AKKA Cluster

Eureka - Zuul - Ribon .

. RestAPI . RestAPI AKKA

Kafka . .

, .

. AKKA

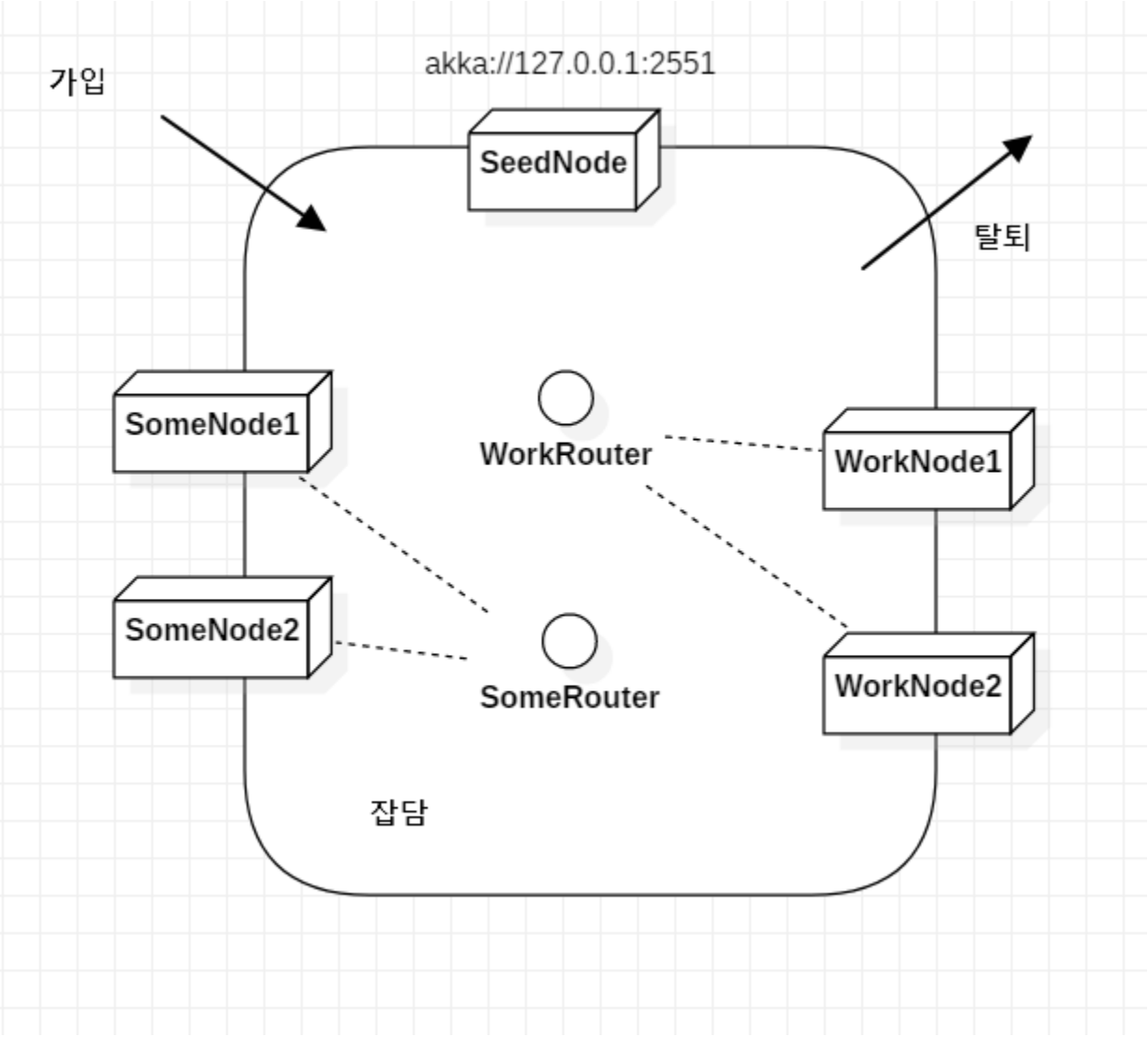
.

