(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 104461557 B
(45)授权公告日 2018.07.24

(21)申请号 201410804442.1

(22)申请日 2014.12.19

(65)同一申请的已公布的文献号
申请公布号 CN 104461557 A

(43)申请公布日 2015.03.25

(73)专利权人 北京奇虎科技有限公司
地址 100088 北京市西城区新街口外大街28号D座112室(德胜园区)
专利权人 奇智软件(北京)有限公司

(72)发明人 王浩宇

(74)专利代理机构 北京智汇东方知识产权代理事务所(普通合伙) 11391
代理人 康正德 孙晓芳

(51)Int.Cl.
G06F 8/71(2018.01)

(56)对比文件
CN 102438047 A,2012.05.02,全文.
CN 103944960 A,2014.07.23,全文.
CN 102004964 A,2011.04.06,全文.
王宾."Hadoop集群的部署与管理系统的设计与实现".《中国优秀硕士学位论文全文数据库信息科技辑》.2013,(第10期),第28-34页.

审查员 张昕

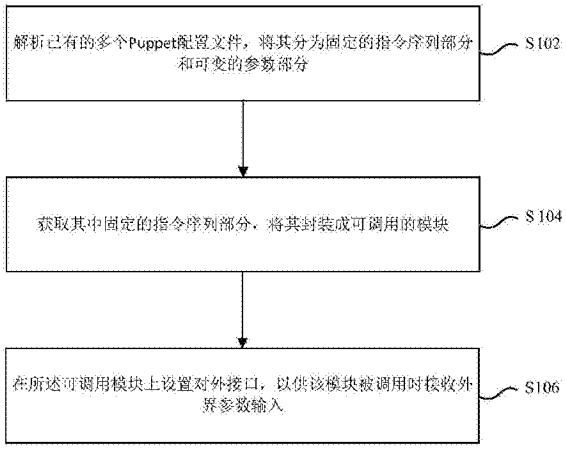
权利要求书1页 说明书7页 附图1页

(54)发明名称

Puppet配置数据的处理方法及装置

(57)摘要

本发明提供了一种Puppet配置数据的处理方法及装置。Puppet配置数据的处理方法,包括:解析已有的多个Puppet配置文件,将其分为固定的指令序列部分和可变的参数部分;获取其中所述固定的指令序列部分,将其封装成可调用的模块;以及在该模块上设置对外接口,以供所述模块被调用时接收外界参数输入。采用本发明能够增加了业务的稳定性,并能够节省配置文件生成所需的资源。



CN 104461557 B

1.一种Puppet配置数据的处理方法,包括:

解析已有的多个Puppet配置文件,将其分为固定的指令序列部分和可变的参数部分;

获取其中所述固定的指令序列部分,将其封装成可调用的模块;以及

在该模块上设置对外接口,以供所述模块被调用时接收外界参数输入。

2.根据权利要求1所述的方法,其中,还包括:

当所述模块通过所述对外接口接收到所述外界参数时,所述模块与所述外界参数合成生成配置文件。

3.根据权利要求1或2所述的方法,其中,所述模块接收外界参数输入之后,还包括:将所述外界参数格式化为多系统的通用格式,以供后续其他系统调用。

4.根据权利要求3所述的方法,其中,所述通用格式为JSON格式。

5.根据权利要求1或2所述的方法,其中,

所述固定的指令序列部分包括下列逻辑至少之一:先安装软件包,后同步配置文件;需不需要重启服务;

所述可变的参数部分包括下列至少之一:软件包的名字、版本、配置文件的路径。

6.一种Puppet配置数据的处理装置,包括:

解析组件,适于解析已有的多个Puppet配置文件,将其分为固定的指令序列部分和可变的参数部分;

封装组件,适于获取其中所述固定的指令序列部分,将其封装成可调用的模块;

