(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 103442049 B
(45)授权公告日 2016.08.31

(21)申请号 201310367864.2

(22)申请日 2013.08.22

(73)专利权人 浪潮电子信息产业股份有限公司
地址 250014 山东省济南市高新区舜雅路1036号

(72)发明人 王恩东 张东 刘正伟 亓开元 刘俊朋 郭峰 刘成平 高飞 朱波

(51)Int.Cl.
H04L 29/08(2006.01)

(56)对比文件
CN 1722664 A,2006.01.18,
US 2012110570 A1,2012.05.03,
CN 102970332 A,2013.03.13,
CN 102411541 A,2012.04.11,
李明杰.浪潮云海OS V2.0产品解析(上).《科技浪潮》.2012,第30-31页.

审查员 楚彩虹

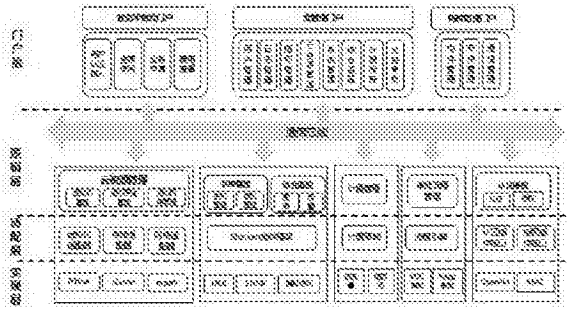
权利要求书3页 说明书8页 附图5页

(54)发明名称

一种面向构件的混合型云操作系统体系结构及其通信方法

(57)摘要

本发明提供一种面向构件的混合型云操作系统体系结构及其通信方法,基于层次、对象和消息模型建立混合型架构,并采用面向构件思想对组成构件及其处理环境进行管理,在此基础上,对构件处理集群进行高效路由、读写分离和负载均衡,满足对云操作系统地的开放兼容、松耦合和可扩展需求,解决现有云操作系统的自管理问题、构件水平伸缩问题和有状态构件的高可用问题,与现有技术相比,本发明提出的面向构件的混合型架构完善了开放兼容、可扩展、松耦合的构件化云操作系统体系结构,并通过构件水平伸缩集群和高可用集群的通信方法保障了云操作系统的可伸缩性和高可用性。



CN 103442049 B

1.一种面向构件的混合型云操作系统,其特征在于基于层次、对象和消息模型建立混合型架构,并采用面向构件思想对组成构件及其处理环境进行管理,在此基础上,对构件处理集群进行高效路由、读写分离和负载均衡,满足对云操作系统的开放兼容、松耦合和可扩展需求,解决现有云操作系统的自管理问题、构件水平伸缩问题和有状态构件的高可用问题;

从层次模型角度,系统自上至下分为门户层、逻辑层、适配层和实现层,各层相对独立,通过在各层分别定义标准接口增强开放性,通过在各层适配不同功能实现增强兼容性;

从对象模型角度,云操作系统由云门户、云管理门户、云资源管理、监控管理、计量计费、业务审批、授权认证功能模块组成,各功能组件通过基于Rest消息的调用进行通信,可自由组合,按需分布部署,并且可根据需求增值开发新模块,通过在逻辑层的不同模块间实现互操作增强平台的可扩展性,引入基于消息的通信方式支持异步调用,使用消息通信接口JMS传输Rest消息,使系统架构进一步解耦,在此基础上,采用面向构件设计,构件管理门户负责管理构件的元数据并对其运行状态进行监控;

对象架构虽然能够实现按需扩展和分布部署,但属于RPC(RemoteProcessCall)同步通信方式,发送端等待接收端返回后才能继续执行,双方进程紧耦合,在对象架构的基础上引入基于消息的通信方式,使用消息通信接口JMS传输Rest信息,使得发送和接收端生命周期可以不同,并支持异步调用,使系统架构进一步解耦;

基于上述混合型架构的云操作系统能够满足开放兼容和扩展需求,在此基础上,基于面向构件设计思想,构件管理门户负责管理构件的元数据信息,支持注册、删除、修改和查询操作,其中,

构件是一个三元组包括:名称、服务集合、访问地址、描述;

服务是一个四元组,包括:名称、类型、消息协议、参数列表、key名称、功能描述、非功能描述;

除对构件进行描述和管理,构件管理模块进一步对其处理环境进行监控,为保障构件的可伸缩性和可用性提供基础服务,完善云操作系统的自管理能力,在构件注册时,系统为其分配用户名user、密码psw以及唯一的构件id,之后构件处理集群的接入过程为

1)处理集群向地址为url的系统总线发起接入请求,系统总线验证接入节点的用户名、密码和id,如验证通过,建立连接,代码为:

connection=ConnectionFactory.createConnection(user,psw,url);

2)建立一个写操作主题,构件的每个处理节点向该主题订阅写操作;

write_topic=session.createTopic(id+"WRITE_TOPIC");

write_topic_consumer=session.createConsumer(write_topic);

3)构件在writeTopicListener的onMessage方法中实现写处理,并向系统总线注册;write_topic_consumer.setMessageListener(writeTopicListener);

4)根据构件处理节点数mum建立读操作队列组,每个处理节点对应一个队列订阅读操作read_queue=session.createMultiQueue(num,id+"READ_QUEUE");read_queue_consumer=session.createConsumer(read_queue);

5)在readQueueListener的onMessage方法中实现具体的读处理功能,并向系统总线注册read_queue_consumer.setMessageListener(readQueueListener);

在上述读写分离队列组基础上，在构件管理模块中对各构件的服务类型进行区分，设定其为幂等或非幂等，幂等操作属于无状态操作，在同一状态下每次执行的结果相同；非幂等操作属于有状态操作，同一状态下每次执行的结果不同，路由器根据服务类型进行路由，非幂等操作发送到唯一的写队列，幂等操作根据负载均衡策略发送到不同的读队列；

读操作负载均衡流程过程为

1)计算节点的处理能力；

若节点i的CPU频率、内存容量和I/O带宽分别为C_i，M_i和B_i，集群的各种资源为节点各种资源之和，即$C=\Sigma C_i$，$M=\Sigma M_i$，$B=\Sigma B_i$；

则节点i的CPU权值为$W_i^{CPU}=C_i/C$，，内存容量$W_i^{RAM}=M_i/M$，I/O带宽$W_i^{IO}=B_i/B$；

若构件服务所需资源比例分别为p^{CPU}，p^{RAM}，p^{IO}，则节点i的处理能力为$W_i=p^{CPU}W_i^{CPU}+p^{RAM}W_i^{RAM}+p^{IO}W_i^{IO}$；

2)根据读写操作权值计算各节点的负载；

若读队列L_r同写队列L_w的读写操作开销比a，则节点i的负载$L_i=L^R{}_i+aL^W$各节点的负载状态$S_i=L_i/W_i$；

3)选择负载最轻的节点进行路由；

写操作采用流水方式进行，以提高数据写入效率，其过程为节点1首先写入数据，写入一个数据分片64KB后，在继续接收数据的同时向节点2转发已写入的64K数据，节点2至节点n以相同方式接收和转发数据，直到节点n写入最后一个不超过64KB数据分片；