? .

,

.

: 04. Clustering



, . /

( ) 1 .

. ()

. ip-port .

WorkRouter WorkNode .

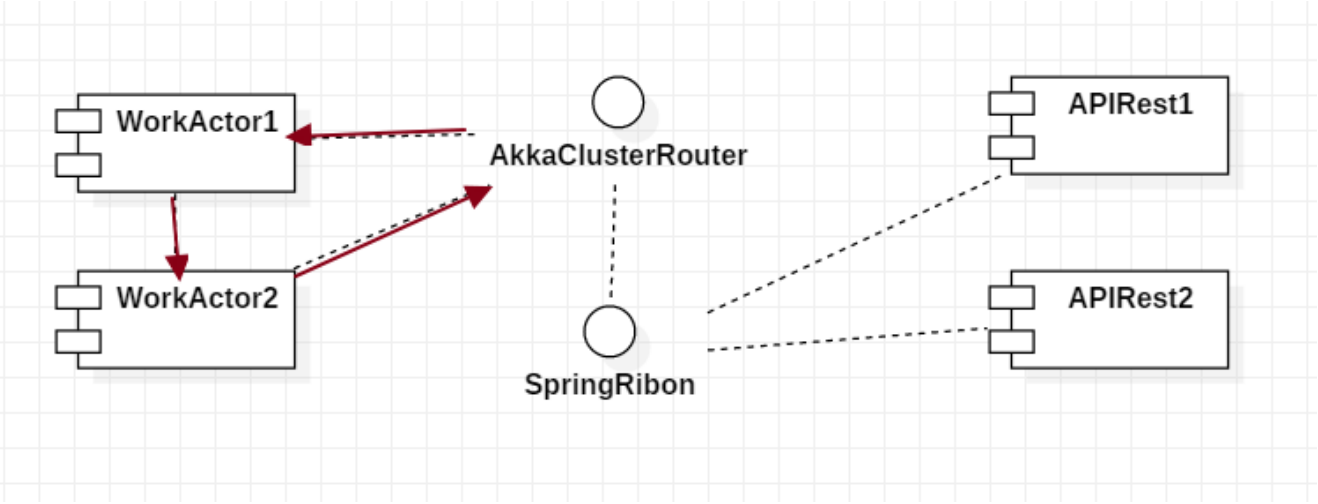
.

.

.

.

AKKA



Actor1 Actor2 Ribon

Actor Ribon Rest .

