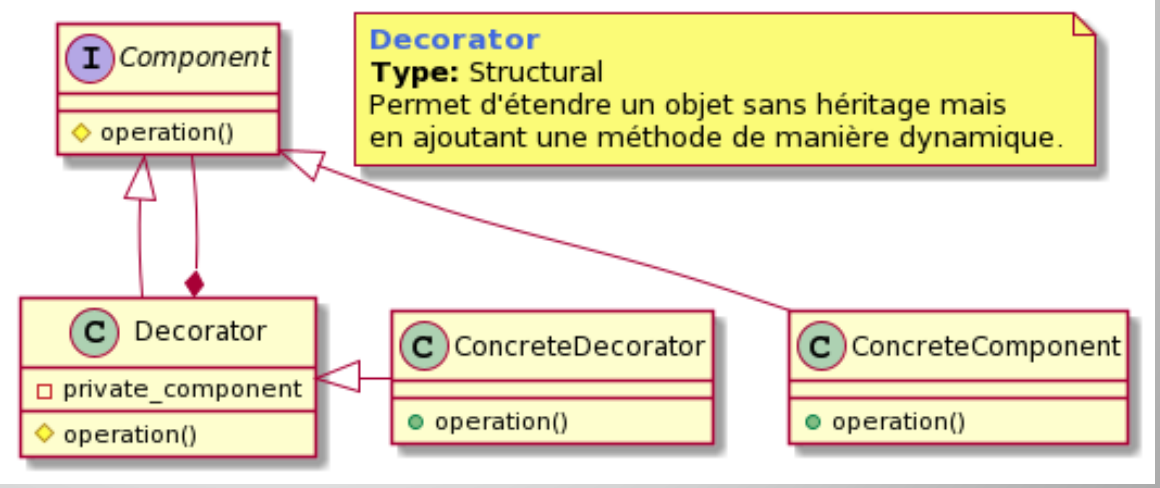
```
int main() {
  Implementor* implementation = new ConcreteImplementorA;
  Abstraction* abstraction = new Abstraction(implementation);
  std::cout << abstraction->operation();
  std::cout << std::endl;
  delete implementation;
  delete abstraction;

  implementation = new ConcreteImplementorB;
  abstraction = new RefinedAbstraction(implementation);



  std::cout << abstraction->operation();
  std::cout << std::endl;
delete implementation;
delete abstraction;
}
```



Structural / Decorator

- Ajouter des responsabilités dynamiquement
- https://godbolt.org/z/9f8j3xM6r

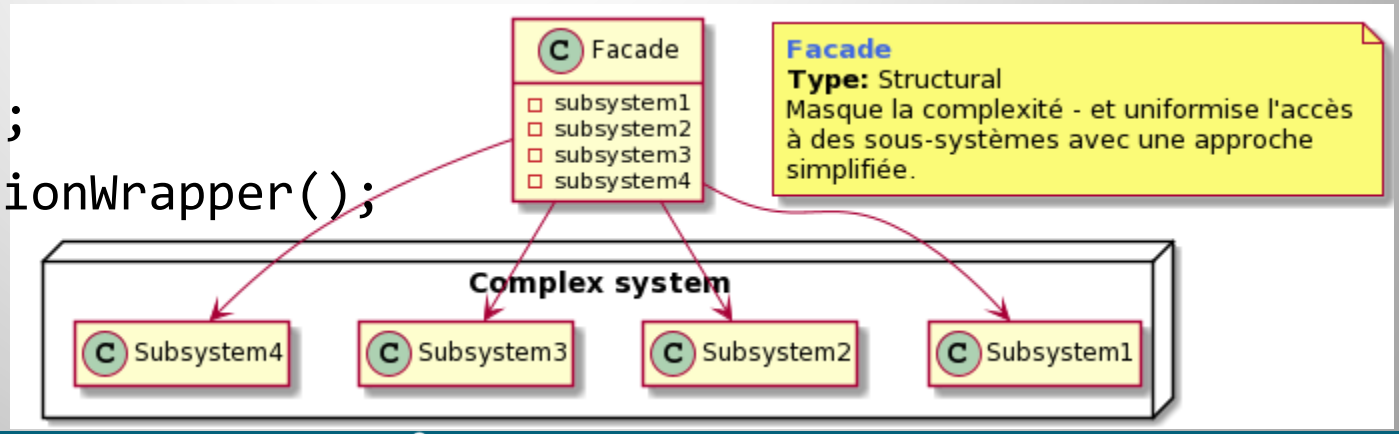
```
int main() {
  Component *cc = new ConcreteComponent();
  ConcreteDecoratorA *cda = new ConcreteDecoratorA(cc);
  ConcreteDecoratorB *cdb = new ConcreteDecoratorB(cc);

  cda->operation();
  cdb->operation();

}
```



Structural / Facade

- Interface de sous-système simplifiée
- https://godbolt.org/z/oWTWTaPc3
- `private:`

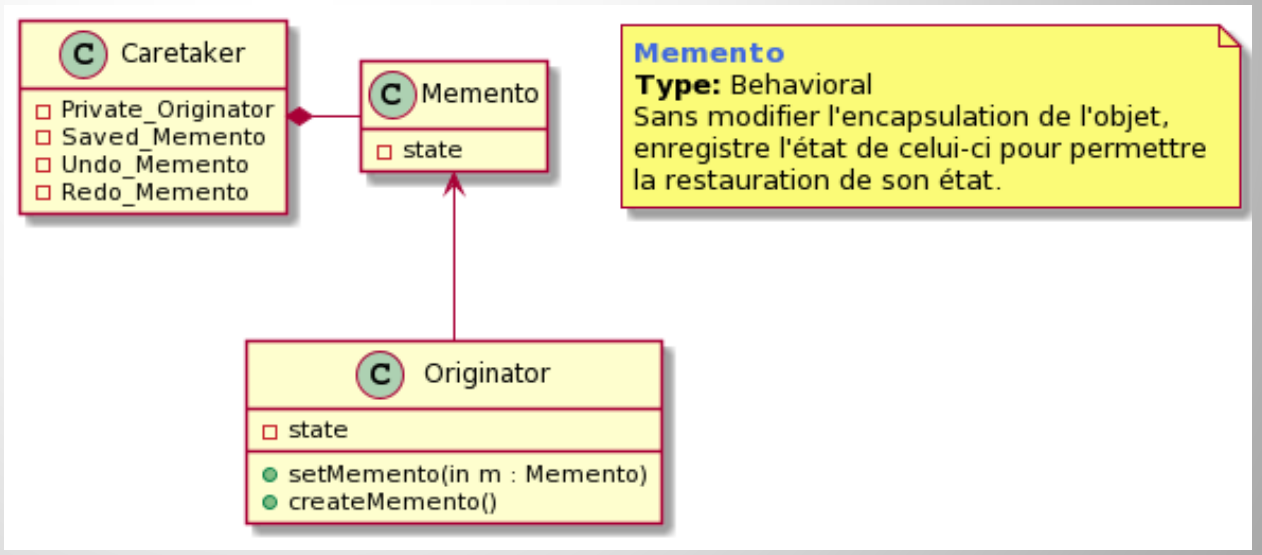
```
  SubSystem1 *subsystem1;
  SubSystem2 *subsystem2;
  SubSystem3 *subsystem3;
  SubSystem4 *subsystem4;
};
int main() {
  Facade facade;
  facade.operationWrapper();
}
```



Behavioral / Memento

- Enregistrer l'état d'un objet sans briser l'encapsulation
- https://godbolt.org/z/v7qW3TWqb