在上述通信方法基础上，节点监控模块进一步将构件处理节点的加入、退出、失效和恢复事件发送给构件管理模块，其中，节点加入事件是指为构件添加一个处理节点；节点退出事件是指为构件撤销一个处理节点；节点失效事件是指构件的一个处理节点不可用；节点恢复事件是指构件一个不可用处理节点恢复可用。

2.一种面向构件的混合型云操作系统的通信方法，其特征在于，包括高可用构件集群通信方法和水平伸缩构件集群通信方法，其中：

高可用构件集群通信方法，是构件管理模块为每个构件集群建立一个读写分离队列组，构件管理模块进一步将构件的服务类型区分为幂等或非幂等，路由器根据服务类型进行路由，非幂等操作发送到唯一的写队列，幂等操作根据负载均衡策略发送到不同的读队列，在此基础上，节点监控模块将处理节点的加入、退出、失效和恢复事件发送给构件管理模块，构件管理模块进一步根据节点变化事件调整队列结构，此方法通过读写分离和负载均衡提高有状态构件集群的通信性能，根据节点变化调整队列结构保障构件的高可用性，构件管理模块进一步根据监控模块发送的节点变化事件调整队列结构，流程如下：

1)当节点加入时，在队列组中为该节点建立一个读操作队列，该节点向该队列订阅读操作，并向写主题订阅写主题；

2)当节点退出时，删除该节点对应的读队列，关闭其对写主题的订阅；

3)当节点失效时，停止向该节点的读队列发送读请求，在写主题中为该节点保留写操作；

4)当节点恢复时，在写主题中同步写操作，恢复向该节点对应的读队列发送读操作请求；

无状态集群的负载均衡方法将操作完全均衡的分配到各个节点上，在有状态情况下将

造成处理结果不一致，上述方法通过建立读写分离队列并针对节点能力负载均衡，能够避免此问题，提高有状态构件集群的通信性能，并且根据节点变化事件调整队列结构保障了有状态集群的高可用性；

水平伸缩构件集群通信方法，是构件管理模块根据节点数为构件的每个服务建立多边形队列结构，按Hash值进行二分查找路由，算法效率由远小于关键字规模的Hash分组数目决定，提高了路由效率，并且在节点数发生变化的情况下只需调整部分节点数据状态，提高了动态伸缩效率，在上述通信方法基础上，在初始化阶段根据节点处理能力划分各节点所需处理的Hash区间，形成路由表，实现按比例分布数据状态，达到负载均衡；

构件管理模块根据节点数为构件的每个服务建立多边形队列组，每个队列对应一个Hash值区间或分组，按Hash值进行二分查找路由，路由算法如下：

1.计算key的Hash值h；

2.初始化分组下限i为1，分组上限j为分组总数m；

3.计算中间分组t＝(i+j)/2，查看分组t是否包含h；

4.若h小于当前分组下限t_l，更新上限j＝t-1，返回步骤3；

5.若h大于当前分组上限t_u，更新下限i＝t+1，返回步骤3；

6.否则，输出分组t，返回区间所在节点进行路由。

一种面向构件的混合型云操作系统体系结构及其通信方法

技术领域

[0001] 本发明涉及云计算技术领域，具体地说是一种面向构件的混合型云操作系统体系结构及其通信方法。

背景技术

[0002] 随着云计算的兴起，传统的数据中心迅速向云数据中心转型。在数据中心从初级向高级形态演进的物理资源整合、应用与虚拟化接合、自动化管理、数据中心协同四个阶段中，云操作系统(COS,Cloud Operating System)发挥着重要作用，承担着对上接口应用、对下管理硬件的中间功能，将大量的异构设备融合为逻辑资源池，动态调度给云应用，完成对终端的服务。

[0003] 云数据中心环境具有动态、异构、大规模和单点易失效的特征，因此，COS需采用广泛兼容的开放架构，既考虑对第三方软硬件的兼容性，也将二次开发纳入其中，提供完善标准的接口API；针对云计算环境对功能的动态变化需求，COS需采用可扩展的构件化设计，在虚拟化、资源调度基本构件基础上，便于运维管理、计量计费和自助服务构件的增值开发和按需部署；此外，COS还要采用伸缩性和高可用设计，达到云计算追求的规模扩展和业务连续性目标。

[0004] 针对云数据中心对COS的松耦合、可扩展、可伸缩和高可用需求，采用传统OS的单一模块架构可以实现COS模块间的高效调用，但耦合紧密、结构复杂，系统难于扩展；采用层次架构可以使各模块间的组织结构和依赖关系清晰化，提高COS的可靠性、可移植性和维护性，但软件栈层次太深使得内核过于庞大，并且模块间的耦合程度仍然较高，不适于构建分布处理环境；在上述架构基础上，开源软件OpenStack和CloudStack基于消息队列建立了松耦合的云管理架构，但缺乏面向构件的设计，无法控制构件生命周期，需要组成模块自行考虑伸缩和高可用方式，加重了模块的开发部署负担和运行开销。遵循高内聚低耦合原则，应该从COS层面增加对构件的管理并保障其可扩展性、可伸缩性和高可用性，这其中主要面对的问题是

[0005] 1.当前的云操作系统缺乏自包含性，无法对组成构件进行描述和管理，也无法对其处理环境进行动态监控。

[0006] 2.针对构件的高可用处理集群，现有通信协议基于构件的无状态性假设设计，缺乏读写分离机制和负载均衡策略，无法实现对有状态构件处理集群的高可用和高性能支持。

[0007] 3.针对构件的水平伸缩(scale-out)处理集群，现有基于树的路由算法效率受到关键字规模扩大的影响，而基于Hash的路由算法在节点变化时会造成大量的数据移动，并且缺乏均衡异构节点负载的数据分布方法。

[0008] 因此，如何在COS中提供对构件及其处理集群的管理和监控机制，以及实现消息的高效路由和负载均衡，成为COS架构中亟待解决的技术问题。

发明内容

[0009] 本发明的目的是提供一种面向构件的混合型云操作系统体系结构及其通信方法。

[0010] 本发明的目的是按以下方式实现的，基于层次、对象和消息模型建立混合型架构，并采用面向构件思想对组成构件及其处理环境进行管理，在此基础上，对构件处理集群进行高效路由、读写分离和负载均衡，满足对云操作系统的开放兼容、松耦合和可扩展需求，解决现有云操作系统的自管理问题、构件水平伸缩问题和有状态构件的高可用问题，

[0011] 从层次模型角度，系统自上至下分为门户层、逻辑层、适配层和实现层，各层相对独立，通过在各层分别定义标准接口增强开放性，通过在各层适配不同功能实现增强兼容性；