接口设置组件,适于在该模块上设置对外接口,以供所述模块被调用时接收外界参数输入。

7.根据权利要求6所述的装置,其中,当所述模块通过所述对外接口接收到所述外界参数时,所述模块与所述外界参数合成生成配置文件。

8.根据权利要求6或7所述的装置,其中,所述封装组件还适于:所述模块接收外界参数输入之后,将所述外界参数格式化为多系统的通用格式,以供后续其他系统调用。

9.根据权利要求8所述的装置,其中,所述通用格式为JSON格式。

10.根据权利要求6或7所述的装置,其中,

所述固定的指令序列部分包括下列逻辑至少之一:先安装软件包,后同步配置文件;需不需要重启服务;

所述可变的参数部分包括下列至少之一:软件包的名字、版本、配置文件的路径。

Puppet配置数据的处理方法及装置

技术领域

[0001] 本发明涉及互联网应用领域，特别是涉及一种Puppet配置数据的处理方法及装置。

背景技术

[0002] puppet是一种Linux、Unix、windows平台的集中配置管理系统，使用自有的puppet描述语言，可管理配置文件、用户、cron任务、软件包、系统服务等。puppet把这些系统实体称之为资源，puppet的设计目标是简化对这些资源的管理以及妥善处理资源间的依赖关系。

[0003] 现有技术中，puppet的配置是写/etc/puppet/manifests/site.pp配置文件中的，但是，配置文件在使用过程必然是需要经常变动的，任意一个参数的变化都会引起配置文件的变化，进而需要重启主服务器(Master)。对于业务而言，重启Master所需的时间较长，若经常性重启Master，对业务本身的可持续性造成非常大的困扰，造成业务不稳定。

