

(12) 按照专利合作条约所公布的国际申请

(19) 世界知识产权组织
国际局

(43) 国际公布日
2021 年 7 月 8 日 (08.07.2021)



(10) 国际公布号
WO 2021/135545 A1

(51) 国际专利分类号:
G06F 9/48 (2006.01)

(21) 国际申请号: PCT/CN2020/123184

(22) 国际申请日: 2020 年 10 月 23 日 (23.10.2020)

(25) 申请语言: 中文

(26) 公布语言: 中文

(30) 优先权:
202010620870.4 2020年6月30日 (30.06.2020) CN

(71) 申请人: 平安科技 (深圳) 有限公司 (PING AN TECHNOLOGY (SHENZHEN) CO., LTD.) [CN/CN]; 中国广东省深圳市福田区福田街道福安社区益田路 5033 号平安金融中心 23 楼时梦如, Guangdong 518000 (CN)。

(72) 发明人: 王竹云 (WANG, Zhuyun); 中国广东省深圳市福田区福田街道福安社区益田路 5033 号平安金融中心 23 楼时梦如, Guangdong 518000 (CN)。

(74) 代理人: 深圳国新南方知识产权代理有限公司 (SHENZHEN CHINA INNOVATION SOUTH INTELLECTUAL PROPERTY AGENCY CO., LTD.); 中国广东省深圳市南山区深圳湾科技生态园 11 栋 A 座 501C 姜宇, Guangdong 518000 (CN)。

(81) 指定国 (除另有指明, 要求每一种可提供的国家保护): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK,

(54) Title: KUBERNETES MULTI-CLUSTER MANAGEMENT SYSTEM AND METHOD, TERMINAL AND STORAGE MEDIUM

(54) 发明名称: 一种 kubernetes 多集群管理系统、方法、终端及存储介质

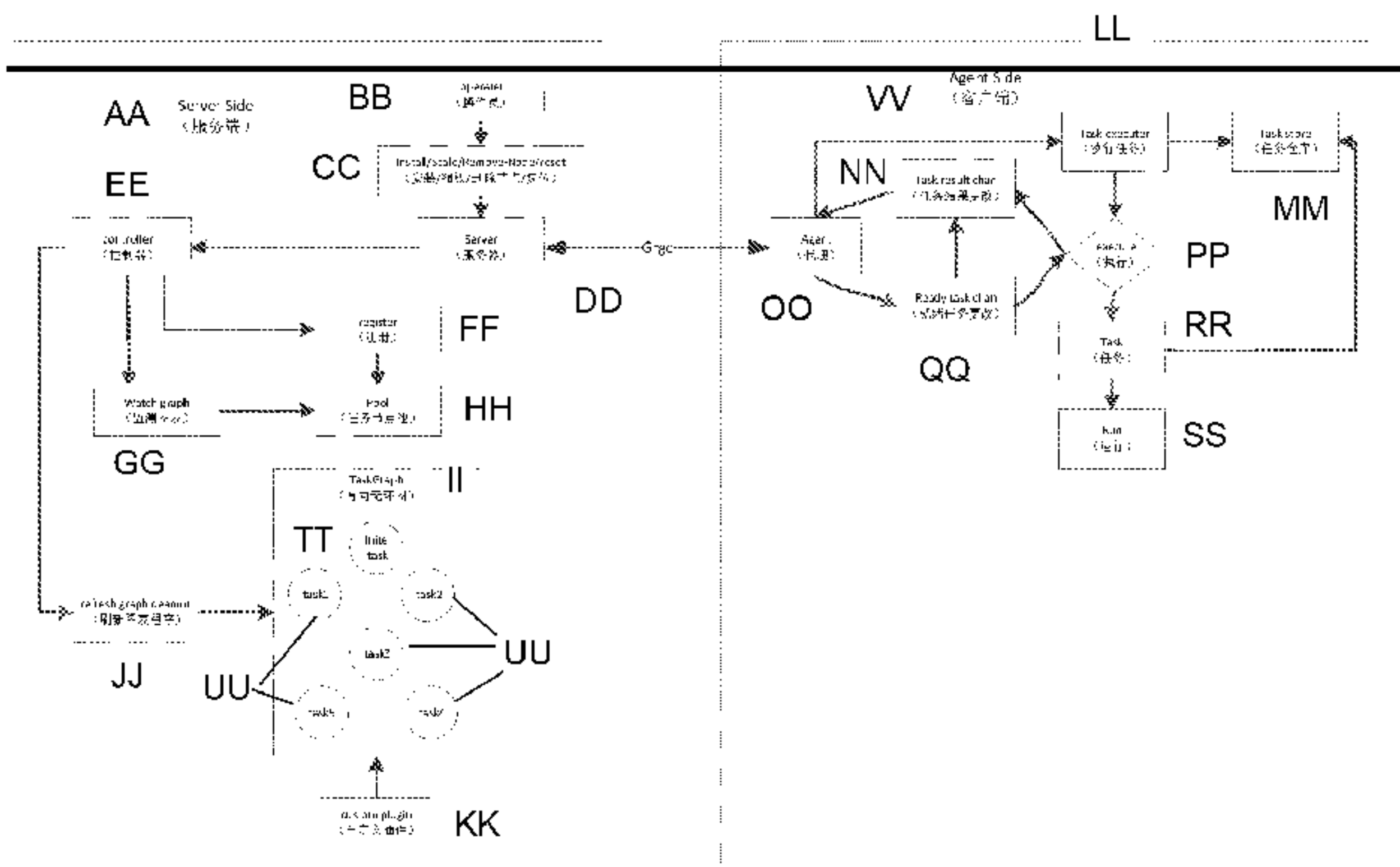


图 1

AA	Server side	LL	Task executor
BB	Operator	MM	Task store
CC	Install/Scale/Remove-Node/Reset	NN	Task result change
DD	Server	OO	Agent
EE	Controller	PP	Execute
FF	Register	QQ	Ready task change
GG	Watch graph	RR	Task
HH	Pool	SS	Run
II	TaskGraph (directed acyclic graph)	TT	Inite task
JJ	Refresh graph daemon	UU	Task
KK	Custom plugin	VV	Agent side

(57) Abstract: Provided are a kubernetes multi-cluster management system and method, a terminal and a storage medium. The multi-cluster management system is a system constructed on the basis of a distributed task scheduling framework and a gRPC protocol. The system comprises an agent side and a server side, wherein the agent side decomposes a kubernetes cluster task into a set number of tasks, and divides the tasks and task nodes according to task roles; and the server side generates, by means of a parser, a directed acyclic graph composed of tasks, and calls a controller to distribute the tasks in the directed acyclic graph to the task nodes, corresponding to

LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW。

(84) 指定国(除另有指明，要求每一种可提供的地区保护)：ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 欧亚 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 欧洲 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG)。

本国际公布：

- 包括国际检索报告(条约第21条(3))。
- 在修改权利要求的期限届满之前进行，在收到该修改后将重新公布(细则48.2(h))。
- 根据申请人的请求，在条约第21条(2)(a)所规定的期限届满之前进行。

the task roles of the tasks, in the agent side. By means of the above-mentioned system, parallel operations, such as deployment, deletion and capacity expansion and shrinkage, of a plurality of k8s clusters are realized, the operation efficiency is improved, and the state and progress of a task can be checked at any time; in addition, a connection mode where SSH is replaced by gRPC is used, such that the problem of it not being possible to use the SSH in various network environments is solved.