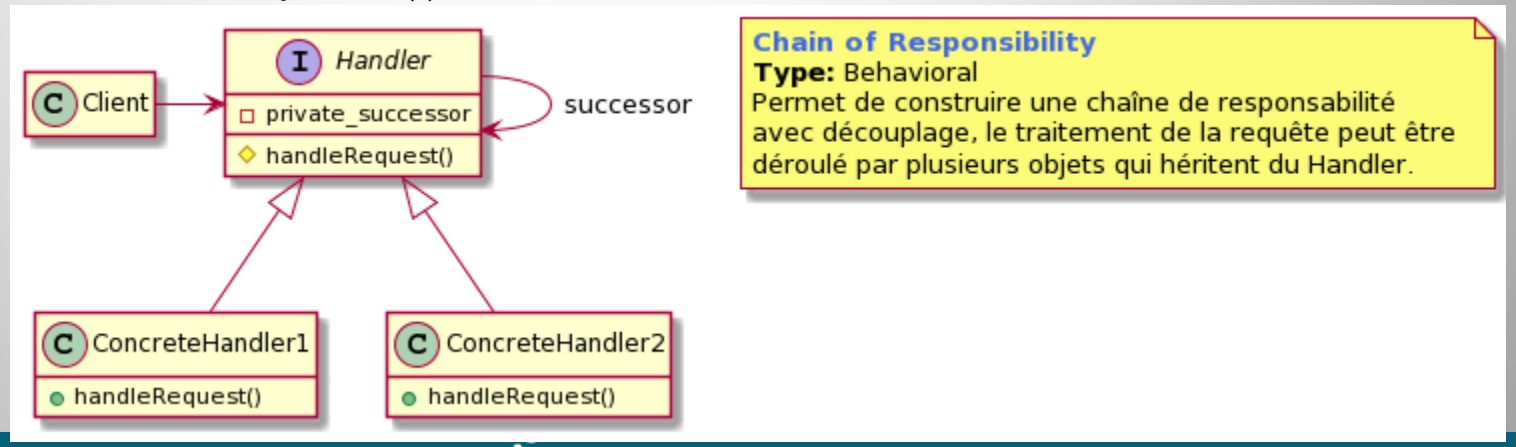
```
int main() {
    Originator *originator = new Originator();
    CareTaker *careTaker = new CareTaker( originator);

    originator->setState(0);
    careTaker->save();
    originator->setState(1);
    careTaker->save();
    originator->setState(2);
    careTaker->save();
    careTaker->printSavedStates();
    careTaker->undo();
    careTaker->printSavedStates();
    careTaker->redo();
    careTaker->printSavedStates();
    careTaker->undo();
    careTaker->undo();
    careTaker->undo();
    careTaker->printSavedStates();
}
```



Behavioral / Chain of Responsability

- Découpler le client et le fournisseur
- https://godbolt.org/z/KMzn87ThM

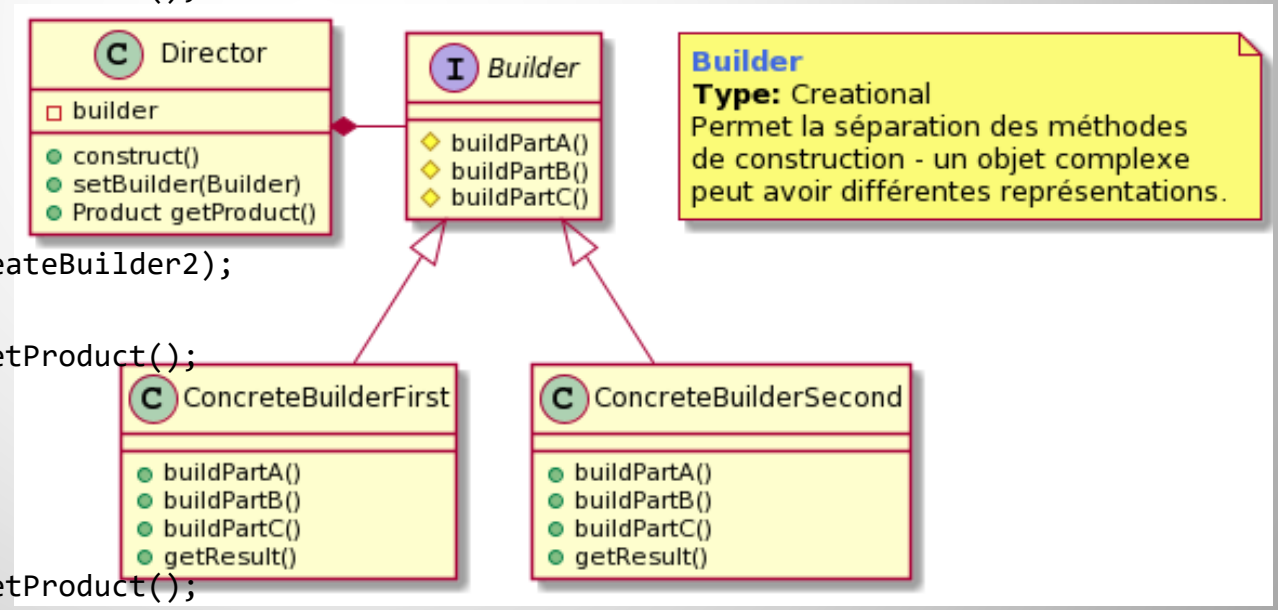
```
int main() {
    Client* client = new Client();
    client->h = new ConcreteHandler1();
    Handler* h2 = new ConcreteHandler2();
    client->h->handleRequest();
    client->h->setSuccessor(h2);
    client->h->handleRequest();
    //h2->setSuccessor(client->h);
    client->h->handleRequest();
}
```



Creational / Builder

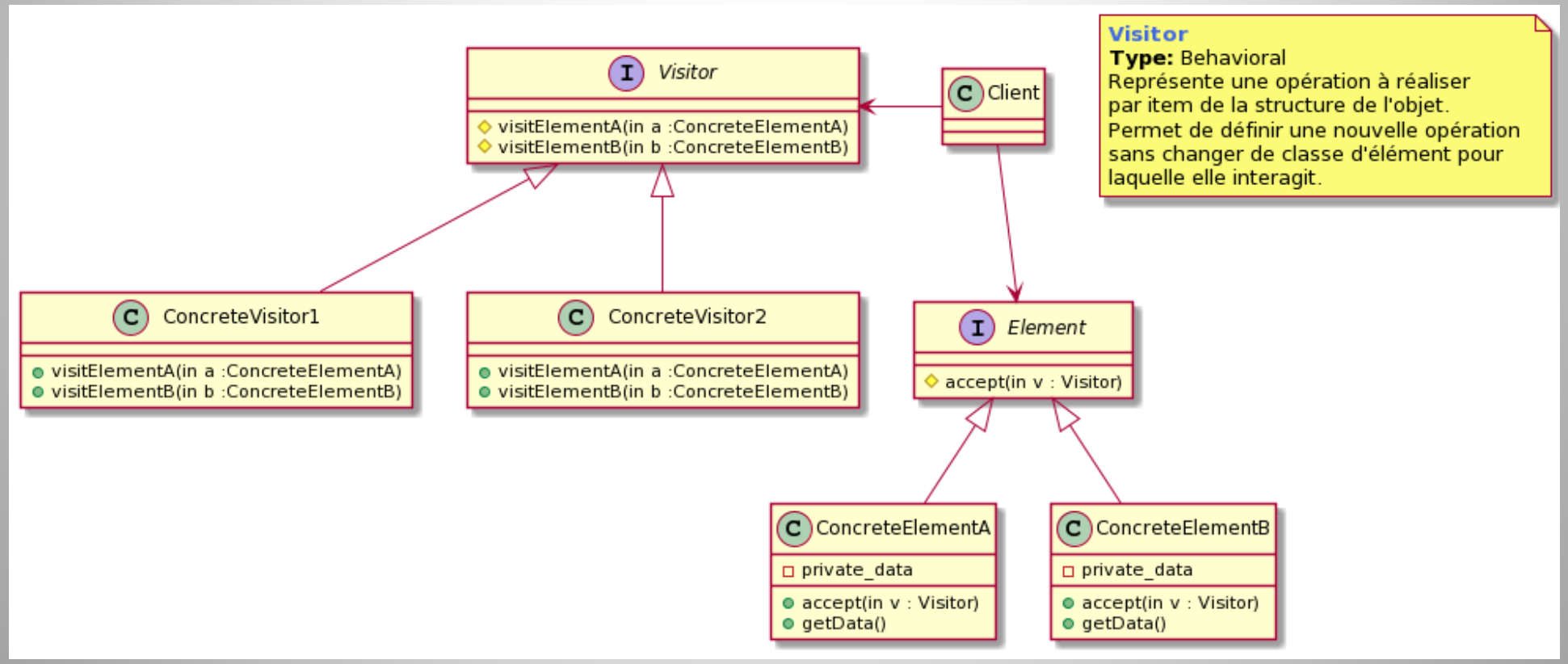
- L'algorithme de création est indépendant de la structure
- https://godbolt.org/z/zh8vjWj44

```
int main(int argc, char* argv[]) {
  Director director;
  director.setBuilder(new ConcreateBuilder1);
  director.construct();
  Product product1 = director.getProduct();
  product1.checkProduct();

  director.setBuilder(new ConcreateBuilder2);
  director.construct();
  Product product2 = director.getProduct();
  product2.checkProduct();

  director.setBuilder(0);
  director.construct();
  Product product3 = director.getProduct();
  product3.checkProduct();
}
```



