

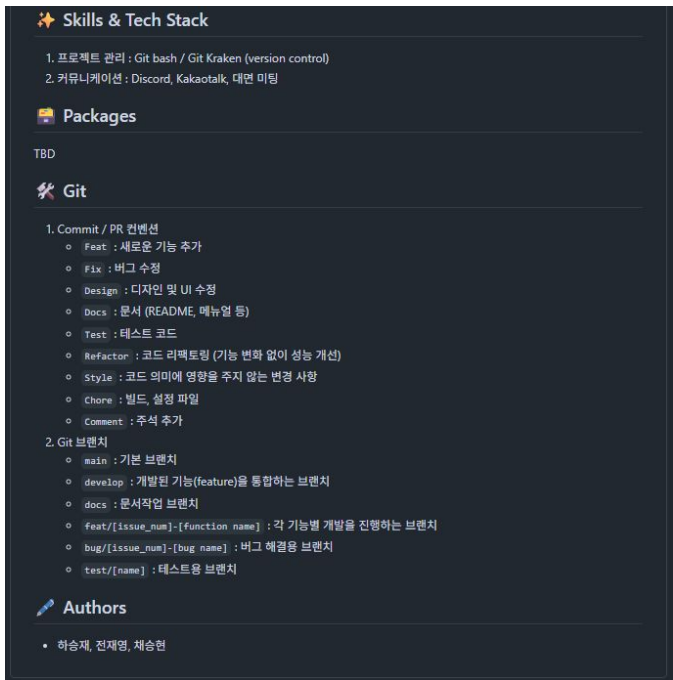
SD progress presentation

team white

Contents

1. how we manage the project?
2. master-worker connection
3. sorting algorithm
4. project class structure - class diagram
5. weekly progress & milestone

1. Management



github: version control system

+ **gitkraken, fork**

discord: save file, online meeting

kakaotalk: simple communication

google drive: make ppt file, report

boardmix: make UML

git convention

2.a. sampling

[illegible]

gensort

```
test_val_sorl X test_val_sorl X test_val_sorl X  
test_val_sorl  
1 |jw|EBj0| 00000000000000000000000000000E4  
2 |A9d|T|-| 00000000000000000000000000000B98  
3 |CqZ|XmW| 0000000000000000000000000000019  
4 |Fz;D|7d| 000000000000000000000000000001AB  
5 |TFUv|-8n^ 000000000000000000000000000001FF  
6 |OB|P|Nrv| 0000000000000000000000000000014A  
7 |p3|Z|7v| 0000000000000000000000000000021  
8 |13|f|-0 00000000000000000000000000000385  
9 |9wSc3_1| 00000000000000000000000000000380  
10 |ma|b|^f8^ 00000000000000000000000000000000  
11 |^N|L|-tM| 00000000000000000000000000000000  
12 |T;-Q|0|bq| 00000000000000000000000000000000  
13 |tp(L|-7L| 00000000000000000000000000000000  
14 |uhT|qc^-| 00000000000000000000000000000000  
15 |ujA|L_E| 00000000000000000000000000000000  
16 |r^Eh8|D~ 00000000000000000000000000000000  
17 |fu_b|zu|U| 00000000000000000000000000000000  
18 |cvr|v|sr$| 00000000000000000000000000000000  
19 |Lq|v|n7| 00000000000000000000000000000000  
20 |V5|258|ql| 00000000000000000000000000000000  
21 |I|NE|S|yq| 00000000000000000000000000000000  
22 |fd-ZU|JEN| 00000000000000000000000000000000  
23 |f^|e|-8|P| 00000000000000000000000000000000  
24 |frx|-a|16| 00000000000000000000000000000000  
25 |f^ru^v|Sp| 00000000000000000000000000000000  
26 |r^duq|Zc| 00000000000000000000000000000000  
27 |K|84|ja1| 00000000000000000000000000000000  
28 |vd|f|frA| 00000000000000000000000000000000  
29 |1huZ|-xgk| 00000000000000000000000000000000  
30 |^de=dlZ| 00000000000000000000000000000000  
31 |Zk~-R_y| 00000000000000000000000000000000  
32 |tf|r|-jm| 00000000000000000000000000000000  
33 |wd|f|EmG| 00000000000000000000000000000000  
34 |pv_XSpG3| 00000000000000000000000000000000
```

```
Windows PowerShell x white@vm-1-master:~/valsort x + v
[white@vm-1-master valsort]$ ls
valsort
[white@vm-1-master valsort]$ chmod +x valsort
[white@vm-1-master valsort]$ ls
valsort
[white@vm-1-master valsort]$ |
```

valsort

2.b. Build Libraries Setting

OS: window 11

+ **WSL**(CentOS 7 or Ubuntu)

Java: jdk 8 (JDK 8u202)