(57) 摘要：一种kubernetes多集群管理系统、方法、终端及存储介质，所述多集群管理系统为基于分布式任务调度框架和grpc协议构建的系统；所述系统包括客户端和服务端，所述客户端将kubernetec集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，所述服务端通过解析器生成由任务组成的有向无环图，调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点。上述系统实现了多个k8s集群并行的部署、删除、扩缩容等操作，提升了操作效率，且任务状态、进度随时可查；同时使用grpc取代ssh的连接方式，解决了ssh在诸多网络环境中无法使用的问题。

一种kubernetes多集群管理系统、方法、终端及存储介质

[0001] 本申请要求于2020年6月30日提交中国专利局、申请号为202010620870.4，发明名称为“一种kubernetes多集群管理系统、方法、终端及存储介质”的中国专利申请的优先权，其全部内容通过引用结合在本申请中。

技术领域

[0002] 本申请涉及集群管理技术领域，特别是涉及一种kubernetes多集群管理系统、方法、终端及存储介质。

背景技术

[0003] Kubernetes（简称k8s，是用8代替8个字符“ubernete”而成的缩写）是一个开源的，用于管理云平台中多个主机上的容器化的应用。虽然利用Kubernetes可以实现简单并且高效的部署容器化，然而Kubernetes的集群管理复杂度较高，尤其部署时需要对操作系统、软件进行很多的预安装、预配置的过程，对集群管理员的技术要求较高。

[0004] 目前使用较为广泛的、生产可用的Kubernetes集群管理方案为基于ansible的开源项目Kubespray。Kubespray对整个k8s集群的安装过程实现了自动化，并且部署步骤较为严谨，对系统、软件配置进行了优化，生产可用。但发明人发现Kubespray的劣势也较为明显，具体表现为：

[0005] 1. Kubespray使用ssh（Secure Shell，安全外壳协议）连接目标服务器，在某些对ssh限制较多的网络环境中，无法使用或连接缓慢；

[0006] 2. 安装过程为任务串行，任务量较多时，安装时间过长；

[0007] 3. 模版渲染工作在server端进行，然后分发至目标服务器，效率较低、网络开销较大；

[0008] 4. 只支持单集群安装；

[0009] 5. 安装过程中无法查询安装进度，且无法取消。

发明概述

技术问题

[0010] 本申请提供了一种kubernetes多集群管理系统、方法、终端及存储介质，能够在一定程度上解决现有技术中存在的不足。

问题的解决方案

技术解决方案

[0011] 为解决上述技术问题，本申请采用的技术方案为：

[0012] 一种kubernetes多集群管理系统，所述多集群管理系统为基于分布式任务调度框架和grpc协议构建的系统；所述系统包括客户端和服务端，所述客户端包括：

[0013] 任务分解模块：用于将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

[0014] 任务执行模块：用于通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端；

[0015] 所述服务端包括：

[0016] 数据解析模块：用于通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图；

[0017] 任务分发模块：用于调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任務角色相对应的任务节点。

[0018] 本申请实施例采取的另一技术方案为：一种kubernetes多集群管理方法，包括：

[0019] 通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

- [0020] 所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；
- [0021] 所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。
- [0022] 本申请实施例采取的又一技术方案为：一种终端，所述终端包括处理器、与所述处理器耦接的存储器，其中，
- [0023] 所述存储器存储有程序指令；
- [0024] 所述处理器执行所述存储器存储的所述程序指令时，执行以下步骤：
- [0025] 通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；
- [0026] 所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；
- [0027] 所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。
- [0028] 本申请实施例采取的又一技术方案为：一种存储介质，存储有处理器可运行的程序指令，所述程序指令被处理器执行时，使得处理器执行以下步骤：
- [0029] 通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；
- [0030] 所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述

客户端中与所述任务的任务角色相对应的任务节点；

[0031] 所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。

发明的有益效果

有益效果

[0032] 本申请的有益效果是：本申请实施例利用golang语言自研的分布式任务调度框架，并采用基于grpc协议的server-agent架构构建kubernetes多集群管理系统，通过将kubernetes集群任务分解成多个Task，按角色对Task及TaskNode进行划分，使得不同角色的Task分发到与之任务角色相对应的TaskNode上执行，从而实现了多个k8s集群并行的部署、删除、扩缩容等操作，提升了操作效率，且任务状态、进度随时可查；同时使用grpc取代ssh的连接方式，解决了ssh在诸多网络环境中无法使用的问题，并支持自定义plugin，增加了操作的可扩展性。

对附图的简要说明

附图说明

[0033] 图1是本申请实施例的kubernetes多集群管理系统第一结构示意图；

[0034] 图2是本申请实施例的kubernetes多集群管理系统第二结构示意图；

[0035] 图3是本申请实施例的kubernetes多集群管理方法流程示意图；

[0036] 图4是本申请实施例的终端结构示意图；

[0037] 图5是本申请实施例的存储介质结构示意图。

发明实施例

本发明的实施方式

[0038] 下面将结合本申请实施例中的附图，对本申请实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅是本申请的一部分实施例，而不是全部的实施例。基于本申请中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

[0039] 本申请中的术语“第一”、“第二”、“第三”仅用于描述目的，而不能理解

为指示或暗示相对重要性或者隐含指明所指示的技术特征的数量。由此，限定有“第一”、“第二”、“第三”的特征可以明示或者隐含地包括至少一个该特征。本申请的描述中，“多个”的含义是至少两个，例如两个，三个等，除非另有明确具体的限定。本申请实施例中所有方向性指示（诸如上、下、左、右、前、后……）仅用于解释在某一特定姿态（如附图所示）下各部件之间的相对位置关系、运动情况等，如果该特定姿态发生改变时，则该方向性指示也相应地随之改变。此外，术语“包括”和“具有”以及它们任何变形，意图在于覆盖不排他的包含。例如包含了一系列步骤或单元的过程、方法、系统、产品或设备没有限定于已列出的步骤或单元，而是可选地还包括没有列出的步骤或单元，或可选地还包括对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0040] 在本文中提及“实施例”意味着，结合实施例描述的特定特征、结构或特性可以包含在本申请的至少一个实施例中。在说明书中的各个位置出现该短语并不一定均是指相同的实施例，也不是与其它实施例互斥的独立的或备选的实施例。本领域技术人员显式地和隐式地理解的是，本文所描述的实施例可以与其它实施例相结合。

[0041] 请参阅图1，是本申请实施例的kubernetes多集群管理系统第一结构示意图。本申请利用golang语言的分布式任务调度框架，并采用基于grpc协议的Server-Agent架构构建kubernetes多集群管理系统，该系统可同时实现多个k8s集群的部署、删除、扩缩容等操作，并使用grpc替代ssh进行任务调度，解决了ssh连接受限的问题。具体的，kubernetes多集群管理系统包括Server（服务）端和Agent（客户）端，其中，Server端包括以下组件：

[0042] Task（任务）：用于描述任务元数据的结构体，任务元数据包括任务ID、名称、状态、依赖、参数、运行结果等；Task带有状态机，状态机由状态寄存器和组合逻辑电路构成，能够根据控制信号按照预先设定的状态进行状态转移，是协调相关信号动作，完成特定操作的控制中心。

[0043] TaskGraph（任务图）：由Task组成的自带状态机的有向无环图，TaskGraph用于将多个Task放入多个线程执行，并且可以设定多个Task之间的依赖关系；Task

kGraph包含每一个Task的状态、依赖关系、参数及执行结果等信息，TaskGraph及Task的状态包括not ready（未就绪）、ready（已就绪）、running（运行中）、succuss（成功）、failed（失败）及canceled（取消）等状态。