发明内容

[0004] 鉴于上述问题，提出了本发明以便提供一种克服上述问题或者至少部分地解决上述问题的Puppet配置数据的处理方法和相应的装置。

[0005] 基于本发明的一个方面，提供了一种Puppet配置数据的处理方法，包括：

[0006] 解析已有的多个Puppet配置文件，将其分为固定的指令序列部分和可变的参数部分；

[0007] 获取其中所述固定的指令序列部分，将其封装成可调用的模块；以及

[0008] 在该模块上设置对外接口，以供所述模块被调用时接收外界参数输入。

[0009] 可选地，Puppet配置数据的处理方法还包括：

[0010] 当所述模块通过所述对外接口接收到所述外界参数时，所述模块与所述外界参数合成生成配置文件。

[0011] 可选地，所述模块接收外界参数输入之后，还包括：将所述外界参数格式化为多系统的通用格式，以供后续其他系统调用。

[0012] 可选地，所述通用格式为JSON格式。

[0013] 可选地，所述固定的指令序列部分包括下列至少之一：先安装软件包，后同步配置文件；需不需要重启服务；

[0014] 所述可变的参数部分包括下列逻辑至少之一：软件包的名字、版本、配置文件的路径。

[0015] 基于本发明的另一个方面，提供了一种Puppet配置数据的处理装置，包括：

[0016] 解析组件，适于解析已有的多个Puppet配置文件，将其分为固定的指令序列部分和可变的参数部分；

[0017] 封装组件，适于获取其中所述固定的指令序列部分，将其封装成可调用的模块；

[0018] 接口设置组件，适于在该模块上设置对外接口，以供所述模块被调用时接收外界参数输入。

[0019] 可选地，当所述模块通过所述对外接口接收到所述外界参数时，所述模块与所述外界参数合成生成配置文件。

[0020] 可选地，所述封装组件还适于：所述模块接收外界参数输入之后，将所述外界参数格式化为多系统的通用格式，以供后续其他系统调用。

[0021] 可选地，所述通用格式为JSON格式。

[0022] 可选地，所述固定的指令序列部分包括下列逻辑至少之一：先安装软件包，后同步配置文件；需不需要重启服务；

[0023] 所述可变的参数部分包括下列至少之一：软件包的名字、版本、配置文件的路径。

[0024] 在本发明实施例中，将Puppet配置文件划分为两部分，一部分是固定的指令序列部分(即静态参数)，另一部分是可变的参数部分(即动态参数)，静态参数和动态参数分开，在使用时将两者结合，容易形成自动化控制，使用起来省时省力。将两部分划分开之后，将固定的指定序列部分封装成可调用模块，并在其上设置对外接口，以供该模块被调用时接收外界参数的输入。随后，将外界参数匹配到对应的指令中，生成配置文件。在本发明实施例中，固定的指令序列部分封装成模块后是固定存在的，对于主服务器而言，即配置文件主体是固定存在的，并未出现变化。其变化的仅仅是通过对外接口输入的外界参数，配置文件主体不变，那么，主服务器不需要重启，避免因多次重启耗时所造成的业务不稳定的情况，能够对业务提供稳定的支持。另外，因指令序列部分被封装为可调用模块，对于配置文件自身而言，这部分指令序列部分可以直接调用，不需要重新利用代码写入多个指令序列，配置文件的准确性、可靠性以及配置文件的生成效率都大大提高，尤其是对于复杂的指令序列，效果大大增加。由于模块可调用性的便利，进一步增加了业务的稳定性，并能够节省配置文件生成所需的资源。

[0025] 上述说明仅是本发明技术方案的概述，为了能够更清楚了解本发明的技术手段，而可依照说明书的内容予以实施，并且为了让本发明的上述和其它目的、特征和优点能够更明显易懂，以下特举本发明的具体实施方式。

[0026] 根据下文结合附图对本发明具体实施例的详细描述，本领域技术人员将会更加明了本发明的上述以及其他目的、优点和特征。

附图说明

[0027] 通过阅读下文优选实施方式的详细描述，各种其他的优点和益处对于本领域普通技术人员将变得清楚明了。附图仅用于示出优选实施方式的目的，而并不认为是对本发明的限制。而且在整个附图中，用相同的参考符号表示相同的部件。在附图中：

[0028] 图1示出了根据本发明一个实施例的Puppet配置数据的处理方法的处理流程图；

[0029] 图2示出了根据本发明一个实施例的Puppet配置数据的处理装置的结构示意图。

具体实施方式

[0030] 下面将参照附图更详细地描述本公开的示例性实施例。虽然附图中显示了本公开的示例性实施例，然而应当理解，可以以各种形式实现本公开而不应被这里阐述的实施例

所限制。相反，提供这些实施例是为了能够更透彻地理解本公开，并且能够将本公开的范围完整的传达给本领域的技术人员。

[0031] 为解决上述技术问题，本发明实施例提供了一种Puppet配置数据的处理方法。首先介绍puppet的配置文件结构。下述部分是puppet的配置文件的一个示例：

[0032] **node node001v.add.zwt.qihoo.net {**

[0033]

```
        package { 'nginx':
            ensure => '1.2.9',
        }
        file { '/usr/local/nginx/cong/nginx.conf':
            source => 'puppet:///module/nginx/nginx.conf',
            owner => 'root',
            group => 'root',
            mode => '0644',
            notify => Service['nginx'], # nginx will restart whenever you edit
this file.
            require => Package['nginx'],
        }
        service { 'nginx':
            ensure => running,
            enable => true,
            hasstatus => true,
            hasrestart => true,
        }
    }
```

[0034] 上面的一段配置是主服务器(Master)指导node001v.add.zwt.qihoo.net同步的动作序列，其包含的配置步骤如下：

[0035] 1、先安装前端(nginx)软件包，要求版本必须是1.2.9；

[0036] 2、在软件包安装之后，同步配置文件nginx.conf，并配置为：如果配置文件有变化，通知Master重启；

[0037] 3、如有必要，即当确有配置文件变化时，重启Master。

[0038] 请参考图1，其示出了根据本发明一个实施例的Puppet配置数据的处理方法的处理流程图。参见图1，该方法包括步骤S102、步骤S104以及步骤S106。