Behavioral / Visitor

- les opérations élémentaires sont appliquées à un objet sans modifier sa classe
- https://godbolt.org/z/6zsxsaPoq



Behavioral / Visitor

```
int main() {
    ConcreteVisitor1 visitor1;
    ConcreteVisitor2 visitor2;
    ConcreteElementA elementA("String ElementA ConcreteElementA");
    elementA.accept(visitor1);
    elementA.accept(visitor2);
    ConcreteElementB elementB("String ElementB ConcreteElementB");
    elementB.accept(visitor1);
    elementB.accept(visitor2);
}
```

Les 23 Patterns

- Il nous reste à voir 9 patterns
- Existe-t-il le pattern absolu?

- ➔La POO oriente naturellement à concentrer les propriétés.
- ➔la vision modulaire est un choix.

Embarqué

- 2 approches :
  - Bare metal (ie microcontrôleur)
  - Linux (ou OS RT)

Gains :

- Bare metal : fiabilité, difficulté d'évolution
- Linux :
  - Versatilité (pas trop d'adhérence à l'implémentation)
  - Gain de productivité (cycle de vie)
  - Docker (ou Podman)

Edge Computing

- **Définition** : Un mode du traitement de données qui vise à effectuer les opérations au plus près de la source des données au sens physique.
- **Avantages** :
  - Evite la saturation de la bande passante / fluctuation du réseau sans fil
  - Elimine les latences de communication
  - Permet l'obtention des résultats en temps réel
- **Contraintes** :
  - Coût des unités de calculs (arbitrage entre le coût et la performance)
  - Complexité de la mise au point
  - Sécurité des données locales
  - Alimentation électrique et reprise sur erreur

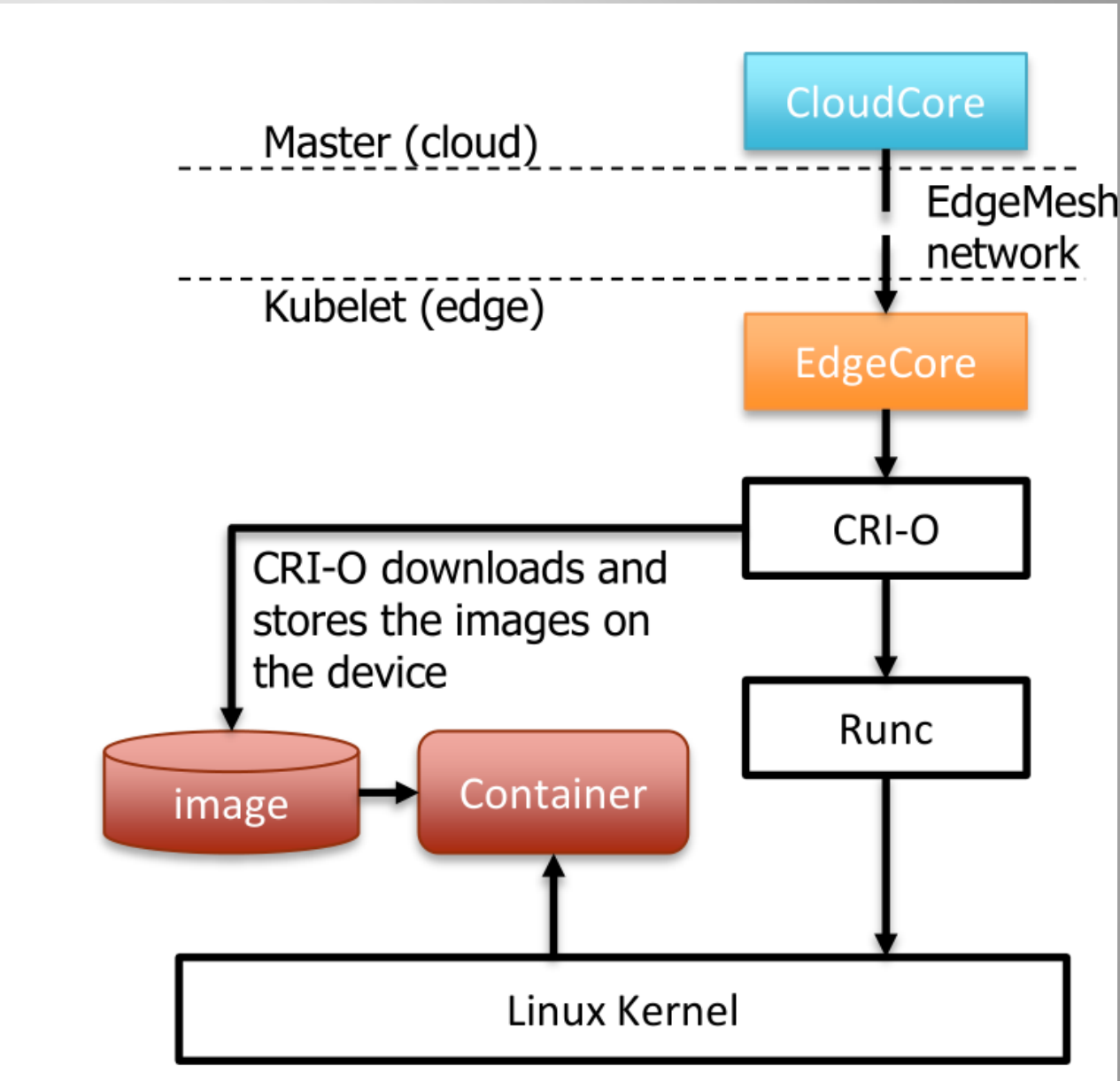
Exemple d'architecture

- Octobre 2023
- On the Containerization and Orchestration of RISC-V architectures for Edge-Cloud computing - Francesco Lumpp,Francesco Barchi,Andrea Acquaviva,Nicola Bombieri

- ESAAM '23: Proceedings of the 3rd Eclipse Security, AI, Architecture and Modelling Conference on Cloud to Edge ContinuumOctober 2023, pp 21–28 https://doi.org/10.1145/3624486.3624490
- Dans la forge RiscV_architecture_Edge.pdf
  - ➔quelle(s) est (sont) l'idée (s) « force(s) » de l'article ?

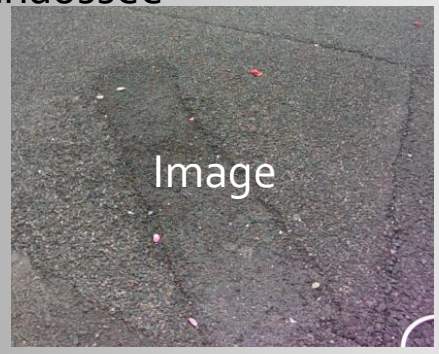
Le bénéfice du déploiement Edge sur Linux

- Container Runtime Interface for the Open Container Initiative
- ➔limites de l'approche ?