- [0044] Parser（解析器）：用于将Agent端发送的TaskFile和Inventory解析为TaskGraph；
- [0045] Controller（控制器）：用于通过状态机控制、管理TaskGraph及Task的生命周期，并将状态为ready 的Task分配给Agent端相应的TaskNode；
- [0046] Cache（存储器）：用于缓存TaskGraph；
- [0047] Pool：由TaskNode组成的任务节点池。
- [0048] Agent端包括以下组件：
- [0049] TaskNode（任务节点）：用于描述Task所运行节点的结构体，包括两个channel（通道），分别用于分发Task以及接收Task的执行结果；
- [0050] Agent（代理）：运行在TaskNode描述的真实节点上，通过grpc接口接收Server端分发的Task后执行Task，并将Task的执行结果返回给Server 端；
- [0051] TaskStore（任务仓库）：Agent端用于储存任务名称与任务函数的映射关系的任务仓库；
- [0052] TaskRole（任务角色）：用于按角色划分Task及TaskNode；
- [0053] TaskFile（任务编排文件）：用于描述Task与TaskRole的Json（JavaScript Object Notation, JS 对象简谱） 文件；
- [0054] Inventory（清单文件）：用于描述TaskNode与TaskRole的Json文件。
- [0055] 基于上述结构，本申请实施例的kubernetes多集群管理系统的操作过程具体如下：
- [0056] 首先在待部署k8s集群的节点上安装并启动Agent端，Agent端将k8s集群任务（包括但不限于部署、删除、扩缩容等）分解成设定数量的Task，通过TaskRole按角色对Task及TaskNode进行划分，使得不同角色的Task分发到与之任务角色相对应的TaskNode上，生成TaskFile及Inventory，并将TaskFile及inventory发送至Server端；
- [0057] Server端同时启动http server、grpc

server以及Controller组件，并初始化Cache；其中，http server用于接收Agent端发送的TaskFile及inventory请求；grpc server用于向Agent端分发Task，以及接收Task的执行结果。Server端接收到TaskFile及Inventory请求后，通过Paster自动解析Task的依赖关系，并转化为TaskGraph，然后调用Controller的StartGraph函数，对TaskGraph进行初始化，更新状态机，同时为TaskGraph创建一个对应的Pool，并将Inventory中的TaskNode添加到Pool中；Controller在TaskGraph启动之后，创建两个协程，一个不断刷新TaskGraph中每一个任务的状态，并将状态为Ready的任务分发给相应的TaskNode；另一个不断从Pool中的TaskNode中获取每一个任务的Task result（执行结果）。TaskGraph的状态机根据Task result更新TaskGraph中对应任务的状态。

[0058] TaskNode上的Agent获取到需要执行的Task时，从TaskStore中检索出相应的任务函数并执行任务，最后将任务执行结果通过grpc接口返回至Server端。

[0059] Server端的Controller在TaskGraph达到最终状态（Success/Failed/Canceled/Timeout）时，停止刷新TaskGraph，本次k8s部署任务结果。

[0060] 上述中，本申请实施例的多集群管理系统还包括任务编排系统和资源池系统，当Agent端提供一个Json格式的Taskfile以及Inventory时，任务编排系统自动解析Task间依赖关系，并将Task做为节点，将依赖指向做为边，构建一个Task的有向无环图；然后通过Controller按照角色将Task和资源进行划分，使得不同角色的Task调度到不同的资源上进行安装，例如将k8s的部署任务分解为kube-master-primary、kube-master-slave及kube-node三种角色，并分别进行主master、从master及负载节点的安装任务。相较于ansible而言，本申请实施例通过有向无环图实现了任务并行，并可按角色对任务进行划分，使得开发任务时无需关心任务依赖，只需关心功能的实现，随后编排成Taskfile即可实现有序调度。

[0061] 基于上述结构，本申请还可以在TaskGraph及Task的各种状态中执行自定义plugin（插件）的操作，例如发送通知、记录任务执行时间、触发另一个TaskGraph等，增加了功能的可扩展性。

- [0062] 基于上述，本申请实施例利用golang语言自研的分布式任务调度框架，并采用基于grpc协议的server-agent架构构建kubernetes多集群管理系统，实现了多个k8s集群并行的部署、删除、扩缩容等操作，提升了操作效率，且任务状态、进度随时可查；同时使用grpc取代ssh的连接方式，解决了ssh在诸多网络环境中无法使用的问题。
- [0063] 本申请实施例中，在K8s的部署过程中，利用golang的context特性，用户可通过发送信号来取消集群部署过程。
- [0064] 在K8s的部署过程，本申请可在Agent端通过模版渲染的方式生成各种yaml文件，解决了kubespray在Server端渲染后再传输给目标服务器带来的额外网络开销的问题。
- [0065] 本申请实施例的底层分布式任务调度框架本身具有通用性，可作为library被引用到其他golang项目中；同时具有可扩展性，Agent端的executor可以扩展支持kubernetes的pod，通过编排Taskfile的方式编排调度pod。基于以上两点，本申请同样适用于多种分布式批量任务调度的应用场景。
- [0066] 请参阅图2，是本申请实施例的kubernetes多集群管理系统第二结构示意图。该系统包括Server端和Agent端，其中Agent端按照于待部署k8s集群的节点上。
- [0067] 具体的，Server端包括：
- [0068] 组件启动模块：用于启动Server端，同时启动http server、grpc server以及Controller组件，并初始化Cache；其中，http server用于接收Agent端发送的TaskFile及inventory请求；grpc server用于向Agent端分发Task，以及接收Task的执行结果。
- [0069] 数据解析模块：用于在接收到Agent端的TaskFile及Inventory请求后，通过Parser自动解析Task的依赖关系，并生成TaskGraph；
- [0070] 任务分发模块：用于调用Controller的StartGraph函数对TaskTGraph进行初始化，更新状态机，同时为TaskGraph创建一个对应的Pool，并将Inventory中的TaskNode添加到Pool中；并在TaskGraph启动之后，调用Controller创建两个协程，一个不断刷新TaskGraph中每一个任务的状态，并将状态为Ready的任务分发给相应的TaskNode；另一个不断从Pool中的TaskNode中获取每一个任务的Task

k result, TaskGraph的状态机根据Task result更新TaskGraph中对应任务的状态; 其中, 当Controller监测到TaskGraph达到最终状态 (Success/Failed/Canceled/Timeout) 时, 停止刷新TaskGraph, 本次 k8s部署任务结果; 用户可通过server端的http接口查询TaskGraph的状态。

[0071] 相应地, Agent端包括:

[0072] 任务分解模块: 用于将k8s集群任务 (包括但不限于部署、删除、扩缩容等) 分解成设定数量的Task, 通过TaskRole按角色对Task及TaskNode进行划分, 使得不同角色的Task分发到与之任务角色相对应的TaskNode上, 生成TaskFile及Inventory, 并将TaskFile及inventory发送至Server端; 其中, TaskFile及Inventory生成示例如下:

[0073] TaskFile示例:

[0074] [

[0075] {

[0076] “role” : “kube-master-primary”, //任务角色

[0077] “Tasks” : [

[0078] {

[0079] “name” : “create-etcd-cert-file”, // 任务名称

[0080] “params” : {...} // 运行参数

[0081] “deps” : [...] // 依赖任务

[0082] },

[0083] {

[0084] “name” : “etcd-start”,

[0085] “params” : {...}

[0086] “deps” : []

[0087] },

[0088] {

[0089] “name” : “install-apiserver”,

[0090] “params” : {}

```
[0091]      "deps" : []
[0092]    }
[0093]    ...
[0094]  ]
[0095] }
[0096] ...
[0097] ]
[0098] Inventory示例:
[0099] [
[0100]   {
[0101]     "role": "kube-master-primary",
[0102]     "nodes": [
[0103]       "30.99.141.77"
[0104]     ]
[0105]   },
[0106]   {
[0107]     "role": "kube-master-slave",
[0108]     "nodes": [
[0109]       "30.99.141.78",
[0110]       "30.99.141.79"
[0111]     ]
[0112]   },
[0113]   {
[0114]     "role": "kube-node",
[0115]     "nodes": [
[0116]       "30.99.141.80",
[0117]       "30.99.141.81"
[0118]     ]
```

[0119] }

[0120]]

[0121] 任务执行模块：用于通过TaskNode上的Agent获取到需要执行的Task，从TaskStore中检索出相应的任务函数并执行任务，最后将任务执行结果通过grpc接口返回至Server端。