[0012] 从对象模型角度，云操作系统由云门户、云管理门户、云资源管理、监控管理、计量计费、业务审批、授权认证功能模块组成，各功能组件通过基于Rest消息的调用进行通信，可自由组合，按需分布部署，并且可根据需求增值开发新模块，通过在逻辑层的不同模块间实现互操作增强平台的可扩展性，在实施例中，最小化安装的云操作系统仅由云门户、云管理门户和云资源管理模块组成，在此基础上，监控、计费、审批或其他模块按需定制和扩展，通过基于Rest消息的调用进行通信，按需分布部署和增值开发，通过在逻辑层的不同模块间实现互操作增强系统的可扩展性；引入基于消息的通信方式支持异步调用，使用消息通信接口JMS传输Rest消息，使系统架构进一步解耦，在此基础上，采用面向构建设计，构件管理门户负责管理构件的元数据并对其运行状态进行监控；

[0013] 对象架构虽然能够实现按需扩展和分布部署，但属于RPC(Remote Process Call)同步通信方式，发送端等待接收端返回后才能继续执行，双方进程紧耦合，随着系统的扩大化和复杂化，构件之间的关联关系过于复杂，针对此问题，在对象架构的基础上引入基于消息的通信方式，使用消息通信接口JMS传输Rest信息，使得发送和接收端生命周期可以不同，并支持异步调用，使系统架构进一步解耦，在实施例中，门户和云资源层之间的虚拟机开启、关闭、挂起、关闭等操作通过异步方式实现，门户发出命令后无需等待响应即可返回，提升了用户交互效果；

[0014] 基于上述混合型架构的云操作系统能够满足开放兼容和扩展需求，在此基础上，基于面向构建设计思想，构件管理门户负责管理构件的元数据信息，支持注册、删除、修改和查询等操作，其中，

[0015] 构件是一个三元组包括：名称、服务集合、访问地址、描述；

[0016] 服务是一个四元组，包括：名称、类型、消息协议、参数列表、key名称、功能描述、非功能描述；

[0017] 除对构件进行描述和管理，构件管理模块进一步对其处理环境进行监控，为保障构件的可伸缩性和可用性提供基础服务，完善云操作系统的自管理能力，在构件注册时，系统为其分配用户名user、密码psw以及唯一的构件id,之后构件处理集群的接入过程为

[0018] 1)处理集群向地址为url的系统总线发起接入请求，系统总线验证接入节点的用户名、密码和id，如验证通过，建立连接，代码为：

[0019] connection＝ConnectionFactory.createConnection(user,psw,url)；

[0020] 2)建立一个写操作主题，构件的每个处理节点向该主题订阅写操作；

[0021] write_topic＝session.createTopic(id+"WRITE_TOPIC");

[0022] write_topic_consumer＝session.createConsumer(write_topic);

[0023] 3)构件在writeTopicListener的onMessage方法中实现写处理，并向系统总线注册；

[0024] write_topic_consumer.setMessageListener(writeTopicListener);

[0025] 4)根据构件处理节点数mum建立读操作队列组，每个处理节点对应一个队列订阅读操作

[0026] read_queue＝session.createMultiQueue(num,id+"READ_QUEUE");

[0027] read_queue_consumer＝session.createConsumer(read_queue);

[0028] 5)在readQueueListener的onMessage方法中实现具体的读处理功能，并向系统总线注册

[0029] read_queue_consumer.setMessageListener(readQueueListener);

[0030] 在上述读写分离队列组基础上，在构件管理模块中对各构件的服务类型进行区分，设定其为幂等或非幂，幂等操作属于无状态操作，在同一状态下每次执行的结果相同；非幂等操作属于有状态操作，同一状态下每次执行的结果不同，路由器根据服务类型进行路由，非幂操作发送到唯一的写队列，幂操作根据负载均衡策略发送到不同的读队列；

[0031] 读操作负载均衡流程过程为

[0032] 1)计算节点的处理能力；

[0033] 若节点i的CPU频率、内存容量和I/O带宽分别为C_i，M_i和B_i，集群的各种资源为节点各种资源之和，即$C＝\sum C_i$，$M＝\sum M_i$，$B＝\sum B_i$；

[0034] 则节点i的CPU权值为$W_i^{CPU}＝C_i/C$，内存容量$W_i^{RAM}＝M_i/M$，I/O带宽$W_i^{IO}＝B_i/B$；

[0035] 若构件服务所需资源比例分别为p^{CPU}，p^{RAM}，p^{IO}，则节点i的处理能力为$W_i＝p^{CPU}W_i^{CPU}+p^{RAM}W_i^{RAM}+p^{IO}W_i^{IO}$；

[0036] 2)根据读写操作权值计算各节点的负载；

[0037] 若读队列L_r同写队列L_w的读写操作开销比a，则节点i的负载$L_i＝L^R{}_i+aL^W$

[0038] 各节点的负载状态$S_i＝L_i/W_i$

[0039] 3)选择负载最轻的节点进行路由；

[0040] 写操作采用流水方式进行，以提高数据写入效率，其过程为节点1首先写入数据，写入一个数据分片64KB后，在继续接收数据的同时向节点2转发已写入的64K数据，节点2至节点n以相同方式接收和转发数据，直到节点n写入最后一个不超过64KB数据分片；

[0041] 在上述通信方法基础上，节点监控模块进一步将构件处理节点的加入、退出、失效和恢复事件发送给构件管理模块，其中，节点加入事件是指为构件添加一个处理节点；节点退出事件是指为构件撤销一个处理节点；节点失效事件是指构件的一个处理节点不可用；节点恢复事件是指构件一个不可用处理节点恢复可用。

[0042] 2、一种面向构件的混合型云操作系统的通信方法，其特征在于，包括高可用构件集群通信方法和水平伸缩构件集群通信方法，其中：

[0043] 高可用构件集群通信方法，是构件管理模块为每个构件集群建立一个读写分离队列组，构件管理模块进一步将构件的服务类型区分为幂等或非幂，路由器根据服务类型进行路由，非幂操作发送到唯一的写队列，幂操作根据负载均衡策略发送到不同的读队列，在

此基础上，节点监控模块将处理节点的加入、退出、失效和恢复事件发送给构件管理模块，构件管理模块进一步根据节点变化事件调整队列结构，此方法通过读写分离和负载均衡提高有状态构件集群的通信性能，根据节点变化调整队列结构保障构件的高可用性，构件管理模块进一步根据监控模块发送的节点变化事件调整队列结构，流程如下：

[0044] 1)当节点加入时，在队列组中为该节点建立一个读操作队列，该节点向该队列订阅读操作，并向写主题订阅写主题；

[0045] 2)当节点退出时，删除该节点对应的读队列，关闭其对写主题的订阅；

[0046] 3)当节点失效时，停止向该节点的读队列发送读请求，在写主题中为该节点保留写操作；

[0047] 4)当节点恢复时，在写主题中同步写操作，恢复向该节点对应的读队列发送读操作请求；

[0048] 无状态集群的负载均衡方法将操作完全均衡的分配到各个节点上，在有状态情况下将造成处理结果不一致，上述方法通过建立读写分离队列并针对节点能力负载均衡，能够避免此问题，提高有状态构件集群的通信性能，并且根据节点变化事件调整队列结构保障了有状态集群的高可用性；