[0039] 步骤S102、解析已有的多个Puppet配置文件，将其分为固定的指令序列部分和可变的参数部分。

[0040] 步骤S104、获取其中固定的指令序列部分，将其封装成可调用的模块。

[0041] 步骤S106、在所述可调用模块上设置对外接口，以供该模块被调用时接收外界参

数输入。

[0042] 在本发明实施例中，将Puppet配置文件划分为两部分，一部分是固定的指令序列部分(即静态参数)，另一部分是可变的参数部分(即动态参数)，静态参数和动态参数分开，在使用时将两者结合，容易形成自动化控制，使用起来省时省力。将两部分划分开之后，将固定的指定序列部分封装成可调用模块，并在其上设置对外接口，以供该模块被调用时接收外界参数的输入。随后，将外界参数匹配到对应的指令中，生成配置文件。在本发明实施例中，固定的指令序列部分封装成模块后是固定存在的，对于主服务器而言，即配置文件主体是固定存在的，并未出现变化。其变化的仅仅是通过对外接口输入的外界参数，配置文件主体不变，那么，主服务器不需要重启，避免因多次重启耗时所造成的业务不稳定的情况，能够对业务提供稳定的支持。另外，因指令序列部分被封装为可调用模块，对于配置文件自身而言，这部分指令序列部分可以直接调用，不需要重新利用代码写入多个指令序列，配置文件的准确性、可靠性以及配置文件的生成效率都大大提高，尤其是对于复杂的指令序列，效果大大增加。由于模块可调用性的便利，进一步增加了业务的稳定性，并能够节省配置文件生成所需的资源。

[0043] 主服务器多次重启，除了耗时之外，对业务有着重大的影响。例如，客户端正连接主服务器进行数据下载，主服务器突然重启，客户端数据未下载完成，甚至于已下载的数据也会丢失，更为严重的可能造成客户端硬件或软件损坏。若对于同一用户，每次主服务器重启后，该用户需要重新与主服务器建立连接才能够继续获取数据，若主服务器需要账号登录，则该用户还需要每次输入账号信息进行认证，操作步骤繁琐。由此可见，多次主服务器重启所造成的恶劣影响足以让用户放弃这一业务，甚至会认为该业务是恶意程序。因此，本发明实施例提供的Puppet配置数据的处理方法的处理流程图不仅仅增加业务的稳定性，还能够提升用户的感受体验。

[0044] 在本发明实施例中，步骤S106通过对外接口接收到外界参数时，模块利用其中封装的指令序列与外界参数合成生成配置文件。例如，可以将指令序列封装到一个模块中，将其称为nginx。与相关技术中提及的配置文件的示例相对比，版本、配置文件的路径被视为可变参数，将其抽离。剩余的指令序列为固定的指令序列部分，将其封装为nginx模块。nginx模块包含运行逻辑，比如：先安装软件包，后同步配置文件，需不需要重启服务等等，都封装在了模块中，使用时不需要去关注或获取。当版本为1.2.9，配置文件的路径为“/usr/local/nginx/cong/nginx.conf”的外界参数通过对外接口输入nginx模块时，其生成的配置文件与相关技术中的配置文件相同。但是，相关技术中的配置文件无法再行变化，若变化就需要重启主服务器。而本实施例中的nginx模块可以通过更改可变参数，以生成包含新参数的配置文件，无须重启主服务器。本实施例中的两个参数仅仅是示意，在实际应用中，每个模块的可变参数根据其模块的运行逻辑以及实际的可变参数的数量决定，在此不做赘述。

[0045] 在一个优选的实施例中，模块接收外界参数输入之后，可以将外界参数格式化为多系统的通用格式，以供后续其他系统调用。本实施例充分开发Puppet的Hiera功能，将模块的参数格式化到单独的通用文件，既实现了配置文件的拆分，保持配置文件的稳定，又可以不重启Master，而通用格式的数据结构又便于二次开发，为多系统的集成奠定了基础，便于实现更高程度的自动化。优选的，通用格式为JSON格式。JSON(Javascript Object

Notation)是一种轻量级的数据交换语言,以文字为基础,且易于让人阅读。

[0046] 具体的,一个JSON格式的文件的示意如下:

[0047]

```
[root@puppet01v ~]# cat/etc/puppet/hiera/node/tianji01v.add.bjdt.json
{
    "classes":[
        "nginx",
    ],
    "nginx::package_name":"addops-nginx",
    "nginx::package_ensure":"1.2.9-4.el6",
    "nginx::nginx_config":"1656_9\/nginx.conf",
}
```