[0122] 请参阅图3，是本申请实施例的kubernetes多集群管理方法的流程示意图。本申请实施例的kubernetes多集群管理方法包括以下步骤：

[0123] S1：启动Server端，并同时启动http server、grpc server以及Controller组件，并初始化Cache；

[0124] 其中，http server用于接收Agent端发送的TaskFile及inventory请求；grpc server用于向Agent端分发Task，以及接收Task的执行结果。

[0125] S2：在待部署k8s集群的节点上安装并启动Agent端，Agent端将k8s集群任务（包括但不限于部署、删除、扩缩容等）分解成设定数量的Task，通过TaskRole按角色对Task及TaskNode进行划分，生成TaskFile及Inventory，并将TaskFile及inventory发送至Server端；

[0126] 其中，Taskfile及Inventory中的role（角色）一一对应，使得不同角色的Task可以分发到与之任务角色相对应的TaskNode上，Agent端通过POST接口将TaskFile及inventory发送至Server端。TaskFile及Inventory生成示例如下：

[0127] TaskFile示例：

[0128] [

[0129] {

[0130] “role” : “kube-master-primary”, //任务角色

[0131] “Tasks” : [

[0132] {

[0133] “name” : “create-etcd-cert-file”, // 任务名称

[0134] “params” : {...} // 运行参数

[0135] “deps” : [...] // 依赖任务

[0136] },

```
[0137]  {
[0138]      "name" : "etcd-start" ,
[0139]      "params" : {...}
[0140]      "deps" : []
[0141]  },
[0142]  {
[0143]      "name" : "install-apiserver" ,
[0144]      "params" : {}
[0145]      "deps" : []
[0146]  }
[0147]  ...
[0148]  ]
[0149]  }
[0150]  ...
[0151]  ]
[0152]  Inventory示例:
[0153]  [
[0154]      {
[0155]          "role": "kube-master-primary",
[0156]          "nodes": [
[0157]              "30.99.141.77"
[0158]          ]
[0159]      },
[0160]      {
[0161]          "role": "kube-master-slave",
[0162]          "nodes": [
[0163]              "30.99.141.78",
[0164]              "30.99.141.79"
```

[0165]]

[0166] },

[0167] {

[0168] "role": "kube-node",

[0169] "nodes": [

[0170] "30.99.141.80",

[0171] "30.99.141.81"

[0172]]

[0173] }

[0174]]

[0175] S3: Server端接收到TaskFile及Inventory请求后, 通过Paster自动解析Task的依赖关系, 并转化为TaskGraph, 然后调用Controller的StartGraph函数对TaskTGraph进行初始化, 更新状态机, 同时为TaskGraph创建一个对应的Pool, 并将Inventory中的TaskNode添加到Pool中;

[0176] S4: Controller在TaskGraph启动之后, 创建两个协程, 一个不断刷新TaskGraph中每一个任务的状态, 并将状态为Ready的任务分发给相应的TaskNode; 另一个不断从Pool中的TaskNode中获取每一个任务的Task result, TaskGraph的状态机根据Task result更新TaskGraph中对应任务的状态;

[0177] S5: TaskNode上的Agent获取到需要执行的Task时, 从TaskStore中检索出相应的任务函数执行Task, 并将执行结果通过grpc接口返回至Server端;

[0178] S6: Controller在TaskGraph达到最终状态 (Success/Failed/Canceled/Timeout) 时, 停止刷新TaskGraph, 本次 k8s部署任务结果;

[0179] 其中, 可通过server端的http接口查询TaskGraph的状态。

[0180] 请参阅图4, 为本申请实施例的终端结构示意图。该终端50包括处理器51、与处理器51耦接的存储器52。

[0181] 存储器52存储有用于实现上述kubernetes多集群管理方法的程序指令。

[0182] 处理器51执行存储器52存储的程序指令时, 执行以下步骤:

[0183] 通过客户端将kubernet集群任务分解为设定数量的任务, 按照任务角色对任

务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

[0184] 所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；

[0185] 所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。

[0186] 其中，处理器51还可以称为CPU（Central Processing Unit，中央处理单元）。处理器51可能是一种集成电路芯片，具有信号的处理能力。处理器51还可以是通用处理器、数字信号处理器（DSP）、专用集成电路（ASIC）、现成可编程门阵列（FPGA）或者其他可编程逻辑器件、分立门或者晶体管逻辑器件、分立硬件组件。通用处理器可以是微处理器或者该处理器也可以是任何常规的处理器等。

[0187] 请参阅图5，图5为本申请实施例的存储介质的结构示意图。本申请实施例的存储介质存储有能够实现上述所有方法的程序文件61，其中，该程序文件61可以以软件产品的形式存储在上述存储介质中，包括若干指令用以使得一台计算机设备（可以是个人计算机，服务器，或者网络设备等）或处理器（processor）执行以下步骤：

[0188] 通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

[0189] 所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；

- [0190] 所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。
- [0191] 而前述的存储介质可以是非易失性，也可以是易失性，存储介质包括：U盘、移动硬盘、只读存储器（ROM，Read-Only Memory）、随机存取存储器（RAM，Random Access Memory）、磁碟或者光盘等各种可以存储程序代码的介质，或者是计算机、服务器、手机、平板等终端设备。
- [0192] 本申请实施例利用golang语言自研的分布式任务调度框架，并采用基于grpc协议的server-agent架构构建kubernetes多集群管理系统，通过将kubernetes集群任务分解成多个Task，按角色对Task及TaskNode进行划分，使得不同角色的Task分发到与之任务角色相对应的TaskNode上执行，从而实现了多个k8s集群并行的部署、删除、扩缩容等操作，提升了操作效率；同时使用grpc取代ssh的连接方式，解决了ssh在诸多网络环境中无法使用的问题，并支持自定义plugin，增加了操作的可扩展性。
- [0193] 在本申请所提供的几个实施例中，应该理解到，所揭露的系统，装置和方法，可以通过其它的方式实现。例如，以上所描述的系统实施例仅仅是示意性的，例如，单元的划分，仅仅为一种逻辑功能划分，实际实现时可以有另外的划分方式，例如多个单元或组件可以结合或者可以集成到另一个系统，或一些特征可以忽略，或不执行。另一点，所显示或讨论的相互之间的耦合或直接耦合或通信连接可以是通过一些接口，装置或单元的间接耦合或通信连接，可以是电性，机械或其它的形式。
- [0194] 另外，在本申请各个实施例中的各功能单元可以集成在一个处理单元中，也可以是各个单元单独物理存在，也可以两个或两个以上单元集成在一个单元中。上述集成的单元既可以采用硬件的形式实现，也可以采用软件功能单元的形式实现。以上仅为本申请的实施方式，并非因此限制本申请的专利范围，凡是利用本申请说明书及附图内容所作的等效结构或等效流程变换，或直接或间接运用在其他相关的技术领域，均同理包括在本申请的专利保护范围内。

权利要求书

- [权利要求 1] 1. 一种kubernetes多集群管理系统，其中，所述多集群管理系统为基于分布式任务调度框架和grpc协议构建的系统；所述系统包括客户端和服务端，所述客户端包括：
- 任务分解模块：用于将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；
- 任务执行模块：用于通过所述任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端；
- 所述服务端包括：
- 数据解析模块：用于通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图；
- 任务分发模块：用于调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点。
- [权利要求 2] 2. 根据权利要求1所述的kubernetes多集群管理系统，其中，所述任务为用于描述任务元数据的结构体，所述任务带有状态机；所述有向无环图包含每一个任务的状态、依赖关系、参数及执行结果，所述每一个任务的状态分别包括未就绪、已就绪、运行中、成功、失败及取消。
- [权利要求 3] 3. 根据权利要求2所述的kubernetes多集群管理系统，其中，所述控制器用于通过状态机控制、管理所述有向无环图及任务的生命周期，并将状态为已就绪的任务分配给客户端对应的任务节点。
- [权利要求 4] 4. 根据权利要求1所述的kubernetes多集群管理系统，其中，所述服务端还包括：