[0049] 水平伸缩构件集群通信方法，是构件管理模块根据节点数为构件的每个服务建立多边形队列结构，按Hash值进行二分查找路由，算法效率由远小于关键字规模的Hash分组数目决定，提高了路由效率，并且在节点数发生变化的情况下只需调整部分节点数据状态，提高了动态伸缩效率，在上述通信方法基础上，在初始化阶段根据节点处理能力划分各节点所需处理的Hash区间，形成路由表，实现按比例分布数据状态，达到负载均衡；

[0050] 构件管理模块根据节点数为构件的每个服务建立多边形队列组，每个队列对应一个Hash值区间或分组，按Hash值进行二分查找路由，路由算法如下：

[0051] 1.计算key的Hash值h；

[0052] 2.初始化分组下限i为1，分组上限j为分组总数m；

[0053] 3.计算中间分组t＝(i+j)/2，查看分组t是否包含h

[0054] 4.若h小于当前分组下限t_l，更新上限j＝t-1，返回步骤3；

[0055] 5.若h大于当前分组上限t_u，更新下限i＝t+1，返回步骤3；

[0056] 6.否则，输出分组t，返回区间所在节点进行路由。

[0057] 上述算法相当于以分组为节点做二分查找，复杂度为$O(\log_2 n)$，算法的效率由分组数目决定，由于分组数目远小于关键字规模，因此提高了路由效率，另外，由于采用了分组策略，因此在节点数量发生变化的情况下只需调整部分分组的关键字规模和对应节点的数据状态，提高了动态伸缩效率。

[0058] 本发明的有益效果是：与现有技术相比，本发明提出的面向构件的混合型架构完善了开放兼容、可扩展、松耦合的构件化云操作系统体系结构，并通过构件水平伸缩集群和高可用集群的通信方法保障了云操作系统的可伸缩性和高可用性。

附图说明

[0059] 图1是面向构件的混合型COS体系结构图；

[0060] 图2是支持有状态构件集群高可用的通信架构图；

[0061] 图3是读写分离队列组示意图；

[0062] 图4是数据写入时序图；

[0063] 图5是节点状态转换图；

[0064] 图6是队列结构调整流程图；

[0065] 图7是水平伸缩构件集群通信架构图；

[0066] 图8是多边队列组路由示意图；

[0067] 图9是多边队列组路由算法流程图。

具体实施方式

[0068] 以下将结合附图及实施例来详细说明本发明的实施方式，借此对本发明如何应用技术手段来解决技术问题，并达成技术效果的实现过程能充分理解并据以实施。需要说明的是，如果不冲突，本发明实施例以及实施例中的各个特征的相互均在本发明的保护范围之内。

[0069] 1面向构件的混合型云操作系统架构

[0070] 从层次模型角度，云操作系统从上至下分为门户层、逻辑层、适配层和实现层四个层次，如图1所示。各层相对独立，通过在各层分别定义标准接口增强系统的开放性，通过在各层适配不同功能实现增强兼容性。在实施例中，通过剥离门户层可实现用户UI和功能逻辑的分离，例如，逻辑层可通过对外提供统一标准的Rest API支持门户或第三方的二次开发；通过抽象出逻辑功能层可以实现对多种具体功能实现的兼容，例如，云资源管理模块通过虚拟化适配器支持多种虚拟化基础设施，监控管理模块通过适配框架兼容多种监控协议，流程管理模块通过流程引擎支持不同审批流程的定制，计费管理和认证授权模块通过预留钩子接口支持插件接入。

[0071] 从对象模型角度，云操作系统由云门户、云管理门户、云资源管理、监控管理、计量计费、业务审批、授权认证等功能模块组成，各功能组件通过基于Rest消息的调用进行通信，可自由组合，按需分布部署，并且可根据需求增值开发新模块，通过在逻辑层的不同模块间实现互操作增强平台的可扩展性。在实施例中，最小化安装的云操作系统仅由云门户、云管理门户和云资源管理模块组成，在此基础上，监控、计费、审批或其他模块可以按需定制和扩展。

[0072] 对象架构虽然可以实现按需扩展和分布部署，但属于RPC(Remote Process Call)同步通信方式，发送端等待接收端返回后才能继续执行，双方进程紧耦合，随着系统的扩大化和复杂化，构件之间的关联关系过于复杂。针对此问题，在对象架构的基础上引入基于消息的通信方式，使用消息通信接口JMS传输Rest信息，使得发送和接收端生命周期可以不同，并支持异步调用，使系统架构进一步解耦。在实施例中，门户和云资源层之间的虚拟机开启、关闭、挂起、关闭等操作通过异步方式实现，门户发出命令后无需等待响应即可返回，提升了用户交互效果。

[0073] 基于上述混合型架构的云操作系统能够满足开放兼容和扩展需求。在此基础上，基于面向构建设计思想，构件管理门户负责管理构件的元数据信息，支持注册、删除、修改和查询等操作，其中，

[0074] 构件是一个三元组{名称、服务集合、访问地址、描述}

[0075] 服务是一个四元组{名称、类型、消息协议、参数列表、key名称、功能描述、非功能描述}

[0076] 除对构件进行描述和管理，构件管理模块进一步对其处理环境进行监控，为保障构件的可伸缩性和可用性提供基础服务，完善云操作系统的自管理能力。

[0077] 2高可用构件集群通信方法

[0078] 本发明提出高可用构件集群的实施例如图2所示，主要包括如下模块

[0079] 构件管理模块负责根据构件信息建立、删除和调整构件的消息队列组；

[0080] 消息路由器负责根据路由信息向所述队列组分发消息。

[0081] 节点监控模块负责检测构件处理集群各节点的加入和退出，获取处理节点的资源配置信息。

[0082] 构件处理集群负责实现构件的具体服务功能，由若干个处理节点组成。

[0083] 构件客户端负责发起对构件服务的使用请求。

[0084] 基于上述架构，在构件注册时，系统为其分配用户名user、密码psw以及唯一的构件id,之后构件处理集群的接入过程为

[0085] 1.处理集群向地址为url的系统总线发起接入请求，系统总线验证接入节点的用户名、密码和id，如验证通过，建立连接。代码为：

[0086] connection＝ConnectionFactory.createConnection(user,psw,url);

[0087] 2.建立一个写操作主题，构件的每个处理节点向该主题订阅写操作；

[0088] write_topic＝session.createTopic(id+"WRITE_TOPIC");

[0089] write_topic_consumer＝session.createConsumer(write_topic);

[0090] 3.构件在writeTopicListener的onMessage方法中实现写处理，并向系统总线注册；

[0091] write_topic_consumer.setMessageListener(writeTopicListener);

