type Server struct{

IP string

Port string

Name string

Path string

State string

currentTerm uint64

currentWaittime uint64

votefor string

}

func (s \*Server)newServer(IP string,port string,name string,path string)(\*Server,error)

func (s \*Server)start(){

s.Init()//初始化服务器

state：=s.state()

for state!=Stopped{

switch state{

case Follower:

s.followerloop()

case Candidate:

s.candidateloop()

case Leader:

s.leaderLoop()

}

state= s.state()

}

}

//跟随者接口

func (s \*server)followerloop(){

设置超时时间；

开启端口监听；

for s.state==Follower{

select{

case 服务器停止Stopped：

s.setState(Stopped)

return

case 监听到消息并分析判断消息类型：

switch{

case “JoinCluster”：

服务器的对等节点加一

case “AppendEntriesRequest”：

s.processAppendEntriesRequest()

case “RequestVote”：

s.processRequestVoteRequest()

case 回复快照：

s.processSnapshotRequest()

}

case 定时器超时：

if s.promotable(){

s.setstate(Candidate)

} else{重置定时器}

}

重置定时器

}

}

//候选者接口

func (s \*Server)candidateloop(){

s.leader=” “

获取节点当前的日志索引和任期；

for s.state==Candidate{

if（节点为投票）{

为自己投票，票数加1，当前任期加1，设置定时器并广播投票请求；

}

if （节点收集到的肯定投票数超过集群服务器数的一大半）{

s.setState(Leader);

return

}

select{

case Stopped：

s.setState(Stopped)

return

case 监听到消息并分析判断消息类型：

switch{

case “JoinCluster”：

服务器的对等节点加一

case “AppendEntriesRequest”：

s.processAppendEntriesRequest()

case “RequestVote”：

//需判断是其他节点发送的投票请求还是选票结果：发送端地址|RequestVote|当前时间|当前任期|日志索引|消息类型|标志|目的地址

if 消息类型==‘0’&&标志==true{ s.processRequestVoteRequest()}

if 消息类型==‘1’&&标志==true{票数加1}

}

case 定时器超时：

重新投票；

}

}

}

//领导者接口

/\*

msg\_str:=strings.Split(string(date[0:n]),”|”)

msg\_str[0]:发送端地址

msg\_str[1]:消息类型

msg\_str[2]:当前时间

msg\_str[3]:当前任期

msg\_str[4]:当前日志索引

msg\_str[5:当前等待时间

\*/

func (s \*Server)leaderloop(){

获取领导者当前的任期和日志索引；

启动心跳事件；

监听消息；

for s.state==Leader{

select{

case stopped：

s.setSatet(Stopped)

return

case 收到消息并判断消息类型：

switch{

case “JoinCluster”：

服务器的对等节点加一

case “AppendEntriesRequest”：

s.processAppendEntriesRequest(msg\_str[0],msg\_str[3]，msg\_str[4])

case “RequestVote”：

s.processRequestVoteRequest(msg\_str[0]，msg\_str[3]，msg\_str[4]， msg\_str[5)

case 客户端的请求（‘Command’）：

将该条请求添加到日志中，并广播请求添加日志（AppendEntries）

case “AppendEntriesResponse”

s.processAppendEntriesResponse()

}

}

}

}

func (s \*Server)processRequestVoteRequest(candidatename string,term uint64，logindex uint64，waitTime uint64){

if term<s.Term(){//当前服务器任期大于请求者任期，不投票

return 2,false

}

if term>s.Term(){ //当前服务器任期小于请求者任期，任期需要改变

s.updateCurrentTerm(term,” “)

}else if s.votefor!=” “&&s.votefor!=candidatename{

return 5，false

}

lastIndex,lastTerm:=s.log.lastInfo()

if lastIndex>logindex||lastTerm>term{

return 3,false

}

if s.WaitTime()>waitTime{

return 4,false

}

s.votsfor=candidatename //可以投票，投给请求的候选者

rerun 1,true

}

func (s \*Server)processAppendEntriesRequest(){

}

func (s \*Server)processAppendEntriesResponse (){

}