[0048] 该JSON格式文件意义为:

[0049] 1、调用一个模块“nginx”;

[0050] 2、为该模块赋予3个参数,分别是:软件包的名字、版本、配置文件的路径。

[0051] 本实施例提供的系统集成是指应用系统集成(Application System Integration),以系统的高度为客户需求提供应用的系统模式,以及实现该系统模式的具体技术解决方案和运作方案,即为用户提供一个全面的系统解决方案。应用系统集成已经深入到用户具体业务和应用层面,在大多数场合,应用系统集成又称为行业信息化解决方案集成。应用系统集成可以说是系统集成的高级阶段,独立的应用软件供应商将成为核心。本发明实施例利用JSON的通用性,将参数格式化为JSON文件,本系统和其他系统均可以调用,为系统高度集成提供了方便。

[0052] 基于同一发明构思,本发明实施例还提供了一种Puppet配置数据的处理装置,用于支持上述任意一个优选实施例或其组合所提供的Puppet配置数据的处理方法。图2示出了根据本发明一个实施例的Puppet配置数据的处理装置的结构示意图。参见图2,该装置至少包括:

[0053] 解析组件210,适于解析已有的多个Puppet配置文件,将其分为固定的指令序列部分和可变的参数部分;

[0054] 封装组件220,与解析组件210耦合,适于获取其中固定的指令序列部分,将其封装成可调用的模块;

[0055] 接口设置组件230,与封装组件220耦合,适于在该模块上设置对外接口,以供所述模块被调用时接收外界参数输入。

[0056] 在一个优选的实施例中,当模块通过对外接口接收到外界参数时,模块与外界参数合成生成配置文件。

[0057] 在一个优选的实施例中,封装组件220还适于:模块接收外界参数输入之后,将外界参数格式化为多系统的通用格式,以供后续其他系统调用。

[0058] 在一个优选的实施例中,通用格式为JSON格式。

[0059] 在一个优选的实施例中,固定的指令序列部分包括下列逻辑至少之一:先安装软

件包，后同步配置文件；需不需要重启服务；

[0060] 可变的参数部分包括下列至少之一：软件包的名字、版本、配置文件的路径。

[0061] 采用本发明实施例提供的Puppet配置数据的处理方法及装置能够达到如下有益效果：

[0062] 在本发明实施例中，将Puppet配置文件划分为两部分，一部分是固定的指令序列部分(即静态参数)，另一部分是可变的参数部分(即动态参数)，静态参数和动态参数分开，在使用时将两者结合，容易形成自动化控制，使用起来省时省力。将两部分划分开之后，将固定的指定序列部分封装成可调用模块，并在其上设置对外接口，以供该模块被调用时接收外界参数的输入。随后，将外界参数匹配到对应的指令中，生成配置文件。在本发明实施例中，固定的指令序列部分封装成模块后是固定存在的，对于主服务器而言，即配置文件主体是固定存在的，并未出现变化。其变化的仅仅是通过对外接口输入的外界参数，配置文件主体不变，那么，主服务器不需要重启，避免因多次重启耗时所造成的业务不稳定的情况，能够对业务提供稳定的支持。另外，因指令序列部分被封装为可调用模块，对于配置文件自身而言，这部分指令序列部分可以直接调用，不需要重新利用代码写入多个指令序列，配置文件的准确性、可靠性以及配置文件的生成效率都大大提高，尤其是对于复杂的指令序列，效果大大增加。由于模块可调用性的便利，进一步增加了业务的稳定性，并能够节省配置文件生成所需的资源。

[0063] 在此处所提供的说明书中，说明了大量具体细节。然而，能够理解，本发明的实施例可以在没有这些具体细节的情况下实践。在一些实例中，并未详细示出公知的方法、结构和技术，以便不模糊对本说明书的理解。

