Progress Presentation #2

Distributed Sorting System

Team Green

Retrospective

- Iteration 3
- Iteration 4
- Iteration 5

Milestones

- Data Abstraction
- Sampling
- Sorting
- Coordinating

Design and Details

- Data Abstraction
- gRPC Services
- Master
- Server
- Libararies

POSTECH

Retrospective

Iterations

POSTECH

Iteration 3

Retrospective

- Setup scalatest (#26, #28)
- Init Subprojects (#27)
  - Utils, Core, RPC, Master, Worker
- Implement Data Abstractions (#29, #30, #31, #32, #33)
  - Key, Record, WorkerMetadata, MasterMetadata, Block
- Implement Basic Operations of Block (#34, #35, #36)
  - Sort, Sample, File Binding, Partition, Merge

Iteration 4

Retrospective

- Refactor Block and Partition (#46, #47)
- Test gensort (#49)
- Setup ScalaPB (#51)
- Define gRPC Services (#52)
  - Master Service, Worker Service, Exchange Service.

Iteration 5

Retrospective

- Refactor Services (#62)
- Implement Servers (#63, #64, #65)
  - Worker Service, Master Service, Exchange Service.
- Implement CLI (#66)

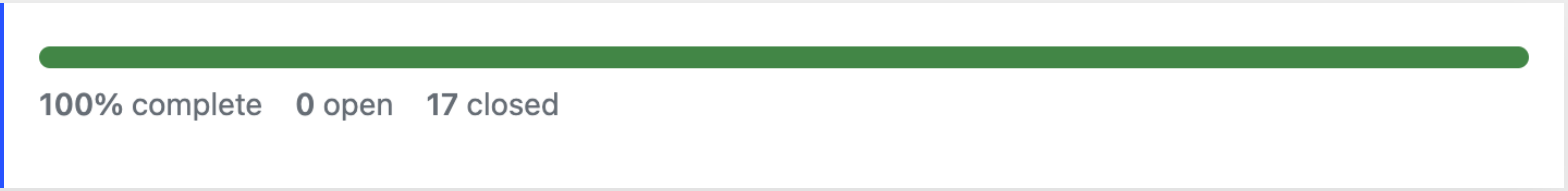
Milestones

Progress

Data Abstraction

Milestones

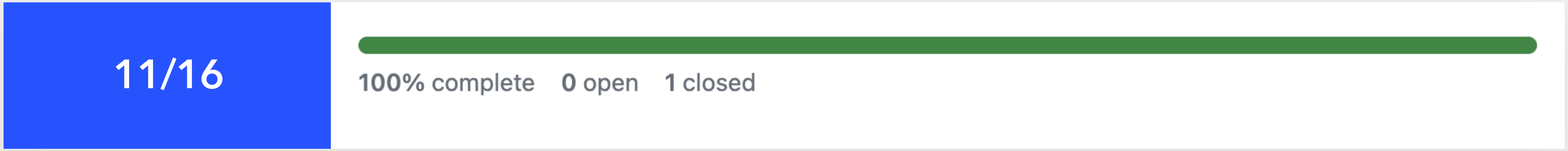
11/9


100% complete 0 open 17 closed

- Key should be constructible from bytes.
- Record should be constructible from bytes.
- Record should be constructible from bytes.
- Block should be constructible from bytes or files.
- Block should be writable to a file.

Sampling

Milestones


11/16

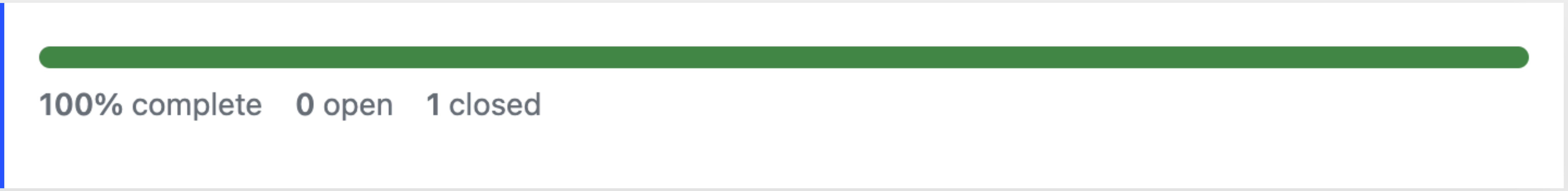
100% complete 0 open 1 closed

- Blocks should be able to be sampled.