组件启动模块：用于在服务端启动时，启动http server、grpc server以及控制器，并初始化存储器；其中，所述http server用于接收客户端发送的任务编排文件及清单文件；所述grpc server用于向所述客户端分发任务，并接收所述客户端返回的任务执行结果；所述存储器用于缓存所述有向无环图。

[权利要求 5] 5. 根据权利要求4所述的kubernetes多集群管理系统，其中，所述任务分发模块调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任務角色相对应的任务节点具体为：

调用所述控制器的StartGraph函数对所述有向无环图进行初始化，更新状态机，同时为所述有向无环图创建一个对应的Pool，并将所述清单文件中的任务节点添加到Pool中；所述Pool为由所述任务节点组成的任务节点池；在所述有向无环图启动之后，调用所述控制器创建两个协程，一个协程不断刷新所述有向无环图中所述任务的状态，并将状态为已就绪的任务分发给相应的任务节点；另一个协程不断从所述Pool的任务节点中获取所述任务的执行结果，所述有向无环图的状态机根据每个任务的执行结果更新所述有向无环图中所述任务的状态。

[权利要求 6] 6. 一种kubernetes多集群管理方法，其中，包括：

通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任務角色相对应的任务节点；

所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根

据任务函数执行所述任务，并将所述任务执行结果返回至服务端。

[权利要求 7] 7. 根据权利要求6所述的kubernetes多集群管理方法，其中，所述任务为用于描述任务元数据的结构体，所述任务带有状态机；

所述有向无环图包含每一个任务的状态、依赖关系、参数及执行结果，所述每一个任务的状态分别包括未就绪、已就绪、运行中、成功、失败及取消。

[权利要求 8] 8. 根据权利要求7所述的kubernetes多集群管理方法，其中，所述调用控制器将所述有向无环图中的任务分发给所述Agent端对应的任务节点包括：

调用所述控制器的StartGraph函数对所述有向无环图进行初始化，更新状态机，同时为所述有向无环图创建一个对应的Pool，并将所述清单文件中的任务节点添加到Pool中；

所述Pool为由所述任务节点组成的任务节点池；

在所述有向无环图启动之后，调用所述控制器创建两个协程，一个协程不断刷新所述有向无环图中所述任务的状态，并将状态为已就绪的任务分发给相应的任务节点；另一个协程不断从所述Pool的任务节点中获取所述任务的执行结果，所述有向无环图的状态机根据每个任务的执行结果更新所述有向无环图中所述任务的状态。

[权利要求 9] 9. 一种终端，其中，所述终端包括处理器、与所述处理器耦接的存储器，其中，

所述存储器存储有程序指令；

所述处理器执行所述存储器存储的所述程序指令时，执行以下步骤：

通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；

所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。

[权利要求 10] 10. 根据权利要求9所述的终端，其中，所述任务为用于描述任务元数据的结构体，所述任务带有状态机。

[权利要求 11] 11. 根据权利要求10所述的终端，其中，所述有向无环图包含每一个任务的状态、依赖关系、参数及执行结果，所述每一个任务的状态分别包括未就绪、已就绪、运行中、成功、失败及取消。

[权利要求 12] 12. 根据权利要求11所述的终端，其中，所述控制器用于通过状态机控制、管理所述有向无环图及任务的生命周期，并将状态为已就绪的任务分配给客户端对应的任务节点。

[权利要求 13] 13. 根据权利要求11所述的终端，其中，所述调用控制器将所述有向无环图中的任务分发给所述Agent端对应的任务节点包括：

调用所述控制器的StartGraph函数对所述有向无环图进行初始化，更新状态机，同时为所述有向无环图创建一个对应的Pool，并将所述清单文件中的任务节点添加到Pool中；

所述Pool为由所述任务节点组成的任务节点池；

在所述有向无环图启动之后，调用所述控制器创建两个协程，一个协程不断刷新所述有向无环图中所述任务的状态，并将状态为已就绪的任务分发给相应的任务节点；另一个协程不断从所述Pool的任务节点中获取所述任务的执行结果，所述有向无环图的状态机根据每个任务的执行结果更新所述有向无环图中所述任务的状态。

[权利要求 14] 14. 根据权利要求13所述的终端，其中，所述调用所述控制器的StartGraph函数对所述有向无环图进行初始化之前，还包括：

在服务端启动时，启动http server、grpc server以及控制器，并初始化存储器；其中，所述http server用于接收客户端发送的任务编排文件及清单文件；所述grpc server用于向所述客户端分发任务，并接收所述客户端返回的任务执行结果；所述存储器用于缓存所述有向无环图。

[权利要求 15] 15. 一种存储介质，其中，存储有处理器可运行的程序指令，所述程序指令被处理器执行时，使得处理器执行以下步骤：

通过客户端将kubernet集群任务分解为设定数量的任务，按照任务角色对任务及任务节点进行划分，生成任务编排文件及清单文件，并将所述任务编排文件及清单文件发送至服务端；其中，所述任务编排文件为用于描述任务与任务角色的文件，所述清单文件为用于描述任务节点与任务角色的文件，所述任务编排文件及清单文件中的任务角色一一对应；

所述服务端通过解析器对所述任务编排文件及清单文件进行解析，生成由所述任务组成的有向无环图，并调用控制器将所述有向无环图中的任务分发给所述客户端中与所述任务的任务角色相对应的任务节点；

所述客户端通过任务节点上的Agent接收所述服务端分发的任务，根据任务函数执行所述任务，并将所述任务执行结果返回至服务端。

[权利要求 16] 16. 根据权利要求15所述的存储介质，其中，所述任务为用于描述任务元数据的结构体，所述任务带有状态机。

[权利要求 17] 17. 根据权利要求16所述的存储介质，其中，所述有向无环图包含每一个任务的状态、依赖关系、参数及执行结果，所述每一个任务的状态分别包括未就绪、已就绪、运行中、成功、失败及取消。

[权利要求 18] 18. 根据权利要求17所述的存储介质，其中，所述控制器用于通过状态机控制、管理所述有向无环图及任务的生命周期，并将状态为已就绪的任务分配给客户端对应的任务节点。

[权利要求 19] 19. 根据权利要求17所述的存储介质，其中，所述调用控制器将所述

有向无环图中的任务分发给所述Agent端对应的任务节点包括：

调用所述控制器的StartGraph函数对所述有向无环图进行初始化，更新状态机，同时为所述有向无环图创建一个对应的Pool，并将所述清单文件中的任务节点添加到Pool中；

所述Pool为由所述任务节点组成的任务节点池；

在所述有向无环图启动之后，调用所述控制器创建两个协程，一个协程不断刷新所述有向无环图中所述任务的状态，并将状态为已就绪的任务分发给相应的任务节点；另一个协程不断从所述Pool的任务节点中获取所述任务的执行结果，所述有向无环图的状态机根据每个任务的执行结果更新所述有向无环图中所述任务的状态。

[权利要求 20] 20.