Scala: scala 3.3.7

Libraries: Scalapb, netty, protobuf, gRPC

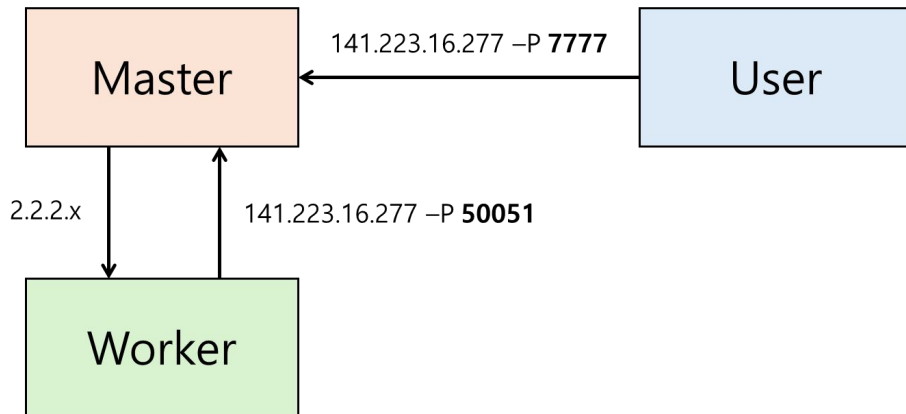
```
white@vm-1-master:~  
[white@vm-1-master ~]$ cat /etc/os-release  
NAME="CentOS Linux"  
VERSION="7 (Core)"  
ID="centos"  
ID_LIKE="rhel fedora"  
VERSION_ID="7"  
PRETTY_NAME="CentOS Linux 7 (Core)"
```

```
white@vm01: ~  
white@vm01:~$ cat /etc/os-release  
NAME="Ubuntu"  
VERSION="18.04.3 LTS (Bionic Beaver)"  
ID=ubuntu  
ID_LIKE=debian  
PRETTY_NAME="Ubuntu 18.04.3 LTS"
```

master uses CentOS, **worker** uses Ubuntu

2.c. server connection test

Host server (0.0.0.0 -P 50051)



```
white@vm-1-master:~$ ls
332project-assembly-0.1.0-SNAPSHOT.jar server.log
white@vm-1-master:~$ rm server.log
white@vm-1-master:~$ ls
332project-assembly-0.1.0-SNAPSHOT.jar
white@vm-1-master:~$ scp 332project-assembly-0.1.0-SNAPSHOT.jar white@2.2.2.101:~/
332project-assembly-0.1.0-SNAPSHOT.jar 100% 23MB 136.8
white@vm-1-master:~$ nohup java -cp 332project-assembly-0.1.0-SNAPSHOT.jar com.example.GreeterServer &
[1] 14653
white@vm-1-master:~$ tail -f server.log
nohup: ignoring input
gRPC 서버가 0.0.0.0:50051 에서 시작되었습니다.
워커 노드의 접속을 대기합니다...

white@vm01:~$ ls
332project-assembly-0.1.0-SNAPSHOT.jar
white@vm01:~$ java -cp 332project-assembly-0.1.0-SNAPSHOT.jar com.example.GreeterClient
마스터 서버가 (141.223.16.227:50051)로 요청 전송: Worker Node (vm01)
클라이언트 종료.
예러 발생 (방화벽 확인): UNAVAILABLE: Channel shutdownNow invoked
white@vm01:~$
```

oh...

3.a. sample sort

step	to do
sampling	for each worker, sample random K data (ex. first 1000 of data).
partitioning	master collects and sorts all samples, then finds the pivots (N-1) to divide them into ranges corresponding to the number of workers (N). → informs this pivot information back to all workers.
shuffle	each worker compares all its data against the pivots provided. then send them to each worker.
merge	each worker bother the sended data and sort them. → combine the sorted data from the workers in order.

in short, **quick sort with multiple pivots!**

3.a. sample sort - **if worker is dead**

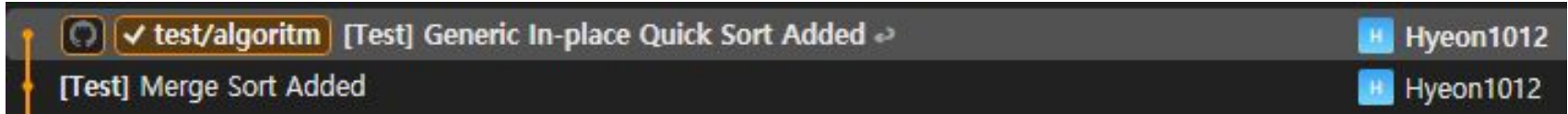
if worker is dead, worker is going to resurrect soon.

but **input data will be still remained.**

a resurrected worker gets the overall progress of sample sort at the master.