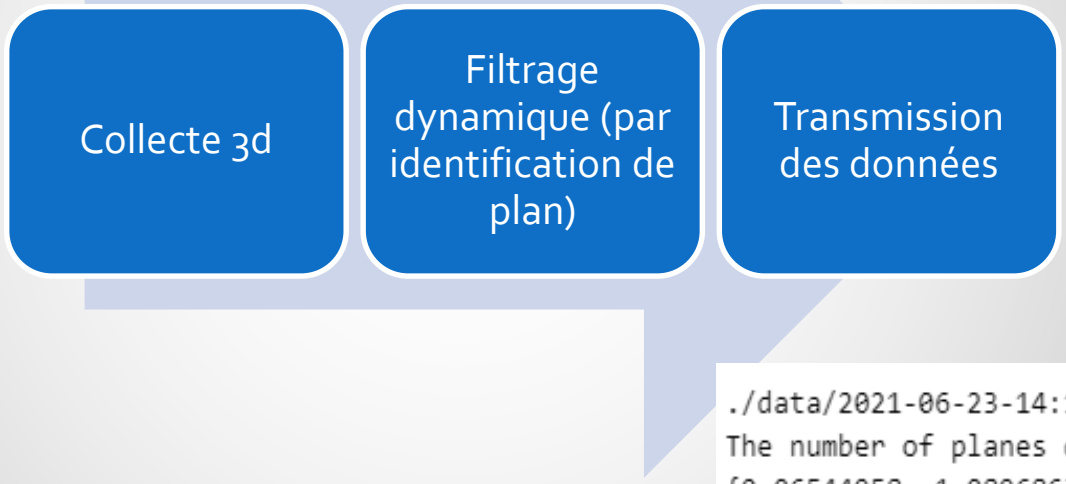


Exemple de traitement : métrologie embarquée

Objet d'intérêt : le revêtement de la chaussée



Détection du nombre de plans à la surface
Indicateur de qualité de la surface




```
./data/2021-06-23-14:10:56:934-Points.ply
The number of planes detected :  2
{0.06544958 -1.0896863 -1 -0.97271776}
{0.06585827 -1.0733582 -1 -0.9721228}
```

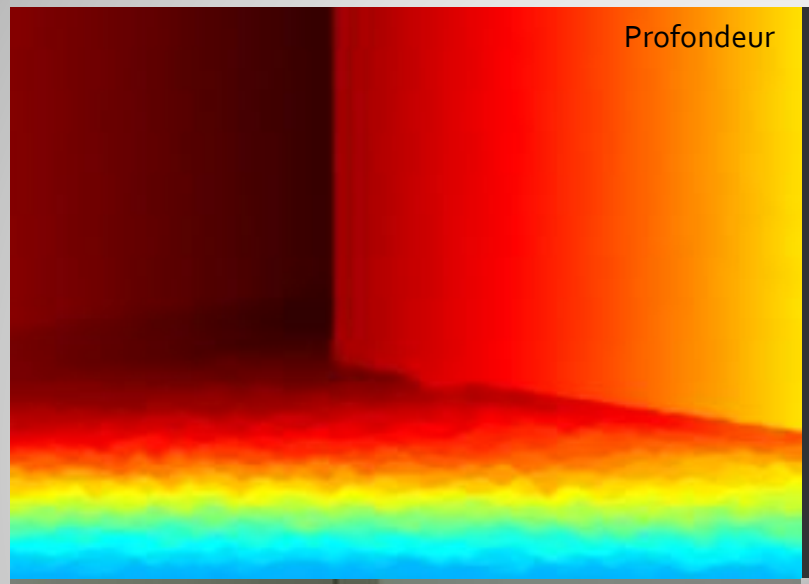
Double objectif :

- construire un indicateur (non présenté)
- optimiser l'acquisition et le traitement

- **1/ implémentation en Golang :**
  - Gain de productivité dans le codage et les tests
  - Vitesse proche de celle du C++
  - Profiler/Multithread
  - Simplicité du déploiement
- **2/ approche « low cost » :**
  - Cible Raspberry PI4B (200€HT) avec Raspbian (debian 10 buster)
  - Disque dur SSD SATA 500gb Samsung 860 Evo (50€HT)
  - Capteur Lidar Intel L515 (350€HT)
  - Alimentation et autres matériels (200€HT)

Collecte des données avec LiDAR INTEL L515

Profondeur



Nuage des points

```
ply
format binary_little_endian 1.0
comment pointcloud saved from Realsense Viewer
element vertex 305866
property float32 x
property float32 y
property float32 z
property uchar red
property uchar green
property uchar blue
element face 608632
property list uchar int vertex_indices
end_header
```

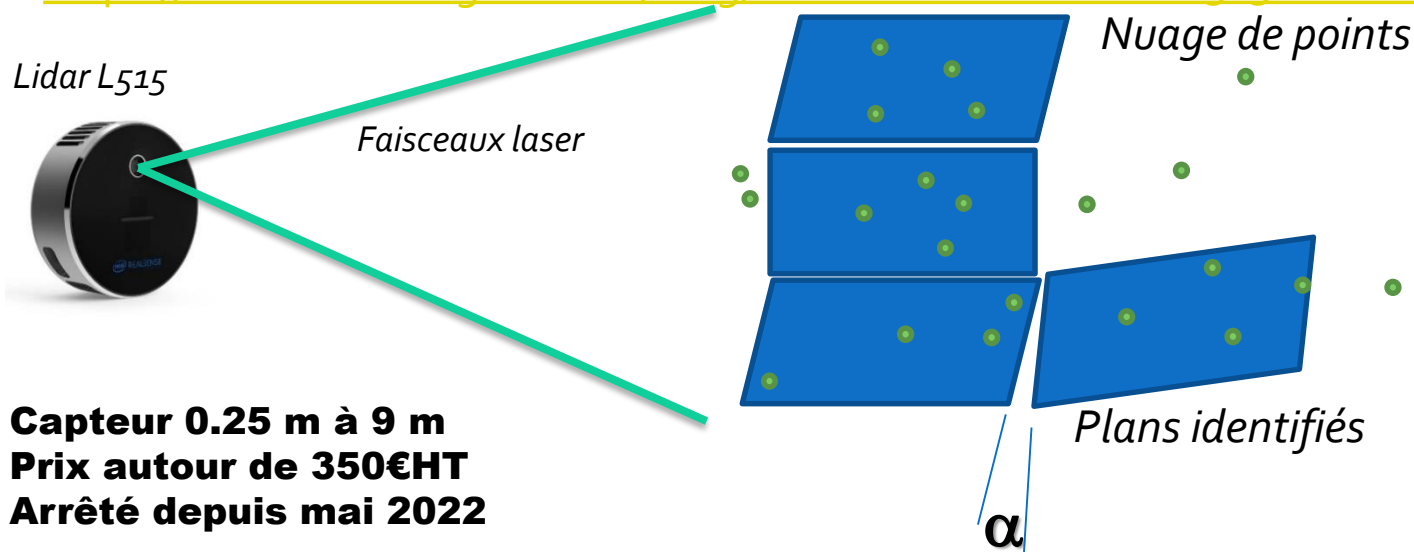
RGB

Infrarouge

La méthode de traitement

- Une méthode non topologique :
  - Capturer un nuage de points de la surface à analyser
  - Identifier les plans
  - Analyser la distribution des angles < α > entre les plans ➔ indicateur non présenté

  ➔obtenir un indicateur de la surface – une surface lisse a une faible distribution
- https://www.techinsights.com/blog/inside-intel-realsense-l515-lidar-camera