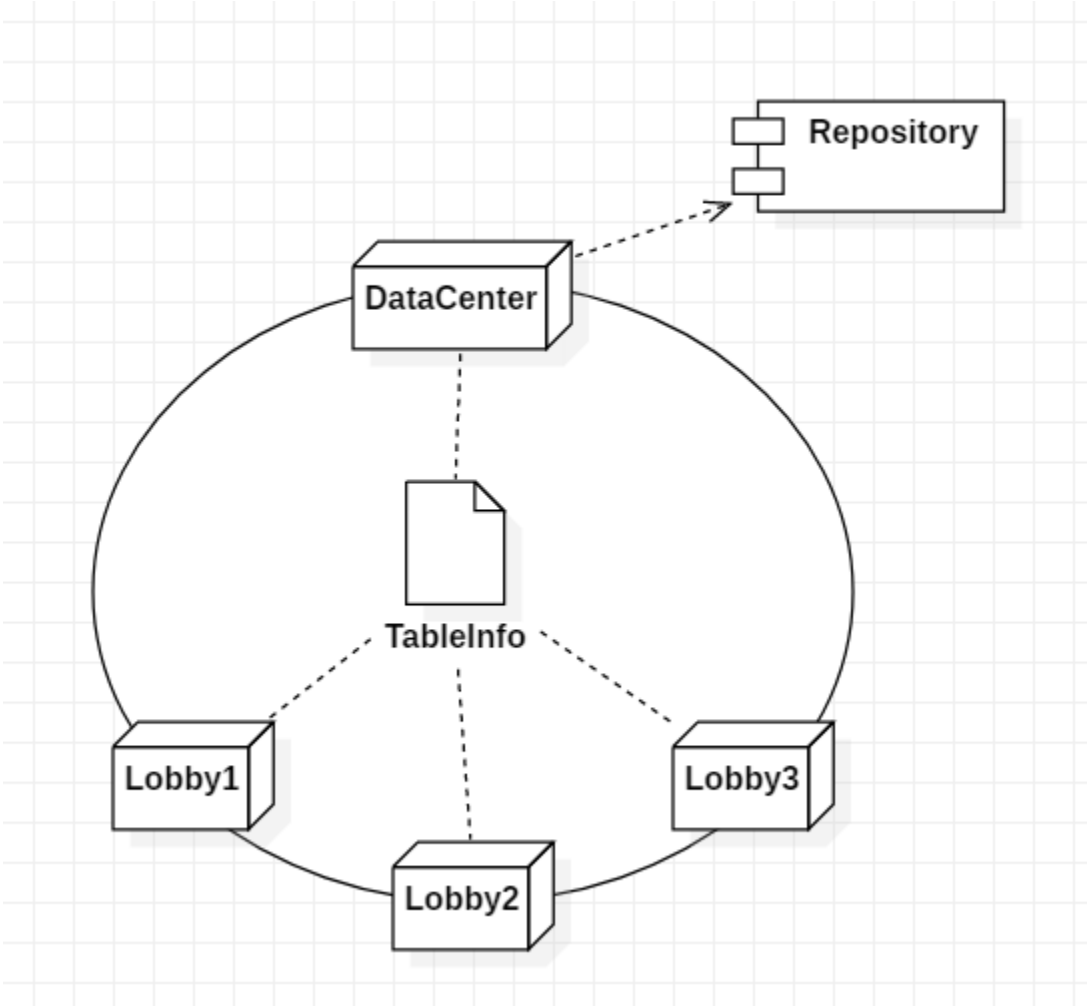
Spring Cloud ?

Spring RestAPI . ,

Actor .

AKKA .

:



.

- : (10), Read . - .
- : , Read

Lobby RestAPI

. RestAPI .

git : https://github.com/psmon/springcloud/tree/master/akka-cluster

test : https://github.com/psmon/springcloud/blob/master/akka-cluster/src/test/java/com/webnori/psmon/cloudspring/akkacluster/actor/ClusterClientTest.java

, .

, Lighthouse .

```
akka {
  actor {
    provider = cluster
  }
  remote {
    enabled-transports = ["akka.remote.netty.tcp"]
    netty.tcp {
      hostname = "127.0.0.1"
      port = 2552
    }
  }

  cluster {
    seed-nodes = [
      "akka.tcp://ClusterSystem@127.0.0.1:2552"
    ]

    roles = ["seed"]

    role{
      seed.min-nr-of-members=1
    }

    # auto downing is NOT safe for production deployments.
    # you may want to use it during development, read more about it in the docs.
    #
    auto-down-unreachable-after = 10s
  }
}

# Enable metrics extension in akka-cluster-metrics.
akka.extensions=["akka.cluster.metrics.ClusterMetricsExtension"]
```

, .

```
@Override
public Receive createReceive() {
  return receiveBuilder()
          .match(ClusterEvent.MemberUp.class, mUp -> {
              log.info("Member is Up: {}", mUp.member());
          })
          .match(ClusterEvent.UnreachableMember.class, mUnreachable -> {
              log.info("Member detected as unreachable: {}", mUnreachable.member());
          })
          .match(ClusterEvent.MemberRemoved.class, mRemoved -> {
              log.info("Member is Removed: {}", mRemoved.member());
          })
          .match(ClusterEvent.MemberEvent.class, message -> {
              // ignore
          })
          .build();
}
```

, .