根据权利要求19所述的存储介质，其中，所述调用所述控制器的StartGraph函数对所述有向无环图进行初始化之前，还包括：

在服务端启动时，启动http server、grpc server以及控制器，并初始化存储器；其中，所述http server用于接收客户端发送的任务编排文件及清单文件；所述grpc server用于向所述客户端分发任务，并接收所述客户端返回的任务执行结果；所述存储器用于缓存所述有向无环图。

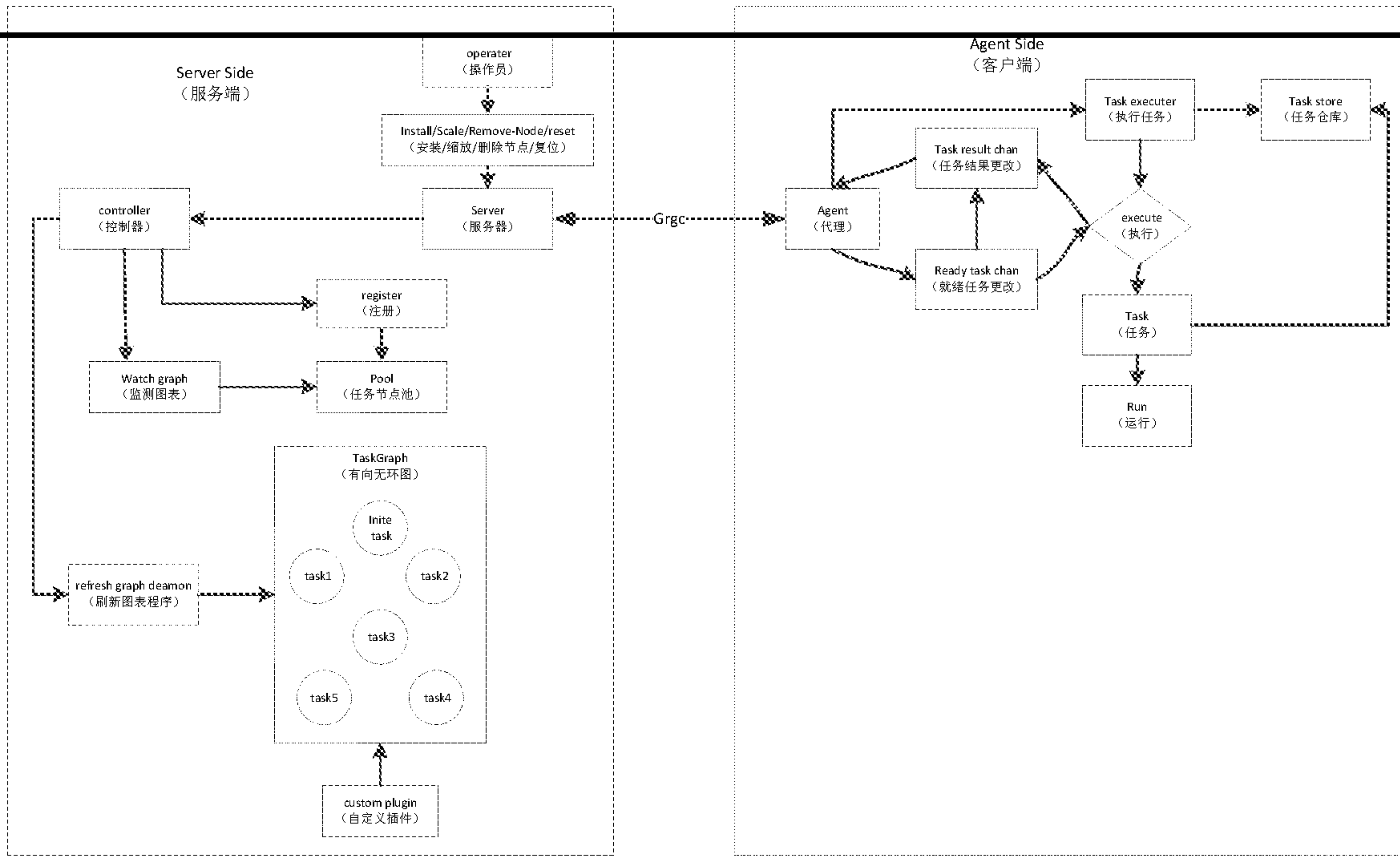


图 1

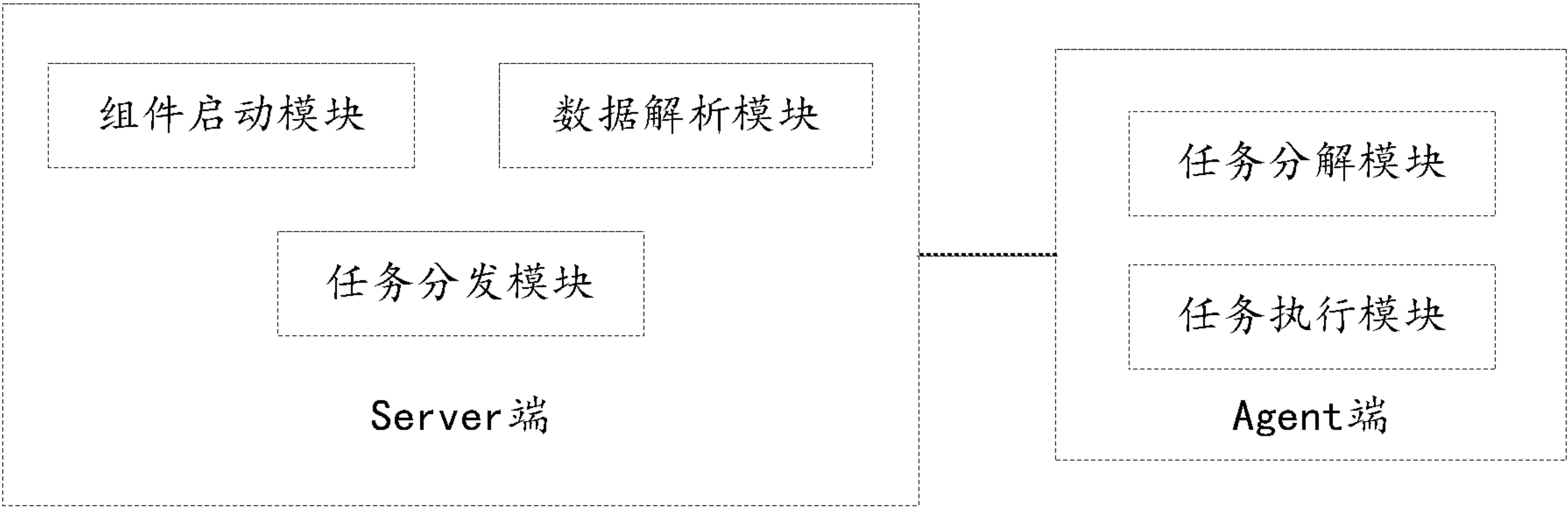


图 2

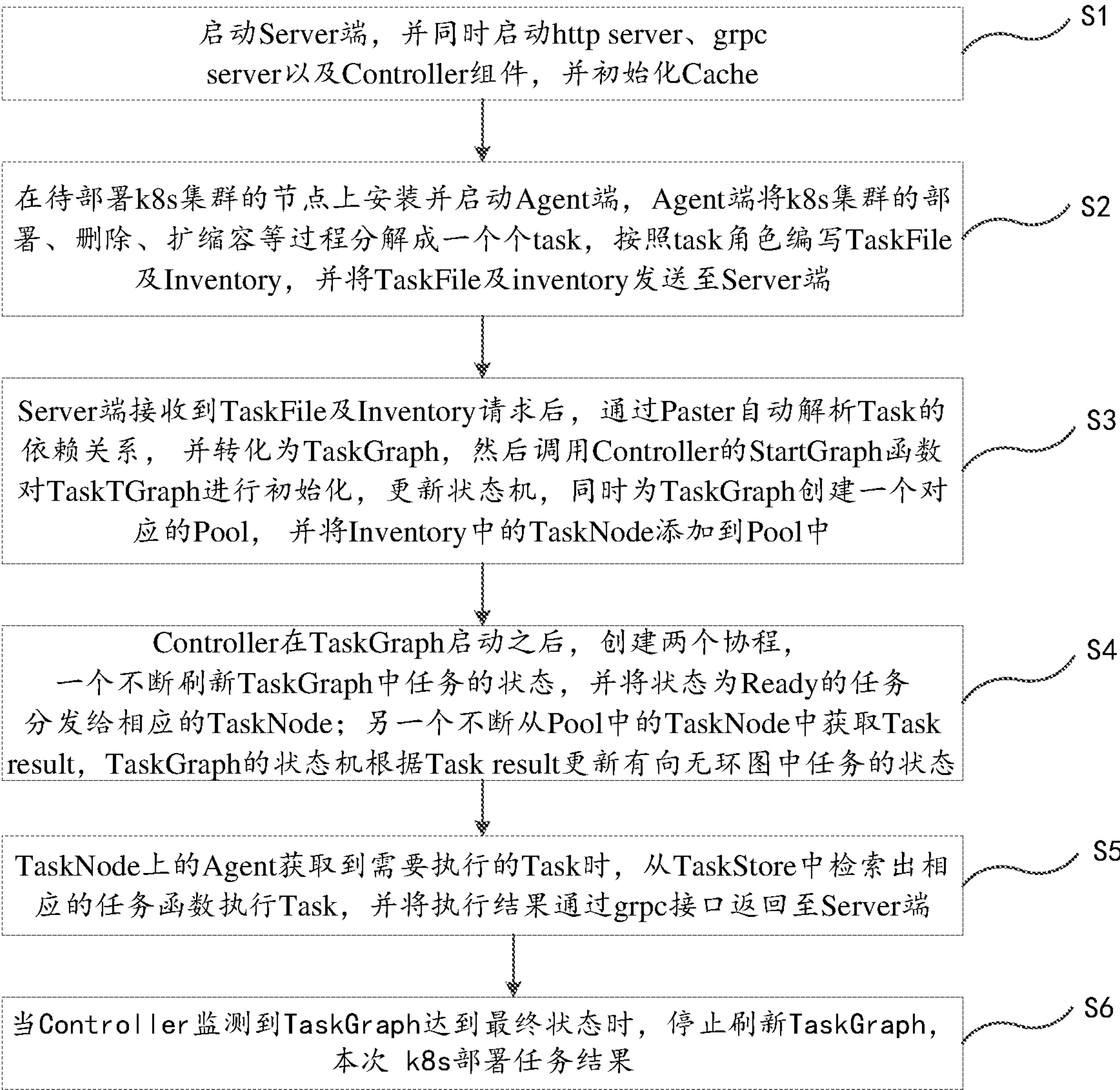


图 3

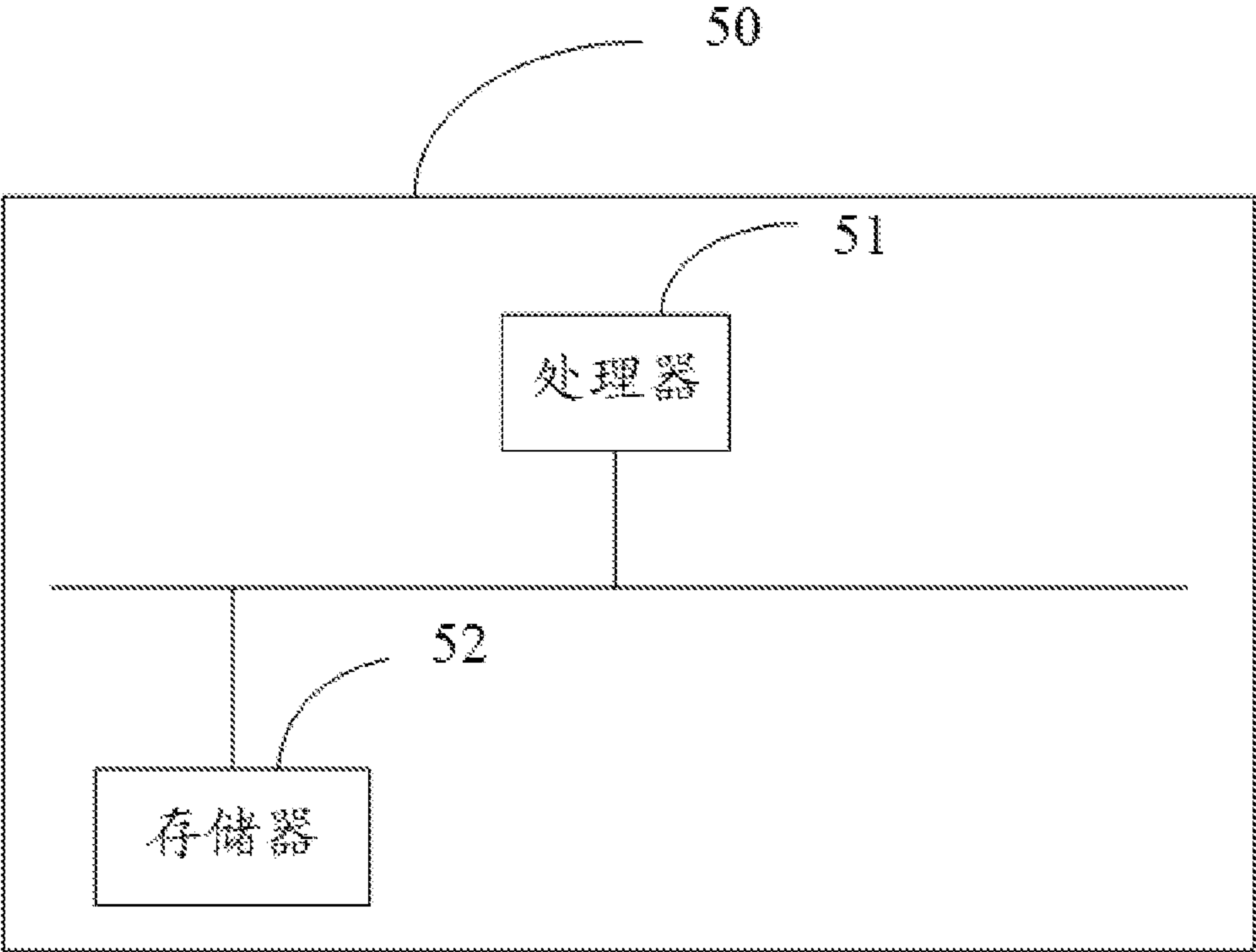


图 4

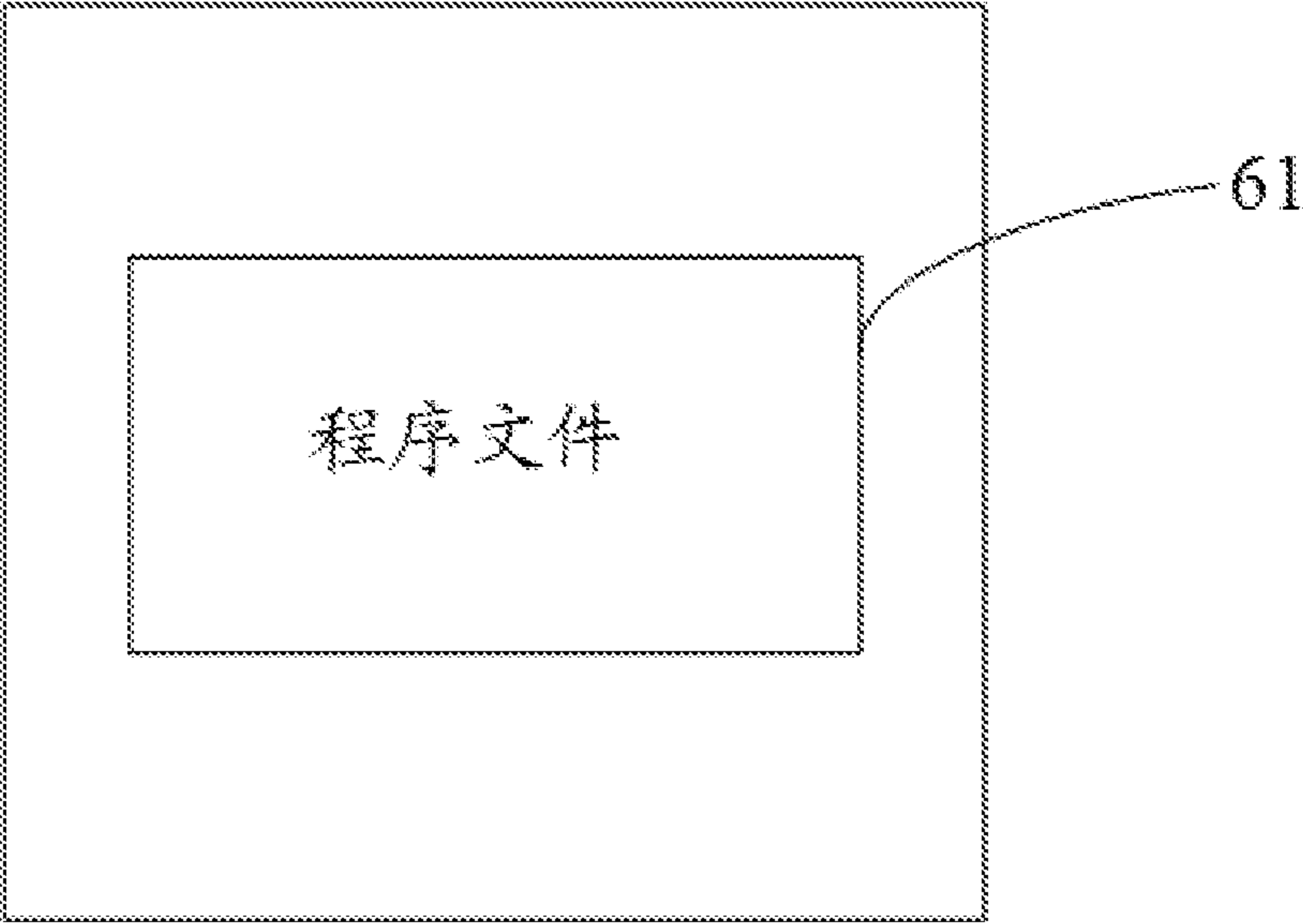


图 5

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2020/123184

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/48(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNPAT, EPODOC, WPI, CNKI, IEEE: 集群, 系统, 服务, 客户, 任务, 节点, 角色, 有向无环图, 文件, 管理器, kubernetes, k8s, system, server, agent, task, node, role, graph, file, controller

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
PX	CN 111782370 A (PING AN TECHNOLOGY (SHENZHEN) CO., LTD.) 16 October 2020 (2020-10-16) claims 1-10	1-20
A	CN 110673935 A (SUZHOU INSPUR INTELLIGENT TECHNOLOGY CO., LTD.) 10 January 2020 (2020-01-10) claims 1-10, description paragraphs [0040]-[0062]	1-20
A	CN 109842642 A (ZTE CORPORATION) 04 June 2019 (2019-06-04) entire document	1-20
A	CN 109067828 A (HANGZHOU CAIYUN TECHNOLOGY CO., LTD.) 21 December 2018 (2018-12-21) entire document	1-20
A	CN 110780998 A (WUHAN UNIVERSITY) 11 February 2020 (2020-02-11) entire document	1-20
A	US 2019391844 A1 (BEIJING BAIDU NETCOM SCIENCE AND TECHNOLOGY CO., LTD.) 26 December 2019 (2019-12-26) entire document	1-20

