调度中心集群

调度请求

(queue)

Redis

ActiveMq

任务线程

执行器集群

ShardingService

日志管理

实时监控

系统管理

产品管理

执行器管理

流程管理

任务管理

用户管理

算法管理

分片广播调度

调度器(quartz)

回调服务(API)

注册服务(API)

Dubbo + Zookeeper

xxl -job内封的RPC

注册线程

执行器服务(jetty)

回调线程

调度结果(queue)

dubbo

DubboService

1.分片广播调用执行

2.分层操作(产品、产品详情、ftp、监控、流程、流程算法、节点信息、redis数据、用户、角色、资源等服务)

redis

1.缓存节点信息

1) 调度任务队列

2) 节点排队任务

3) 节点运行任务

4) 节点串行队列

5) 节点串行队列列表

6) 节点运行核

activeMQ

1.在执行器上暂存需要执行的消息

路由策略

1. FIRST("第一个", new ExecutorRouteFirst()),

2. LAST("最后一个", new ExecutorRouteLast()),

3. ROUND("轮询", new ExecutorRouteRound()),

4. RANDOM("随机", new ExecutorRouteRandom()),

5. CONSISTENT\_HASH("一致性HASH", new ExecutorRouteConsistentHash()),

6. LEAST\_FREQUENTLY\_USED("最不经常使用", new ExecutorRouteLFU()),

7. LEAST\_RECENTLY\_USED("最近最久未使用", new ExecutorRouteLRU()),

8. FAILOVER("故障转移", new ExecutorRouteFailover()),

9. BUSYOVER("忙碌转移", new ExecutorRouteBusyover()),

10. SHARDING\_BROADCAST("分片广播", null);

调度中心的任务

1.回调(保存日志)

callback(List<HandleCallbackParam> callbackParamList)

2.注册

registry(RegistryParam registryParam)

3.移除注册

registryRemove(RegistryParam registryParam)

4.执行任务 基于 quartz scheduler

triggerJob(int jobId)

执行器的任务

1.beat()

2.判断是否空闲

idleBeat(int jobId)

3.删除任务

kill(int jobId)

4.日志

log(long logDateTim, int logId, int fromLineNum)

5.执行任务

run(TriggerParam triggerParam)

6.获取节点信息

getSystemMessage()