[0064] 类似地，应当理解，为了精简本公开并帮助理解各个发明方面中的一个或多个，在上面对本发明的示例性实施例的描述中，本发明的各个特征有时被一起分组到单个实施例、图、或者对其的描述中。然而，并不应将该公开的方法解释成反映如下意图：即所要求保护的本发明要求比在每个权利要求中所明确记载的特征更多的特征。更确切地说，如下面的权利要求书所反映的那样，发明方面在于少于前面公开的单个实施例的所有特征。因此，遵循具体实施方式的权利要求书由此明确地并入该具体实施方式，其中每个权利要求本身都作为本发明的单独实施例。

[0065] 本领域那些技术人员可以理解，可以对实施例中的设备中的模块进行自适应性地改变并且把它们设置在与该实施例不同的一个或多个设备中。可以把实施例中的模块或单元或组件组合成一个模块或单元或组件，以及此外可以把它们分成多个子模块或子单元或子组件。除了这样的特征和/或过程或者单元中的至少一些是相互排斥之外，可以采用任何组合对本说明书(包括伴随的权利要求、摘要和附图)中公开的所有特征以及如此公开的任何方法或者设备的所有过程或单元进行组合。除非另外明确陈述，本说明书(包括伴随的权利要求、摘要和附图)中公开的每个特征可以由提供相同、等同或相似目的的替代特征来代替。

[0066] 此外，本领域的技术人员能够理解，尽管在此所述的一些实施例包括其它实施例中所包括的某些特征而不是其它特征，但是不同实施例的特征的组合意味着处于本发明的范围之内并且形成不同的实施例。例如，在权利要求书中，所要求保护的实施例的任意之一都可以以任意的组合方式来使用。

[0067] 本发明的各个部件实施例可以以硬件实现，或者以在一个或者多个处理器上运行的软件模块实现，或者以它们的组合实现。本领域的技术人员应当理解，可以在实践中使用微处理器或者数字信号处理器(DSP)来实现根据本发明实施例的Puppet配置数据的处理装置中的一些或者全部部件的一些或者全部功能。本发明还可以实现为用于执行这里所描述的方法的一部分或者全部的设备或者装置程序(例如，计算机程序和计算机程序产品)。这样的实现本发明的程序可以存储在计算机可读介质上，或者可以具有一个或者多个信号的形式。这样的信号可以从因特网网站上下载得到，或者在载体信号上提供，或者以任何其他形式提供。

[0068] 应该注意的是上述实施例对本发明进行说明而不是对本发明进行限制，并且本领域技术人员在不脱离所附权利要求的范围的情况下可设计出替换实施例。在权利要求中，不应将位于括号之间的任何参考符号构造成对权利要求的限制。单词“包括”或“包含”不排除存在未列在权利要求中的元件或步骤。位于元件之前的单词“一”或“一个”不排除存在多个这样的元件。本发明可以借助于包括有若干不同元件的硬件以及借助于适当编程的计算机来实现。在列举了若干装置的单元权利要求中，这些装置中的若干个可以是通过同一个硬件项来具体体现。单词第一、第二、以及第三等的使用不表示任何顺序。可将这些单词解释为名称。

[0069] 至此，本领域技术人员应认识到，虽然本文已详尽示出和描述了本发明的多个示例性实施例，但是，在不脱离本发明精神和范围的情况下，仍可根据本发明公开的内容直接确定或推导出符合本发明原理的许多其他变型或修改。因此，本发明的范围应被理解和认定为覆盖了所有这些其他变型或修改。