[0092] 4.根据构件处理节点数mum建立读操作队列组，每个处理节点对应一个队列订阅读操作

[0093] read_queue＝session.createMultiQueue(num,id+"READ_QUEUE");

[0094] read_queue_consumer＝session.createConsumer(read_queue);

[0095] 5.在readQueueListener的onMessage方法中实现具体的读处理功能，并向系统总线注册

[0096] read_queue_consumer.setMessageListener(readQueueListener);

[0097] 在上述读写分离队列组基础上，在构件管理模块中对各构件的服务类型进行区分，设定其为幂等或非幂等。幂等操作属于无状态操作，在同一状态下每次执行的结果相同；非幂等操作属于有状态操作，同一状态下每次执行的结果不同。路由器根据服务类型进行路由，非幂等操作发送到唯一的写队列，幂等操作根据负载均衡策略发送到不同的读队列，如图3所示。其中，

[0098] 读操作负载均衡流程过程为

[0099] 1.计算节点的处理能力。

[0100] 若节点i的CPU频率、内存容量和I/O带宽分别为C_i，M_i和B_i，集群的各种资源为节点各种资源之和，即$C=\sum C_i$，$M=\sum M_i$，$B=\sum B_i$。

[0101] 则节点i的CPU权值为$W_i^{CPU}=C_i/C$，内存容量$W_i^{RAM}=M_i/M$，I/O带宽$W_i^{IO}=B_i/B$。

[0102] 若构件服务所需资源比例分别为p^{CPU}，p^{RAM}，p^{IO}，则节点i的处理能力为$W_i=p^{CPU}W_i^{CPU}+p^{RAM}W_i^{RAM}+p^{IO}W_i^{IO}$。

[0103] 2.根据读写操作权值计算各节点的负载。

[0104] 若读队列L_r同写队列L_w的读写操作开销比a，则节点i的负载$L_i=L^R{}_i+aL^W$

[0105] 各节点的负载状态$S_i=L_i/W_i$

[0106] 3.选择负载最轻的节点进行路由。

[0107] 写操作采用流水方式进行，以提高数据写入效率，如图4所示，其过程为节点1首先写入数据，写入一个数据分片(64KB)后，在继续接收数据的同时向节点2转发已写入的64K数据，节点2至节点n以相同方式接收和转发数据，直到节点n写入最后一个数据分片(不超过64KB)。

[0108] 在上述通信方法基础上，节点监控模块进一步将构件处理节点的加入、退出、失效和恢复事件发送给构件管理模块，节点状态转换关系如图5所示。其中，节点加入事件是指为构件添加一个处理节点；节点退出事件是指为构件撤销一个处理节点；节点失效事件是指构件的一个处理节点不可用；节点恢复事件是指构件一个不可用处理节点恢复可用。

[0109] 构件管理模块进一步根据监控模块发送的节点变化事件调整队列结构，流程如图6所示：

[0110] 1.当节点加入时，在队列组中为该节点建立一个读操作队列，该节点向该队列订阅读操作，并向写主题订阅写主题；

[0111] 2.当节点退出时，删除该节点对应的读队列，关闭其对写主题的订阅；

[0112] 3.当节点失效时，停止向该节点的读队列发送读请求，在写主题中为该节点保留写操作；

[0113] 4.当节点恢复时，在写主题中同步写操作，恢复向该节点对应的读队列发送读操作请求。

[0114] 无状态集群的负载均衡方法将操作完全均衡的分配到各个节点上，在有状态情况下将造成处理结果不一致。上述方法通过建立读写分离队列并针对节点能力负载均衡，能够避免此问题，提高有状态构件集群的通信性能，并且根据节点变化事件调整队列结构保障了有状态集群的高可用性。

[0115] 3水平伸缩构件集群通信方法

[0116] 本发明实施例提供了一种水平伸缩构件集群通信方法，如图7所示。构件管理模块根据节点数为构件的每个服务建立多边形队列组，每个队列对应一个Hash值区间(分组)，按Hash值进行二分查找路由，路由算法为如图9所示

[0117] 1.计算key的Hash值h；

[0118] 2.初始化分组下限i为1，分组上限j为分组总数m；

[0119] 3.计算中间分组t=(i+j)/2，查看分组t是否包含h；

[0120] 4.若h小于当前分组下限t_l，更新上限j=t-1，返回步骤3；

[0121] 5.若h大于当前分组上限t_u，更新下限i=t+1，返回步骤3；

[0122] 6.否则，输出分组t，返回区间所在节点进行路由。

[0123] 上述算法相当于以分组为节点做二分查找，复杂度为$O(\log_2 n)$，算法的效率由分

组数目决定，由于分组数目远小于关键字规模，因此提高了路由效率。另外，由于采用了分组策略，因此在节点数量发生变化的情况下只需调整部分分组的关键字规模和对应节点的数据状态，提高了动态伸缩效率。

[0124] 在上述通信方法基础上，本发明实例提出一种数据状态分布方法，在初始化阶段根据节点处理能力划分各节点所需处理的Hash区间，节点处理能力的计算方法如前所述。此方法能够根据节点处理能力划分各节点的队列所需处理的Hash区间规模，实现按比例分布数据状态，达到负载均衡。