Sorting

Milestones

11/16

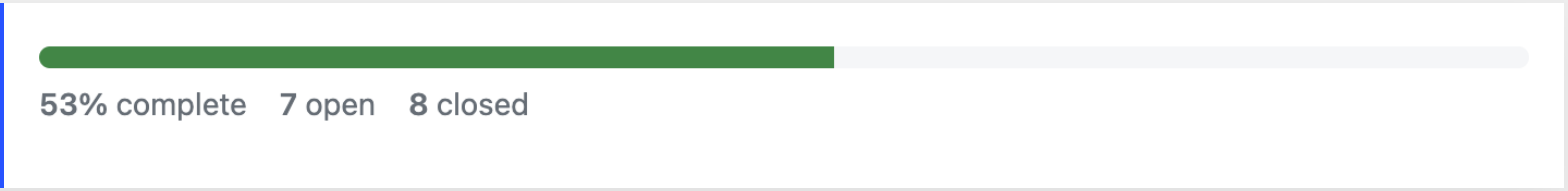

100% complete 0 open 1 closed

- Keys should be comparable.
- Records should be comparable.
- Block should be sortable.

Coordinating

Milestones

11/30


53% complete 7 open 8 closed

- Workers should be able to make partitions and merge them.
- Each client should be able to communicate with server.
- Workers should be able to exchange their partitions.
- Worker should be able to process master's requests.