→ **do something**

3.b. merge sort vs quick sort

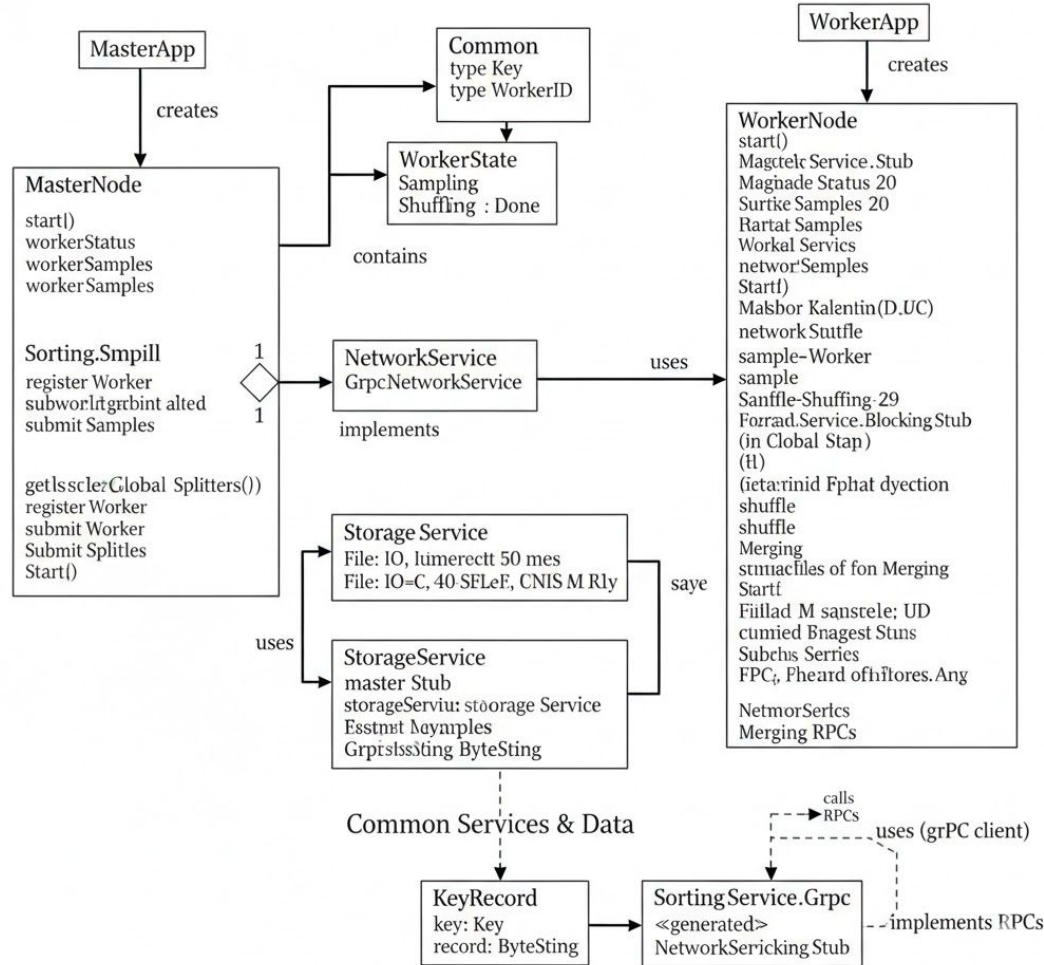


```
✓ test/algorithm [Test] Generic In-place Quick Sort Added Hyeon1012
[Test] Merge Sort Added Hyeon1012
```

```
[Test] Generic In-place Quick Sort Added
- merge sort removed
- I think that in-place quick sort is more efficient than merge sort because it can save memory
```

in fact, **quick sort** is more efficient/faster than merge sort for smaller datasets
so, I implemented **generic in-place quick sort**

Class Diagram: Master-Worker Sorting Application



Review of Weekly Progress

Week 1 Setup & Planning: Project initialization, GitHub repository setup, and establishing code conventions and meeting schedules.

Week 2 System Architecture: Designed the high-level system architecture and discussed core concepts for distributed sorting and tech stack.

Week 3 Environment & Tooling: Configured the development environment, secured VPN/IP access, and set up essential tooling (UML, IntelliJ).

Week 4 Core Prototyping: Began implementation with gRPC/protobuf (using ScalaPB), created Master/Worker prototypes, and generated sample data.

Week 5 Sorting Logic & Integration: Drafted class diagrams, implemented core sorting logic (QuickSort), and prepared for code integration and testing.

MileStone

Milestones

New milestone

Open5Closed0

Sort

Test running program on multiple machines

Overdue by 3 day(s)

Due by November 14, 2025

0% complete

0 open

0 closed

Implement distributed Sorting algorithm

Overdue by 3 day(s)

Due by November 14, 2025

0/1 issues closed

0% complete

1 open

0 closed

System test, all components in hands

Due by December 5, 2025

0% complete

0 open

0 closed

Component test and early system test

Due by November 21, 2025

0% complete

0 open

0 closed

Project Deadline

Due by December 7, 2025

0% complete

0 open

0 closed

Milestones

New milestone

Open5Closed0

Sort

Implement distributed Sorting algorithm

Due by November 25, 2025

0/1 issues closed

0% complete

1 open

0 closed

Test running program on multiple machines

Due by November 25, 2025

0% complete

0 open

0 closed

Component test and early system test

Due by November 28, 2025

0% complete

0 open

0 closed

System test, all components in hands

Due by December 5, 2025

0% complete

0 open

0 closed

Project Deadline

Due by December 7, 2025

0% complete

0 open

0 closed