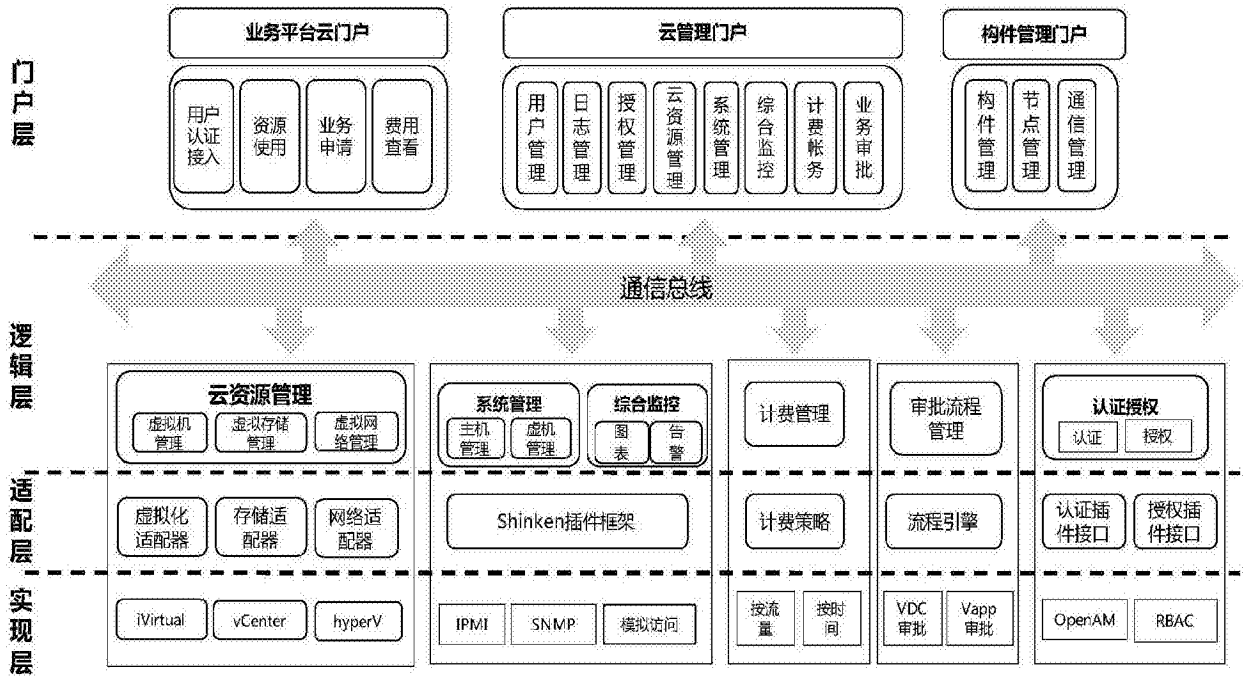

门户层
业务平台云门户
用户认证接入
资源使用
业务申请
费用查看
云管理门户
用户管理
日志管理
授权管理
云资源管理
系统管理
综合监控
计费帐务
业务审批
构件管理门户
构件管理
节点管理
通信管理
通信总线
逻辑层
云资源管理
虚拟机管理
虚拟存储管理
虚拟网络管理
系统管理
主机管理
虚机管理
综合监控
图表
告警
计费管理
审批流程管理
认证授权
认证
授权
适配层
虚拟化适配器
存储适配器
网络适配器
Shinken插件框架
计费策略
流程引擎
认证插件接口
授权插件接口
实现层
iVirtual
vCenter
hyperV
IPMI
SNMP
模拟访问
按流量
按时间
VDC审批
Vapp审批
OpenAM
RBAC

图1

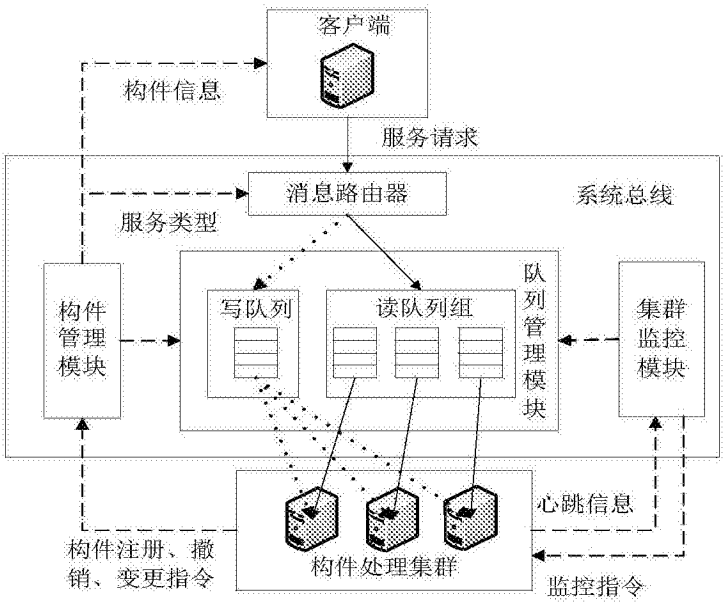

客户端
构件信息
服务请求
消息路由器
系统总线
服务类型
构件管理模块
写队列
读队列组
队列管理模块
集群监控模块
心跳信息
构件注册、撤销、变更指令
构件处理集群
监控指令