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:	“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
“A” document defining the general state of the art which is not considered to be of particular relevance	“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
“E” earlier application or patent but published on or after the international filing date	“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	“&” document member of the same patent family
“O” document referring to an oral disclosure, use, exhibition or other means	
“P” document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search 15 March 2021	Date of mailing of the international search report 31 March 2021
Name and mailing address of the ISA/CN China National Intellectual Property Administration (ISA/ CN) No. 6, Xitucheng Road, Jimenqiao, Haidian District, Beijing 100088 China Facsimile No. (86-10)62019451	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2020/123184

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	111782370	A	16 October 2020	None			
CN	110673935	A	10 January 2020	None			
CN	109842642	A	04 June 2019	None			
CN	109067828	A	21 December 2018	None			
CN	110780998	A	11 February 2020	None			
US	2019391844	A1	26 December 2019	CN	109491776	A	19 March 2019

国际检索报告		国际申请号 PCT/CN2020/123184
A. 主题的分类 G06F 9/48(2006.01) i 按照国际专利分类(IPC)或者同时按照国家分类和IPC两种分类		
B. 检索领域 检索的最低限度文献(标明分类系统和分类号) G06F 包含在检索领域中的除最低限度文献以外的检索文献 在国际检索时查阅的电子数据库(数据库的名称, 和使用的检索词(如使用)) CNPAT, EPDOC, WPI, CNKI, IEEE: 集群, 系统, 服务, 客户, 任务, 节点, 角色, 有向无环图, 文件, 管理器, kubernetes, k8s, system, server, agent, task, node, role, graph, file, controller		