Capteur 0.25 m à 9 m
Prix autour de 350€HT
Arrêté depuis mai 2022

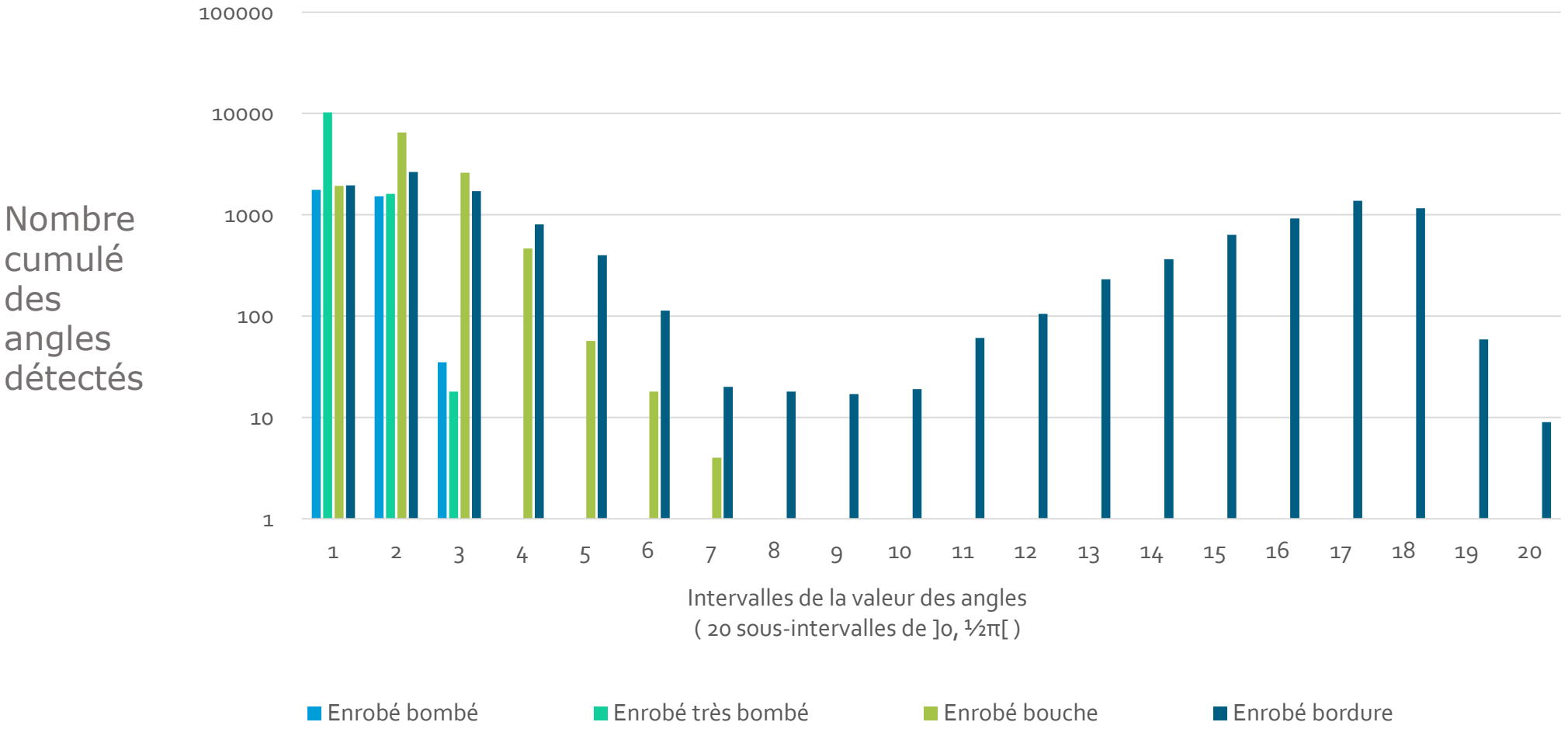
Test sur le terrain

- Jeux de données :

5 états types

Test sur le terrain

- Résultat : itération de 100 plans



- Le LiDAR fait une acquisition dans le format PLY (https://en.wikipedia.org/wiki/PLY_(file_format))
- Bibliothèque de Golang ***plyfile*** qui lit et écrit fichiers des nuages des points en format binaire

La bibliothèque originale

- Ouverture du fichier
- Lecture des binaires

- Parseur du header
- Décodage
- Gestion des mémoire

La bibliothèque revisitée

- Elimination de la présence du C
- Fonctionnalités préservées

https://github.com/cheer-Yuan/plyfile

La cible opérationnelle

Type de connecteur	Finalité de l'exemple	Type d'information	Plan de test
SAE (Service d'Aide à l'Exploitation)	Donner la taille d'un fichier traité.	Entier	Vérifier que l'écriture est effective
Data Bas débit	Vérifier la capacité à lire l'état de la collecte d'un nœud périphérique	API Rest, échange JSON	Identifier les paramètres dans le fichier JSON
Data Haut débit	Transmettre un ensemble de données acquis pas la caméra lidar L515 Intel.	Fichier	Métrologie de l'écriture, conformité du fichier

OS embarqué	Kernel	Processeur	Unité de stockage	Commande de l'applicatif	Options de compilation	Taille du fichier	Durée du traitement	Débit
Raspbian GNU/Linux 10 (buster)	5.10.17-v7l+	ARMv7 rev3 (v7l)	SSD SATA 500gb 860 Evo	./datasaving -n 100 -i 0 -f 0 -r 0 -l 5	-mcpu=cortex-a72 -mtune=cortex-a72 -mfpu=neon-fp-armv8 -g -O3	207Mb	32,988 secondes	6,2 Mb/s

Méthode numérique / RANSAC

- RANSAC = méthode de découpage d'un nuage par graine – méthode non déterministe

- RANSAC fait de l'ajustement de plan – pour déterminer le nombre de plan optimal
- 2 approches pour l'ajustement : la régression par moindre carré (inversion matrice)

$$A^T Ax = A^T b \Rightarrow x = (A^T A)^{-1} A^T b$$

- ou la SVD, par estimation de la perte sur $a(x - \bar{x}) + b(y - \bar{y}) + c(z - \bar{z}) = 0$

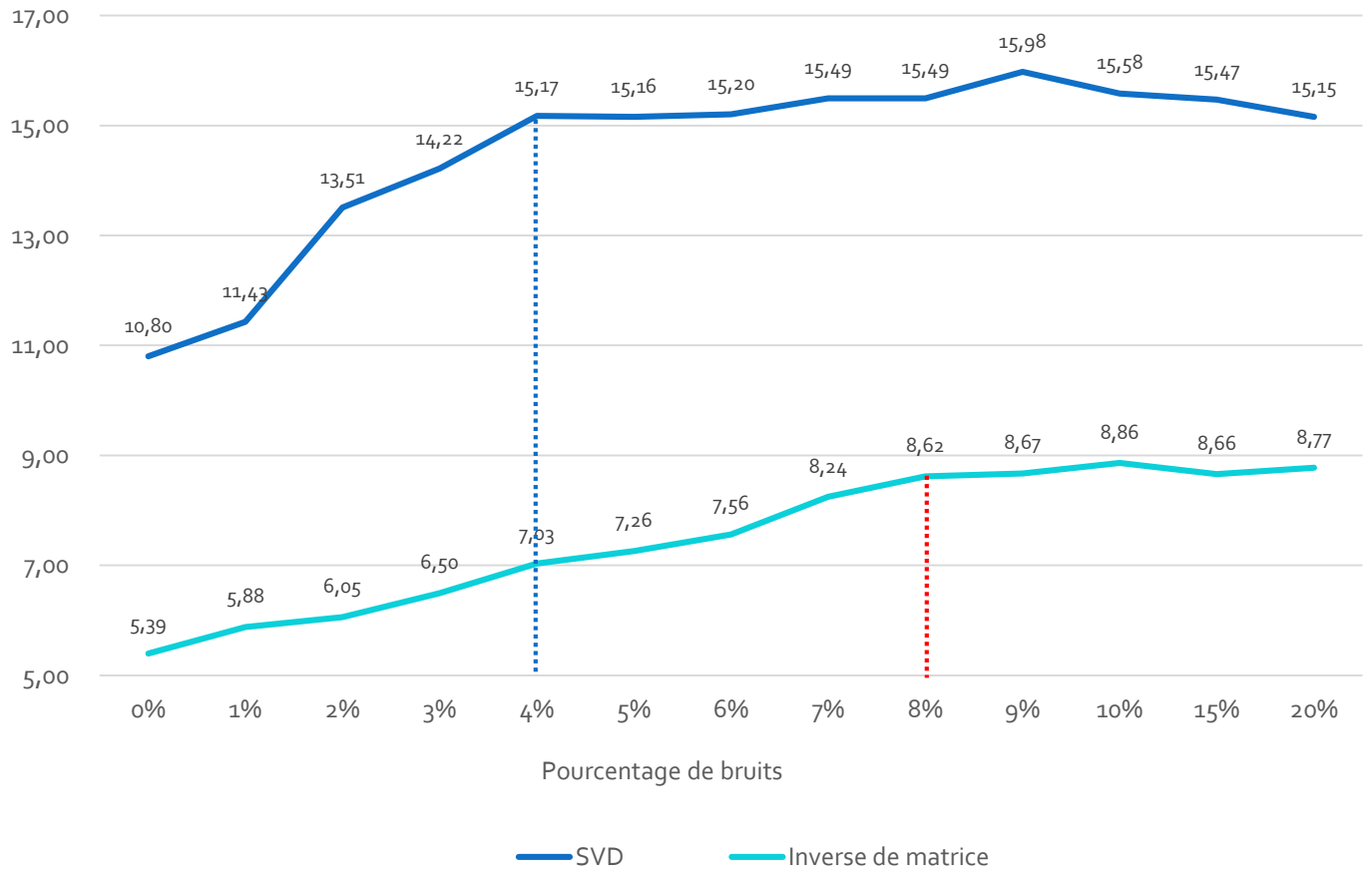
- Comparaison de robustesse entre SVD et inversion de matrice par ajout de bruit