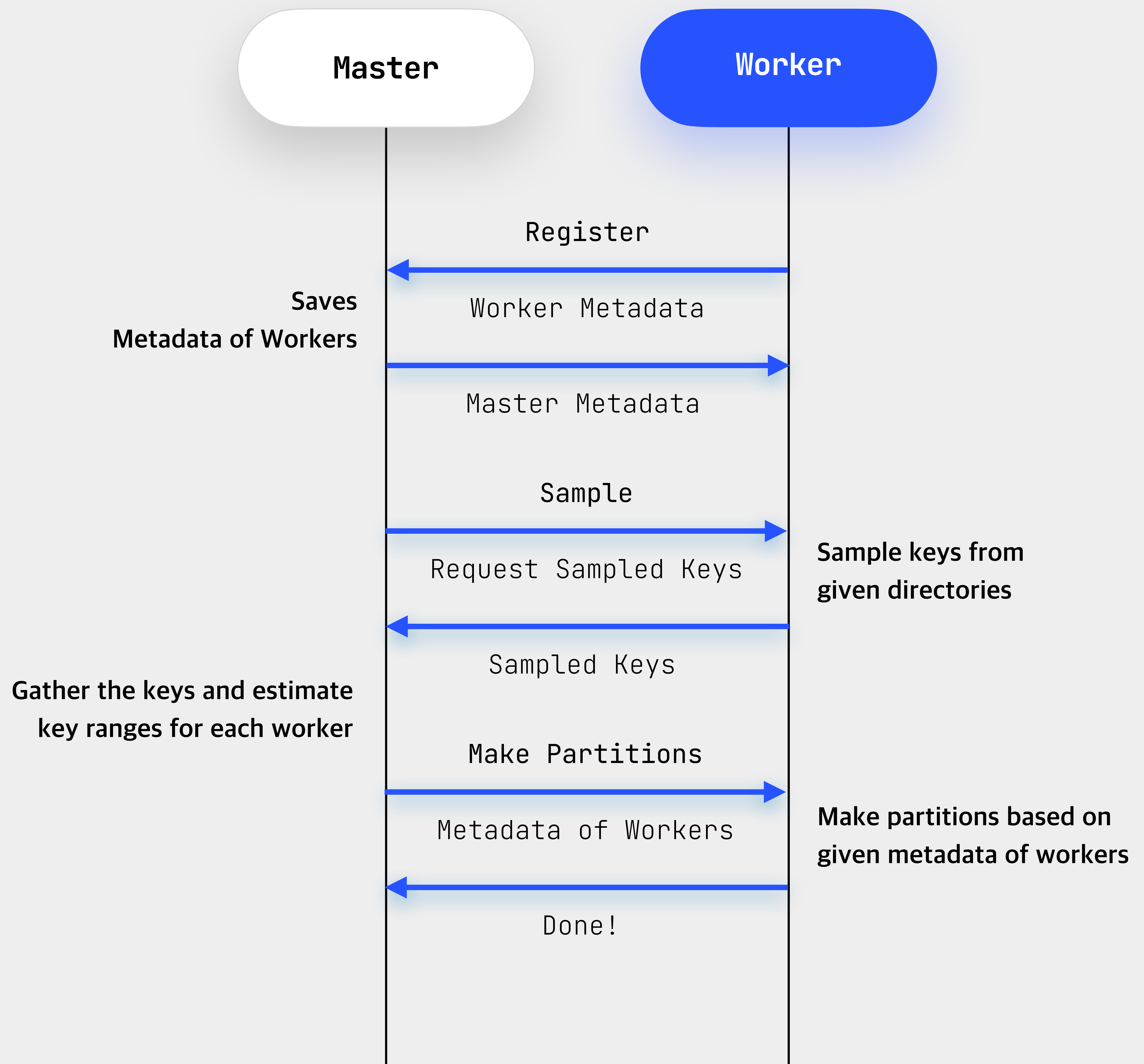
Design and Details

Data, Protocol, Worker and Master

POSTECH

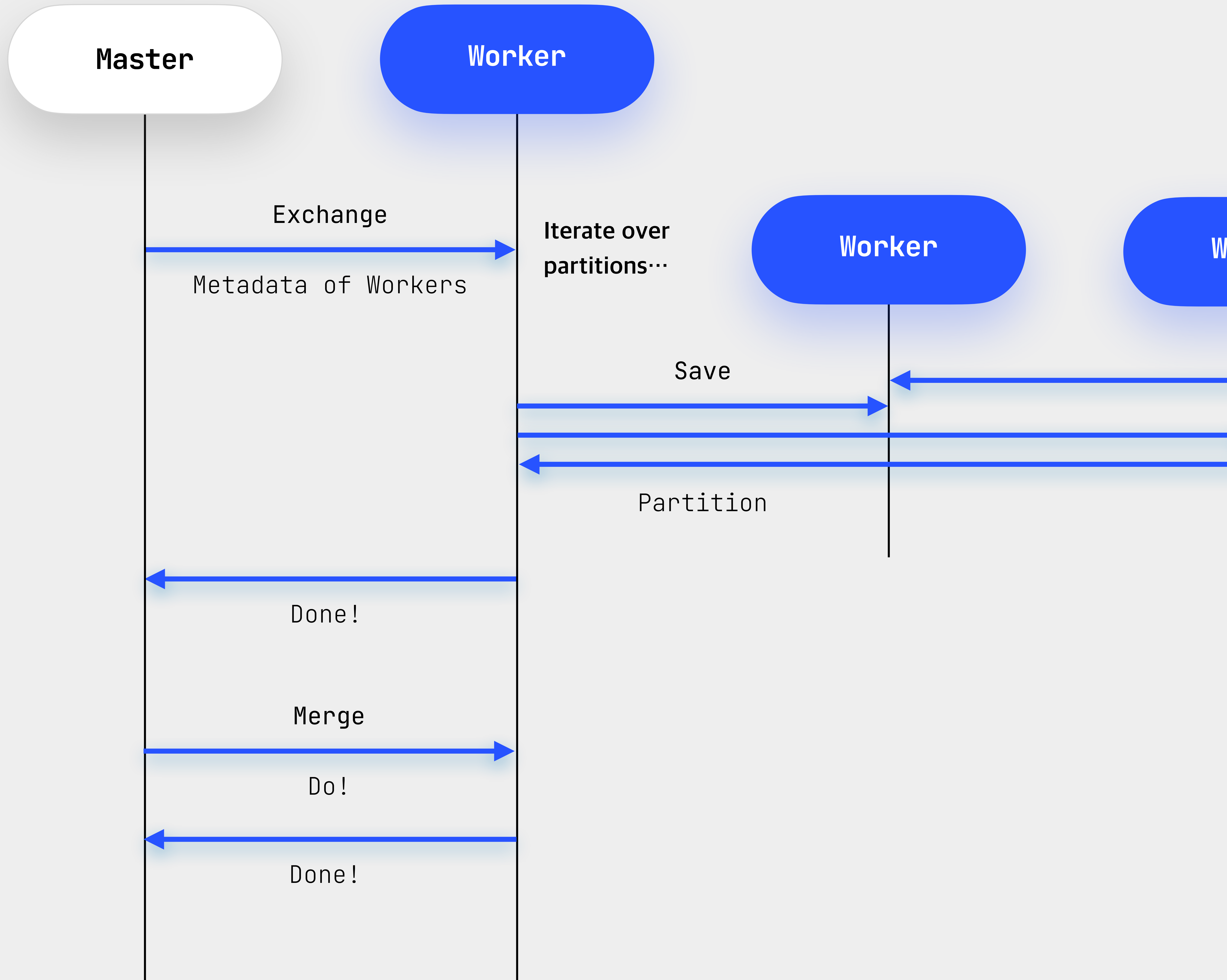
Overview

Design and Details



Overview

Design and Details



Data Abstraction

Design and Details

- Metadata
- Key
- Key Range
- Record
- Block
- Partition

Metadata

Design and Details

```scala
trait Node {
  val host: String
  val port: Int
}

case class MasterMetadata(host: String, port: Int) extends Node

case class WorkerMetadata(host: String, port: Int, keyRange: Option[KeyRange])
    extends Node
```

Key

Design and Details

```scala
class Key(val underlying: Array[Byte]) extends AnyVal with Ordered[Key] {
  def is(that: Key): Boolean

  override def compare(that: Key): Int
}
```

Key Range

Design and Details

```scala
case class KeyRange(from: Key, to: Key) {
  def includes(key: Key): Boolean

  def includes(record: Record): Boolean
}
```