C. 相关文件		
类 型*	引用文件, 必要时, 指明相关段落	相关的权利要求
PX	CN 111782370 A (平安科技深圳有限公司) 2020年 10月 16日 (2020 - 10 - 16) 权利要求1-10	1-20
A	CN 110673935 A (苏州浪潮智能科技有限公司) 2020年 1月 10日 (2020 - 01 - 10) 权利要求1-10, 说明书第 [0040]-[0062]段	1-20
A	CN 109842642 A (中兴通讯股份有限公司) 2019年 6月 4日 (2019 - 06 - 04) 全文	1-20
A	CN 109067828 A (杭州才云科技有限公司) 2018年 12月 21日 (2018 - 12 - 21) 全文	1-20
A	CN 110780998 A (武汉大学) 2020年 2月 11日 (2020 - 02 - 11) 全文	1-20
A	US 2019391844 A1 (BEIJING BAIDU NETCOM SCIENCE AND TECHNOLOGY CO., LTD.) 2019年 12月 26日 (2019 - 12 - 26) 全文	1-20
☐ 其余文件在C栏的续页中列出。 ☒ 见同族专利附件。		
* 引用文件的具体类型: “A” 认为不特别相关的表示了现有技术一般状态的文件 “E” 在国际申请日的当天或之后公布的在先申请或专利 “L” 可能对优先权要求构成怀疑的文件, 或为确定另一篇引用文件的公布日而引用的或者因其他特殊理由而引用的文件(如具体说明的) “O” 涉及口头公开、使用、展览或其他方式公开的文件 “P” 公布日先于国际申请日但迟于所要求的优先权日的文件 “T” 在申请日或优先权日之后公布, 与申请不相抵触, 但为了解发明之理论或原理的在后文件 “X” 特别相关的文件, 单独考虑该文件, 认定要求保护的发明不是新颖的或不具有创造性 “Y” 特别相关的文件, 当该文件与另一篇或者多篇该类文件结合并且这种结合对于本领域技术人员为显而易见时, 要求保护的发明不具有创造性 “&” 同族专利的文件		
国际检索实际完成的日期 2021年 3月 15日		国际检索报告邮寄日期 2021年 3月 31日
ISA/CN的名称和邮寄地址 中国国家知识产权局(ISA/CN) 中国北京市海淀区蓟门桥西土城路6号 100088 传真号 (86-10)62019451		授权官员 王莹 电话号码 86-(10)-53961411

国际申请号
PCT/CN2020/123184

检索报告引用的专利文件			公布日 (年/月/日)		同族专利			公布日 (年/月/日)	
CN	111782370	A	2020年 10月 16日		无				
CN	110673935	A	2020年 1月 10日		无				
CN	109842642	A	2019年 6月 4日		无				
CN	109067828	A	2018年 12月 21日		无				
CN	110780998	A	2020年 2月 11日		无				
US	2019391844	A1	2019年 12月 26日		CN	109491776	A	2019年 3月 19日	