  – Intervalle qui encadre la valeur du bruit aléatoire : **[0.05,0.1[** mètre

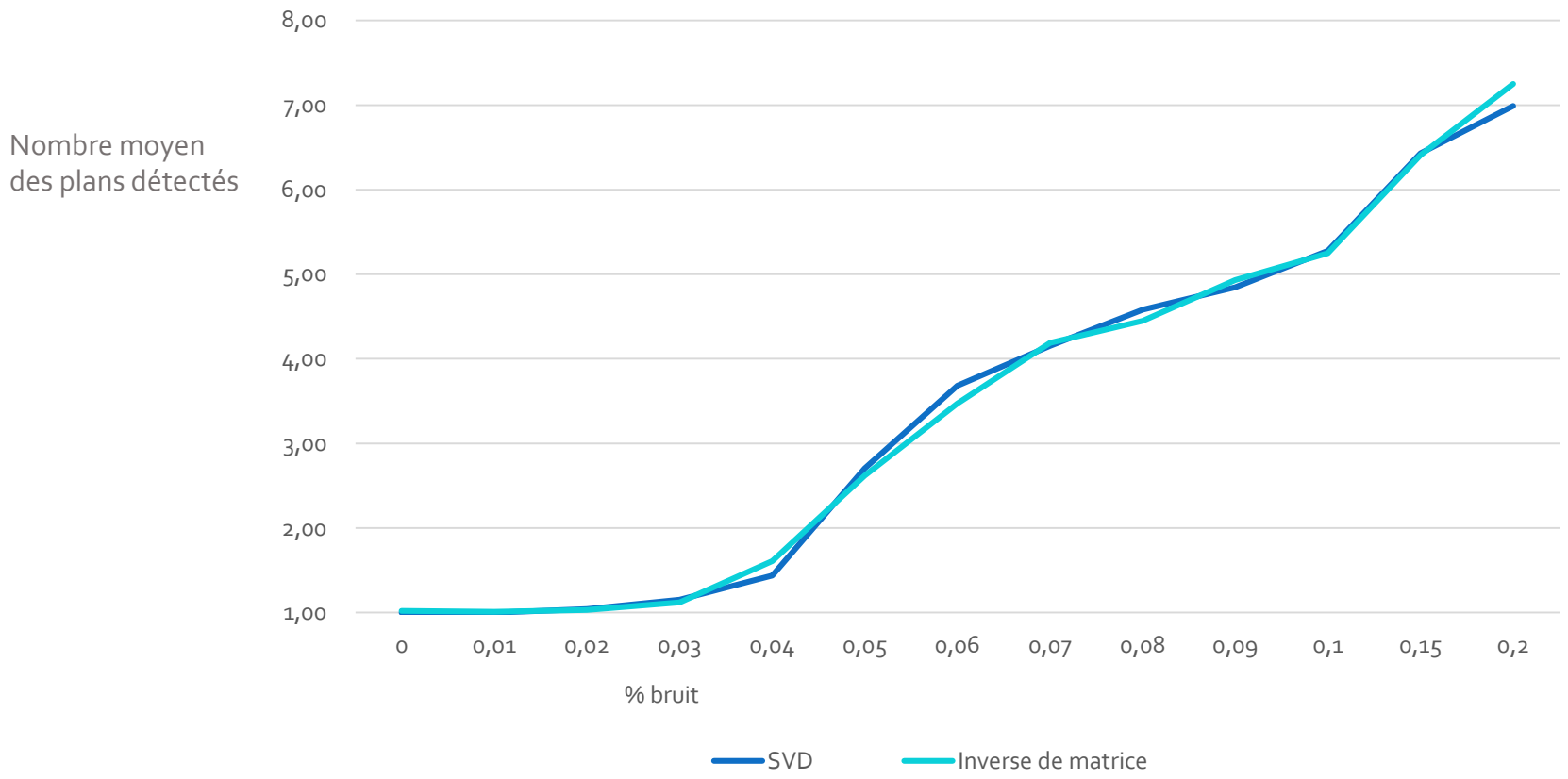
  – Pourcentage de sommets ajoutés d'un bruit : de 0% à 20%

Méthode numérique : bruit et temps de calcul

- Influence sur le temps de calcul :