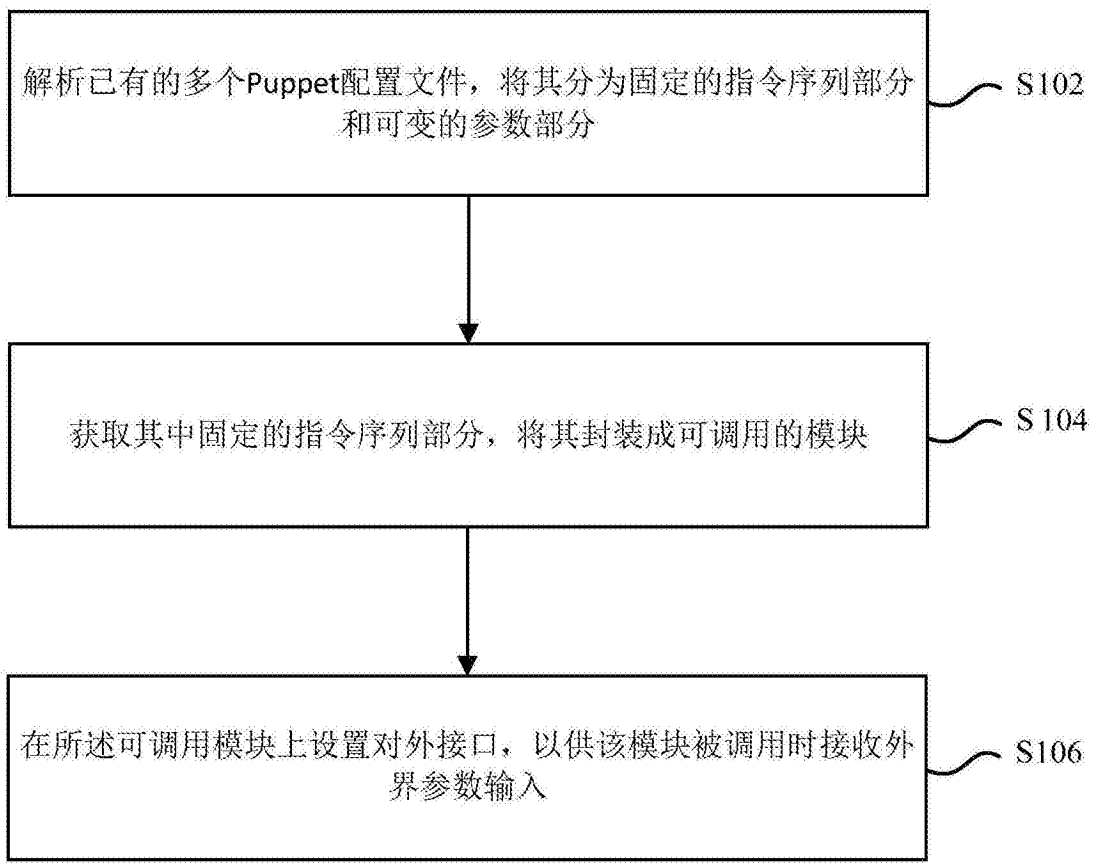

解析已有的多个Puppet配置文件，将其分为固定的指令序列部分和可变的参数部分
S102
获取其中固定的指令序列部分，将其封装成可调用的模块
S 104
在所述可调用模块上设置对外接口，以供该模块被调用时接收外界参数输入
S106

图1

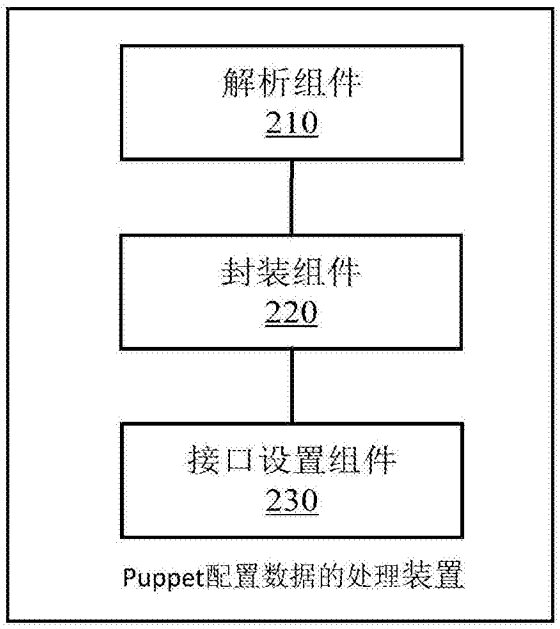

解析组件
210
封装组件
220
接口设置组件
230
Puppet配置数据的处理装置

图2