图2

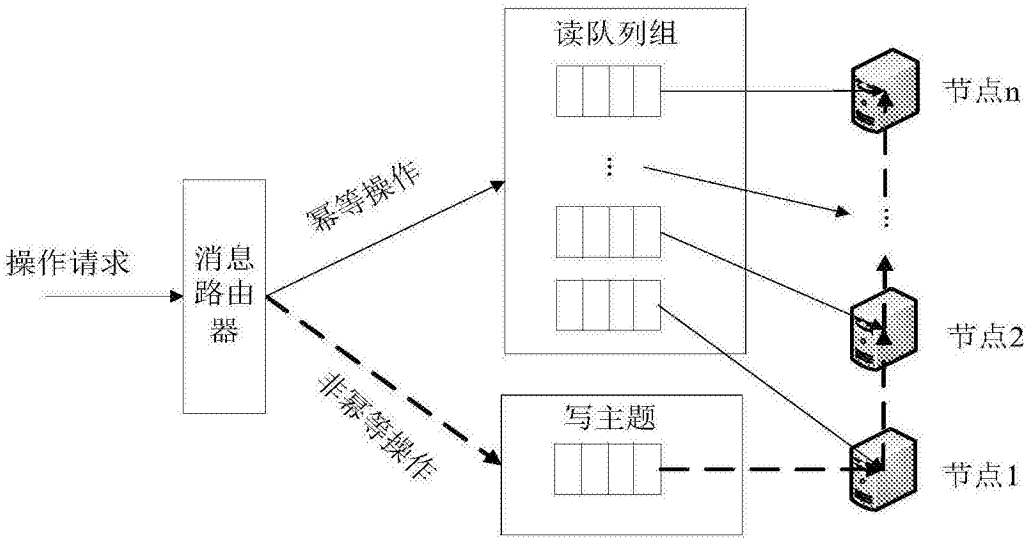

读队列组
节点n
操作请求
消息路由器
幂等操作
非幂等操作
写主题
节点2
节点1

图3

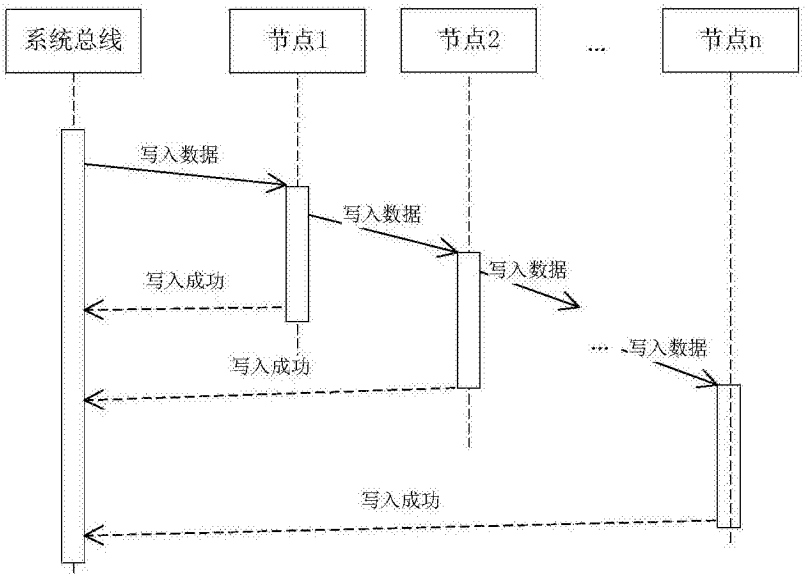

系统总线
节点1
节点2
...
节点n
写入数据
写入数据
写入数据
写入成功
写入成功
... 写入数据
写入成功

图4

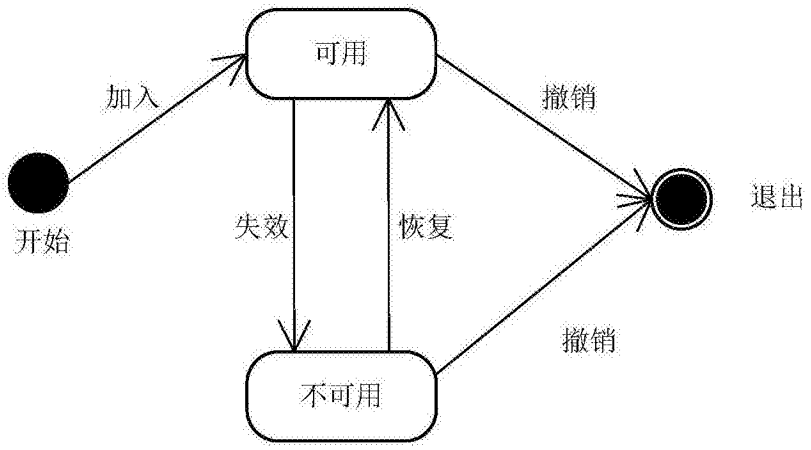

可用
加入
撤销
开始
失效
恢复
退出
撤销
不可用

图5

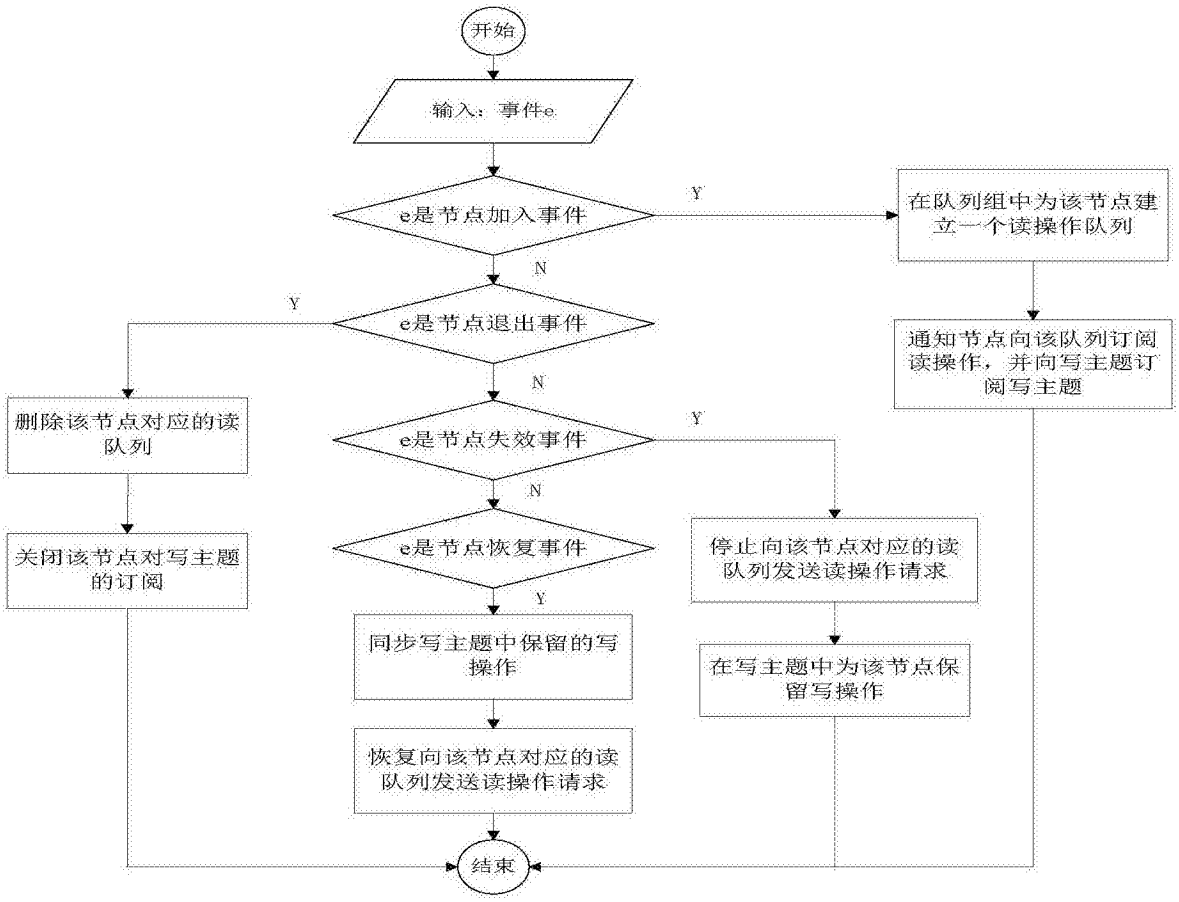

开始
输入：事件e
e是节点加入事件
Y
在队列组中为该节点建立一个读操作队列
N
e是节点退出事件
Y
通知节点向该队列订阅读操作，并向写主题订阅写主题
N
删除该节点对应的读队列
e是节点失效事件
Y
N
关闭该节点对写主题的订阅
e是节点恢复事件
停止向该节点对应的读队列发送读操作请求
Y
同步写主题中保留的写操作
在写主题中为该节点保留写操作
恢复向该节点对应的读队列发送读操作请求
结束