IP

.

```
@Override
public void preStart() {
    // first read for sync , After that no more db load
    tableRepository.findAllTableInfos().forEach(tableEntity -> {
        TableInfo tableInfo = tableEntity.toTableInfo();
        tableInfos.add(tableInfo);
    });
}

@Override
public Receive createReceive() {
    return receiveBuilder()
            .match(TableCMD.class, s -> {
                if(s.cmdType==TableCMD.TableCMDType.SYNC_FIRST){
                    log.info("Received TableCMD message: {}", s);
                    ActorRef requestActor = getSender();
                    seqNumSync++;
                    requestActor.tell(new TableInfoList(seqNumSync,tableInfos),ActorRef.noSender());
                }
            })
            .matchAny(o -> log.info("received unknown message"))
            .build();
}


//Config
cluster {
  seed-nodes = [
    "akka.tcp://ClusterSystem@127.0.0.1:2551"
  ]

  roles = ["datacenter"]
```

git : https://github.com/psmon/springcloud/tree/master/datacenter/src/main/java/com/webnori/psmon/cloudspring/datacenter/domain/DataReplication/TableInfo

Role , .

, Update,Insert,Delete .

```
//Create a TableSync object when it joins the cluster
Cluster.get(system).registerOnMemberUp(new Runnable() {
        @Override
        public void run() {
                system.actorOf(TableInfoActor.props(),"lobbyTableSync");
        }
});


cluster {
  seed-nodes = [
    "akka.tcp://ClusterSystem@127.0.0.1:2551"
  ]

  roles = ["lobby"]

  role{
    seed.min-nr-of-members=1
    datacenter.min-nr-of-members=1
    lobby.min-nr-of-members=1
  }
```

git : https://github.com/psmon/springcloud/tree/master/lobbyapi/src/main/java/com/webnori/psmon/cloudspring/lobbyapi

lobby , (data)

. lobby .

```
ActorRef dataCenter = getContext().actorOf(FromConfig.getInstance().props(),
        "dcTableSyncRouter");


@Override
public void preStart() {
    // first requestData
    sendRefreshTable(TableCMD.TableCMDType.SYNC_FIRST);
    getContext().setReceiveTimeout(Duration.create(10, TimeUnit.SECONDS));
}

private void sendRefreshTable(TableCMD.TableCMDType cmdType) {
    dataCenter.tell(new TableCMD(cmdType),getSelf() );
}

@Override
public Receive createReceive() {
    return receiveBuilder()
            .match(TableInfoList.class, tableList -> {
                tableInfos = tableList.getValues();
                log.info(String.format("===== first Table sync, count:%d",tableInfos.size()));
            })
            .match(ReceiveTimeout.class, message -> {
                log.info("=== Timer");
                sendRefreshTable(TableCMD.TableCMDType.SYNC_FIRST);
            })
            .match(TableCMD.class, cmd -> {
                if(cmd.cmdType == TableCMD.TableCMDType.SYNC_FIRST)
                sender().tell(tableInfos,null);
            })
            .matchAny(o -> log.info("received unknown message"))
            .build();
}
```

, (, , .)

. dataCenter ip .

.

```
akka.actor.deployment {
  /lobbyTableSync/dcTableSyncRouter = {
    # Router type provided by metrics extension.
    router = cluster-metrics-adaptive-group
    # Router parameter specific for metrics extension.
    # metrics-selector = heap
    # metrics-selector = load
    # metrics-selector = cpu
    metrics-selector = mix
    #
    routees.paths = ["/user/dcTableSync"]
    cluster {
      enabled = on
      use-role = datacenter
      allow-local-routees = off
    }
  }
}
```

datacenter .

.

Spring Cloud .

AKKA

, . , .

.

?

DB ? .

AKKA ..

Akka Cluster ,

.

: https://doc.akka.io/docs/akka/2.5/cluster-metrics.html