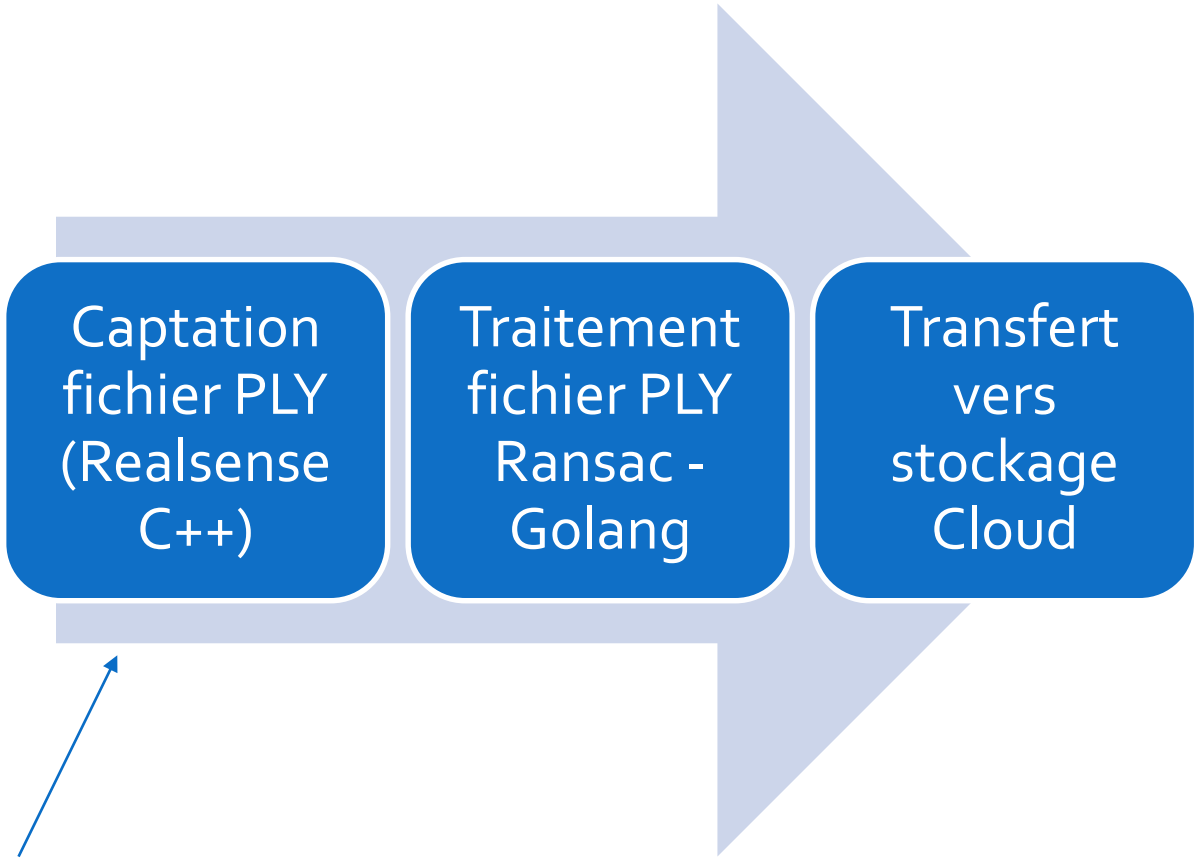


Comportement au bruit



- Les 2 méthodes ont le même comportement

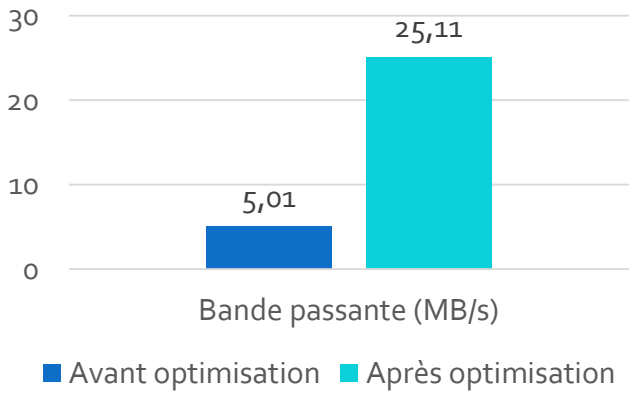
Optimisations

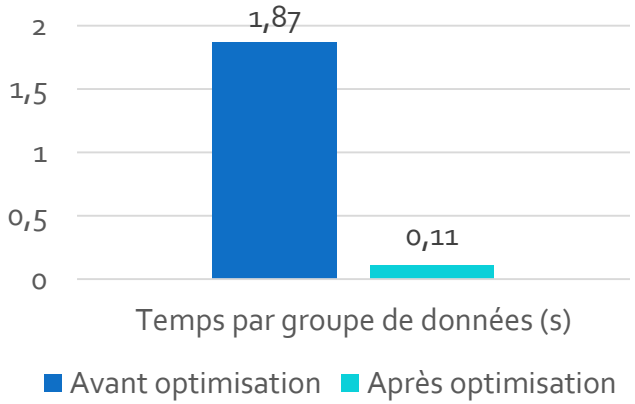


Modification de l'outil de captation pour écrire vers un PIPE
Optimisation du format de sortie (gain sur le volume écrit)
https://github.com/cheer-Yuan/librealsense

Les optimisations

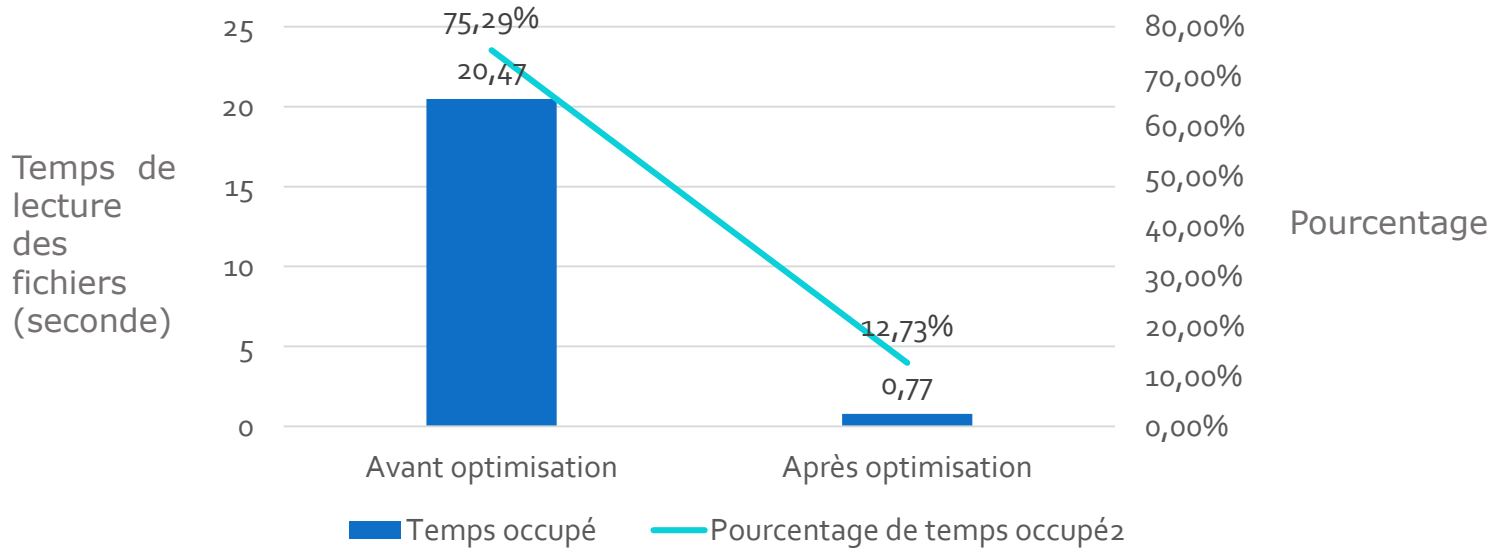
- Plateforme de test : x86, go 1.16, i5-1035G7 @1.2-3.7 GHz
- Optimisations :
  - Elimination des flux non-exigés
  - Elimination du temps d'attente
- Utiliser un SDK modifié :
  - https://github.com/cheer-Yuan/librealsense





Les optimisations

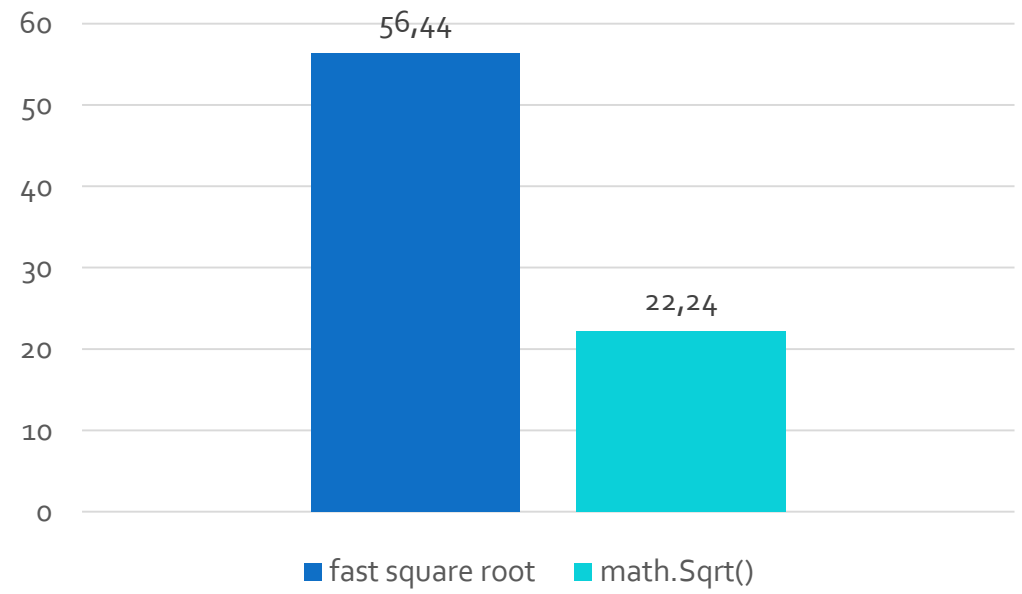
- Traitement de données :
  - Chargement des données : problème identifier par profiling