Record

Design and Details

```scala
class Record(val key: Key, val value: Array[Byte]) extends Ordered[Record] {
  def is(that: Record): Boolean

  def toChars: Array[Char]

  override def compare(that: Record): Int
}
```

Record

Design and Details

```scala
object Record {
  def fromString(string: String, keyLength: Int = 10): Record

  def fromBytes(bytes: Array[Byte], keyLength: Int = 10): Record

  def fromBytesToRecords(
      bytes: LazyList[Byte], keyLength: Int = 10, valueLength: Int = 90
  ): LazyList[Record]

  def sampleWithInterval(
      records: LazyList[Record], interval: Int = 10
  ): LazyList[Key]
}
```

Block

Design and Details

```scala
class Block(val records: LazyList[Record]) extends AnyVal {
  def toChars: LazyList[Char]

  def writeTo(path: Path): File

  def filterByKeyRange(keyRange: KeyRange): Block

  def partition(keyRange: KeyRange): Partition
  def partition(keyRanges: List[KeyRange]): List[Partition]

  def sort(): Block

  def sample(): LazyList[Key]
}
```

Block

Design and Details

```
object Block {
  def fromBytes(
      bytes: LazyList[Byte], keyLength: Int = 10, valueLength: Int = 90
  ): Block

  def fromSource(
      source: Source, keyLength: Int = 10, valueLength: Int = 90
  ): Block

  def fromPath(
      path: Path, keyLength: Int = 10, valueLength: Int = 90
  ): Block
}
```

Partition

Design and Details

```scala
package object core {
  type Partition = (KeyRange, Block)
}
```

gRPC Services

Design and Details

- Master Service
- Worker Service
- Exchange Service

Master Service

Design and Details

```
service Master {
    rpc Register (RegisterRequest) returns (RegisterReply) {}
}
```

Worker Service

Design and Details

```
service Worker {
  rpc Sample (SampleRequest) returns (SampleReply) {}
  rpc Partition (PartitionRequest) returns (PartitionReply) {}
  rpc Exchange (ExchangeRequest) returns (ExchangeReply) {}
  rpc Merge (MergeRequest) returns (MergeReply) {}
}
```

Exchange Service

Design and Details

```
service Exchange {
    rpc SaveRecords (SaveRecordsRequest) returns (SaveRecordsReply) {}
}
```

Master

Design and Details

- Master
- Master Server

Master

Design and Details

```scala
object Master {
  def main(args: Array[String]): Unit = {
    val numberOfWorker = Try(args(0).toInt).getOrElse { ... }

    val server = new MasterServer(...)
    server.startServer()

    // TODO: save worker metadata and interact with workers
  }
}
```

Master Server

Design and Details

```scala
class MasterServer(executionContext: ExecutionContext) { self ⇒
  private[this] val server: Server

  private val workers: List[WorkerMetadata]

  private def start(): Unit

  private def stop(): Unit

  private class MasterImpl extends MasterGrpc.Master {
    override def register(request: RegisterRequest): Future[RegisterReply]
  }
}
```

Worker

Design and Details

- Worker
- Worker Server
- Exchange Server

Worker

Design and Details

```scala
object Worker {
  def main(args: Array[String]): Unit = {
    // Omitted; Parse arguments

    val client = MasterClient(ip, port)

    client.register(workerMetadata)

    // TODO: Spawn worker server
  }
}
```

Worker Server

Design and Details

```scala
class WorkerServer(executionContext: ExecutionContext) { self =>
  // Omitted; Duplicated with MasterServer

  private class WorkerImpl extends WorkerGrpc.Worker {
    override def sample(request: SampleRequest): Future[SampleReply]

    override def partition(request: PartitionRequest): Future[PartitionReply]

    override def exchange(request: ExchangeRequest): Future[ExchangeReply]
  }
}
```

Exchange Server

Design and Details

```
class ExchangeServer(executionContext: ExecutionContext) { self ⇒
  // Omitted; Duplicated with MasterServer

  private class ExchangeImpl extends ExchangeGrpc.Exchange {
    // TODO: Implement
    override def saveRecords(
      request: SaveRecordsRequest
    ): Future[SaveRecordsReply]
  }
}
```

Libraries

Design and Details

- Scalatest, Scalafmt, Scalafix
  - For testing and linting.
- ScalaPB
  - For compiling protobufs and supporting gRPC.
- rxJava
  - TODO; For observing and reacting to state changes from outside.
- log4j
  - TODO; For logging.

Progress Presentation #2

Distributed Sorting System

Gwon Minjae, Dept. of Computer Science & Engineering, POSTECH.

Lee Jiwon, Dept. of Computer Science & Engineering, POSTECH.

Ha Taehyeok, Dept. of Computer Science & Engineering, POSTECH.

Advanced Programming, 2023.