图6

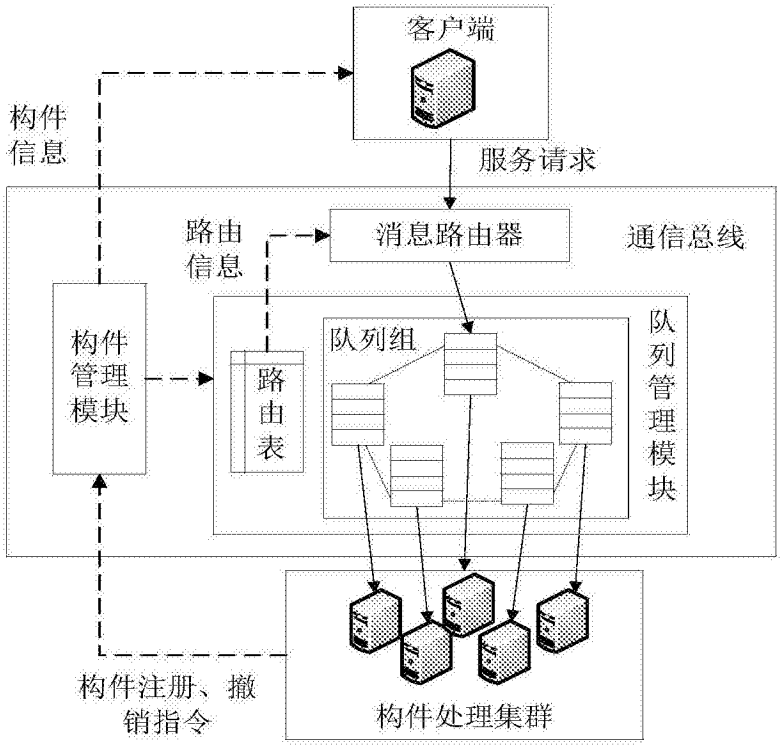

客户端
构件信息
服务请求
路由信息
消息路由器
通信总线
构件管理模块
路由表
队列组
队列管理模块
构件注册、撤销指令
构件处理集群

图7

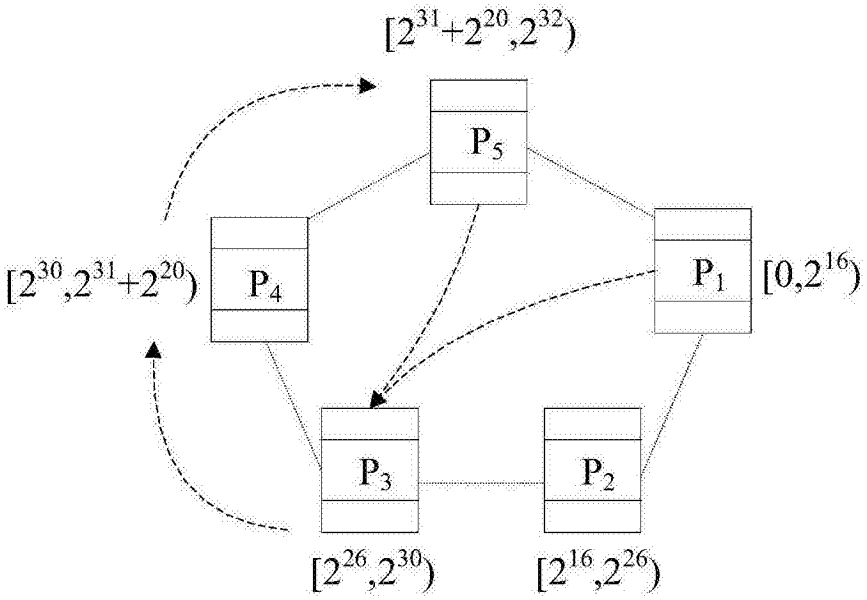

$[2^{31}+2^{20},2^{32})$
P_5
P_4
$[2^{30},2^{31}+2^{20})$
P_1
$[0,2^{16})$
P_3
P_2
$[2^{26},2^{30})$
$[2^{16},2^{26})$

图8

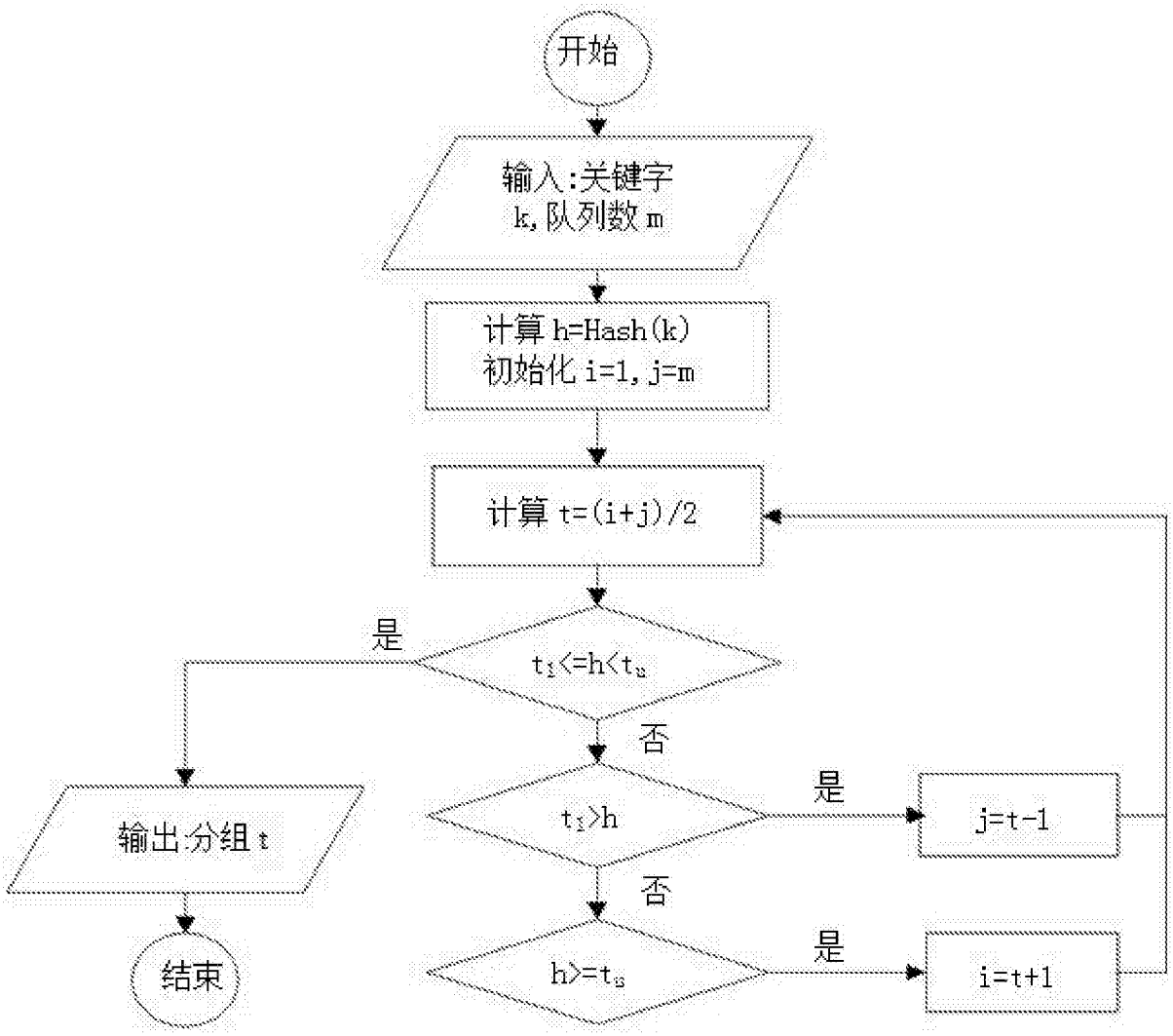

开始
输入:关键字
k,队列数 m
计算 h=Hash(k)
初始化 i=1,j=m
计算 t=(i+j)/2
是
t_1<=h<t_n
否
t_t>h
是
j=t-1
否
h>=t_n
是
i=t+1
输出:分组 t
结束

图9