Les optimisations

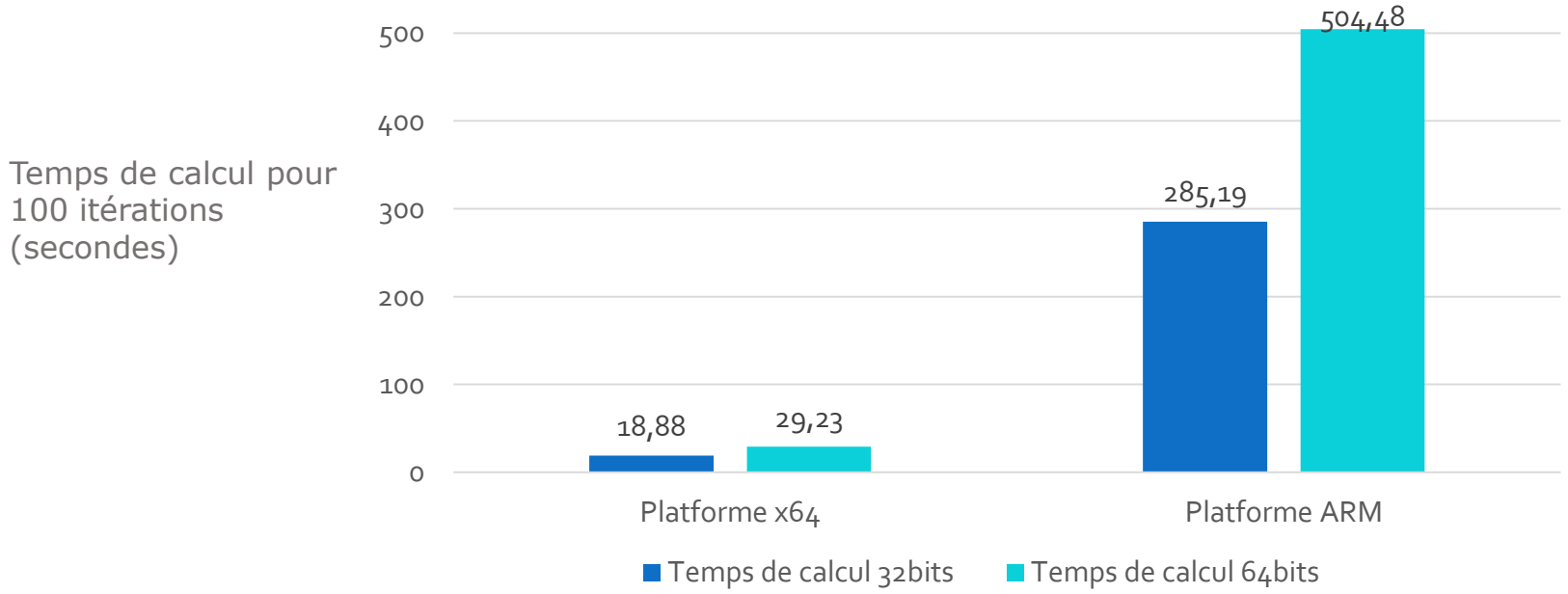
- Traitement de données – optimisation du calcul de distance (pour mesure angle) :
  - Méthode de ***fast square root (méthode d'approximation rapide)***

Temps de calcul : racines carrées de 1 à 10e7 (ms)

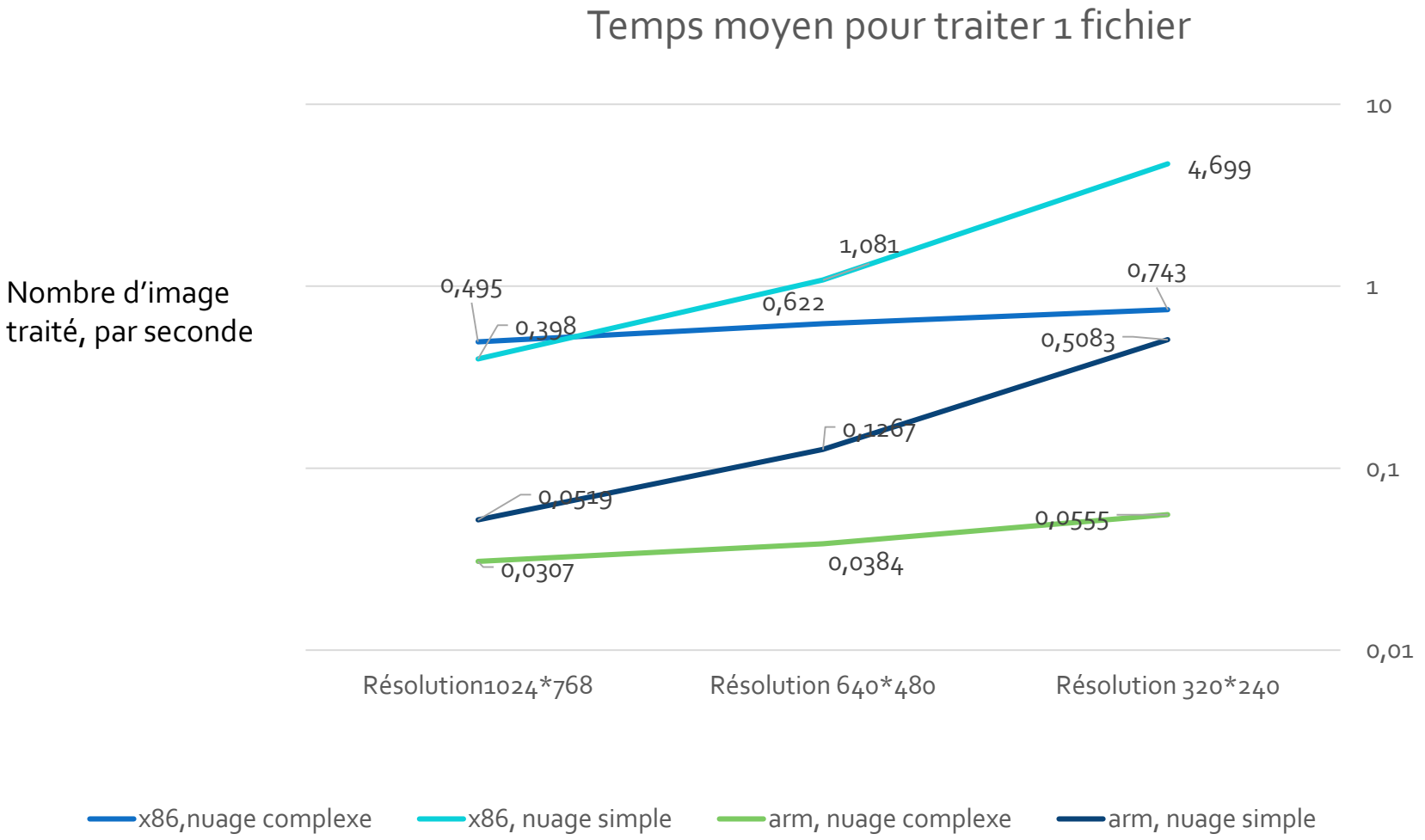


Les optimisations

- Traitement de données :
  - Utilisation de données en 32 bits au lieu de 64 bits



Performance avec différentes résolutions



Processeur Intel i5-1035G7 ; NVMe device Micron CT1000P1SSD8 ; Ubuntu 20.04.2 LTS 5.4.0-70-generic ; g++ 9,3,0 / go 1.16.3

Proposition d'architecture type